Lidar-Based Obstacle Detection and Recognition for Autonomous Agricultural Vehicles

Mikkel Fly Kragh

PhD
Department of Engineering
Aarhus University
2018

ISBN: 978-87-7507-435-8
DOI: 10.7146/aul.288.202

Preface

This thesis would not have been possible without support from a great number of people. First, I would like to thank my main supervisor Rasmus Nyholm Jørgensen for offering me the PhD position and for always seeing (financial) possibilities instead of limitations. You have always defended our interests and helped out during planning and execution of difficult field trials. Thanks to my co-supervisor Henrik Karstoft. You have always been available for technical guidance, discussions, and constructive feedback, and you have been a great help in soliciting my next position. I look forward to our continuous collaboration. Also a huge thanks to my previous co-supervisor Henrik Pedersen for the guidance and for facilitating previous, current, and future positions in the field of computer vision.
I would further like to thank my supervisor at the Australian Centre for Field Robotics, James Underwood, for his great guidance, technical know-how, and willingness to brainstorm and discuss new ideas even years after our collaboration. I have enjoyed every minute of the research stay, and I sincerely hope to cross paths again in the future.
Thanks to all my great colleagues, Anders, Martin, Mads, Søren, Simon, and Rene for continuous feedback and technical discussions. Thanks to Morten Larsen and Morten Laursen for your help in choosing, interfacing, and debugging sensors and hardware for the perception platform. A special thanks to my project partner and friend Peter Christiansen for great collaboration and helpful discussions.
Last, but certainly not least, a special thanks to my loving wife, Astrid. This work would not have been possible without your endless support, patience, and understanding through the entire study. Thank you for joining me on the research stay abroad, and thank you for making it all run smoothly at home when I have been unpredictable and buried in work.

This work is sponsored by the Innovation Fund Denmark as part of the project SAFE - Safer Autonomous Farming Equipment (project no. 16-2014-0).

Aarhus, March 2018 Mikkel Fly Kragh

Abstract

Today, agricultural vehicles are available that can drive autonomously and follow exact route plans more precisely than human operators. Combined with advancements in precision agriculture, autonomous agricultural robots can reduce manual labor, improve workflow, and optimize yield. However, as of today, human operators are still required for monitoring the environment and acting upon potential obstacles in front of the vehicle. To eliminate this need, safety must be ensured by accurate and reliable obstacle detection and avoidance systems.

In this thesis, lidar-based obstacle detection and recognition in agricultural environments has been investigated. A rotating multi-beam lidar generating 3D point clouds was used for point-wise classification of agricultural scenes, while multi-modal fusion with cameras and radar was used to increase performance and robustness. Two research perception platforms were presented and used for data acquisition. The proposed methods were all evaluated on recorded datasets that represented a wide range of realistic agricultural environments and included both static and dynamic obstacles.

For 3D point cloud classification, two methods were proposed for handling density variations during feature extraction. One method outperformed a frequently used generic 3D feature descriptor, whereas the other method showed promising preliminary results using deep learning on 2D range images. For multi-modal fusion, four methods were proposed for combining lidar with color camera, thermal camera, and radar. Gradual improvements in classification accuracy were seen, as spatial, temporal, and multi-modal relationships were introduced in the models. Finally, occupancy grid mapping was used to fuse and map detections globally, and runtime obstacle detection was applied on mapped detections along the vehicle path, thus simulating an actual traversal.

The proposed methods serve as a first step towards full autonomy for agricultural vehicles. The study has thus shown that recent advancements in autonomous driving can be transferred to the agricultural domain, when accurate distinctions are made between obstacles and processable vegetation. Future research in the domain has further been facilitated with the release of the multi-modal obstacle dataset, FieldSAFE.

Resume

Automatiserede landbrugsmaskiner kan allerede i dag køre autonomt og følge ruteplaner mere præcist end mennesker. Ved hjælp af præcisionslandbrug og automatisering kan selvkørende landbrugsmaskiner reducere manuelt arbejde, forbedre det daglige workflow og samtidig optimere udbyttet af afgrøder. Der er dog stadig brug for menneskelige operatører til at monitorere omgivelserne og reagere på eventuelle forhindringer foran et køretøj. For at opnå fuldstændigt selvkørende maskiner er der derfor brug for et sikkerhedssystem, der sikrer præcis og pålidelig detektion og håndtering af forhindringer.

I denne afhandling er lidar-baseret detektion og genkendelse af forhindringer i landbrug blevet undersøgt og præsenteret. En roterende lidar, der genererer 3D-punktskyer ved hjælp af adskillige lasere, har været brugt til punktvis klassificering af forhindringer og strukturer, som er typiske for landbrug. Lidar-teknologien blev desuden kombineret med kamera og radar for at øge både præcision og robusthed. Til opsamling af data blev der præsenteret to sensorplatforme samt tilhørende datasæt, der indeholdt både statiske og dynamiske forhindringer. De udviklede metoder er alle blevet evalueret på baggrund af de optagne datasæt, der repræsenterede en bred vifte af realistiske landbrugsmiljøer.

Til klassificering af 3D-punktskyer blev der præsenteret to metoder, der begge adresserede og håndterede varierende punktdensitet ved beregning af features. Den ene metode udkonkurrerede en ofte anvendt generisk feature-deskriptor, mens den anden metode viste lovende foreløbige resultater ved at anvende deep learning på såkaldte 2D-afstandsbilleder. Lidar-teknologi blev desuden kombineret med farvekamera, termisk kamera og radar gennem fire forskellige metoder. Sammen viste de, at tilføjelse af spatielle, temporale og multimodale sammenhænge gradvist forbedrede klassifikationsraten. Slutteligt blev der præsenteret en metode til at kortlægge detektioner globalt, mens forhindringer blev detekteret langs køretøjets bane for at simulere en reel kørsel.

De udviklede metoder repræsenterer et første skridt i retningen mod selvkørende landsbrugsmaskiner. Studiet har således vist, at nylige fremskridt inden for selvkørende biler kan overføres til landbrugsdomænet, så længe der skelnes mellem forhindringer og afgrøder. Udgivelsen af et multimodalt datasæt til detektion af forhindringer i landbrug muliggør desuden fremtidig forskning inden for området.

Contents

Part I Introduction

Autonomous robots and vehicles are emerging in numerous fields of work. Within the next decade, fully autonomous cars are likely to drive the streets on both highways, urban roads, country lanes, and even dirt roads without any human intervention (Litman, 2017). To ensure safety of both passengers and surrounding traffic, advanced perception systems sense the environment, perform scene understanding, and detect, map, and track obstacles that are near the planned path of the vehicle.

In agriculture, a fleet of small, autonomous field robots can reduce manual labor, optimize yield, distribute workload, and reduce soil compaction (Blackmore et al., 2009; Gebbers and Adamchuk, 2010). The explicitly constructed environments allow for predefined route plans that are optimized for fuel consumption and yield. For the past two decades, autonomous operation has been possible using automated steering systems that follow route plans more precisely than human operators (Abidine et al., 2004). However, autonomous agricultural vehicles need reliable obstacle detection and avoidance systems to ensure safety. Such systems must use a complementary set of perception sensors to increase accuracy and avoid single points of failure. An exponential growth in sensor revenues is thus predicted for robotic vehicles "equipped with a suite of sensors encompassing lidars, radars, cameras, inertial measurement units (IMUs) and Global Navigation Satellite Systems (GNSS)" (Cambou et al., 2018). Lidar is an acronym of light detection and ranging and uses time-of-flight of reflected laser pulses to measure distances. Lidar and radar are both active range sensors that provide distance measurements useful for detecting obstacles based on geometry, whereas passive camera sensors such as color and thermal cameras provide visual clues useful for discriminating object classes. GNSS is a generic term for satellite navigation systems including the Global Positioning System (GPS). Here, the two terms are used interchangeably. IMU and GNSS enable accurate localization required for mapping and avoiding obstacles. Each modality can thus contribute with different physical quantities, and combining modalities with sensor fusion can potentially increase detection performance and provide redundancy.

Obstacle detection and avoidance for agricultural robots has been addressed in a few industrial R&D projects and in multiple research projects. In 2016, Case IH presented their autonomous concept vehicle (Case IH, 2016) with a perception system by Autonomous Solutions including lidar and color camera (ASI, 2016). The generic perception system has further been used for obstacle detection in orchards and vineyards. In scientific research, the CASC project combined a low-cost laser scanner with local navigation sensors to detect and avoid various obstacles from point clouds in orchard environments (Freitas et al., 2012; Bergerman et al., 2012). The same research group has since then investigated stereo vision for human detection (Tabor et al., 2015; Pezzementi et al., 2017) and a combination of lidar and color camera for general obstacle detection in orchards (Moorehead et al., 2012). The QUAD-AV project has explored multiple sensors and fusion approaches for obstacle detection in agricultural environments (Reina et al., 2016a). Methods were proposed for traversability assessment by applying stereo vision and by fusing lidar and stereo. Furthermore, radar and stereo vision were fused for obstacle detection, whereas stereo and thermal imaging were combined for obstacle recognition.

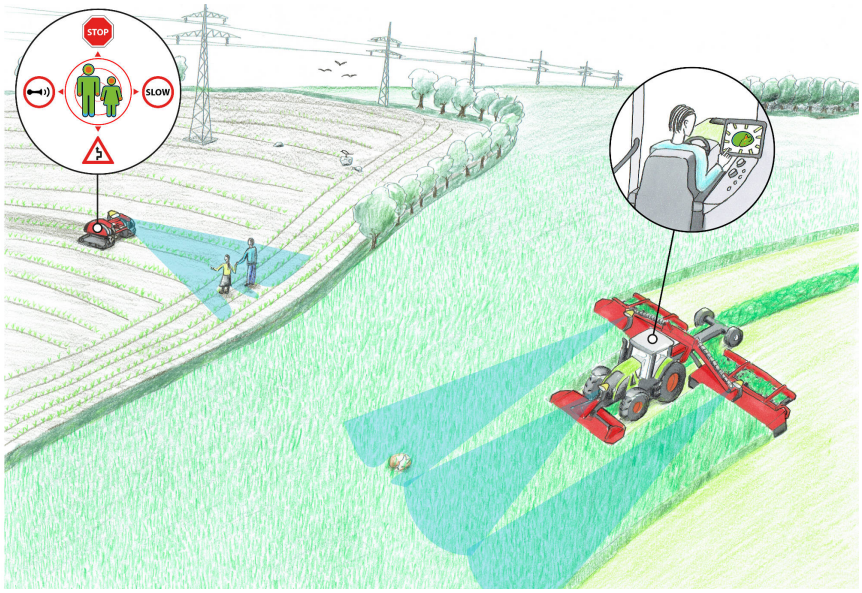


Figure 1.1: Safety scenarios for farming vehicles. Illustration by Bertelsen Design.

The current study is a part of the Safer Autonomous Farming Equipment (SAFE) project, a joint research collaboration between two agricultural machine manufacturers, AgroIntelli and CLAAS, a robotic consulting firm, Conpleks Innovation, and two research institutions, Aarhus University and the University of Southern Denmark. The SAFE project seeks to explore technologies for maximizing the safety of both humans and animals with autonomous farming vehicles, while minimizing the workload and supervision needed by farmers. Figure 1.1 illustrates possible safety-related scenarios. The project addresses all technical disciplines required for full autonomy such as data acquisition, obstacle detection, sensor fusion, localization, mapping, planning, and control. Divided into a number of work packages, one part of the project deals with the perception system, whereas others deal with behavior, automation, and control.

In this thesis, only the perception part of the above pipeline is addressed. This includes data acquisition, obstacle detection, and to some extent object localization and mapping. The presented work focuses on how 3D point clouds acquired with a rotating, multi-beam lidar can be used for obstacle detection and recognition. Methods are proposed for object classification using 1) lidar alone, and 2) lidar combined with other sensing modalities such as color camera, thermal camera, and radar. The main objective of the study is thus to investigate the following two research questions:

1. *How can obstacles be recognized in sparse 3D point clouds from a rotating multi-beam lidar?*
2. *How can lidar technology cooperate with other sensing modalities in agricultural environments to increase object recognition performance?*

1.1 Scope

Obstacle detection and recognition is a crucial part of any robot operating autonomously either indoors or outdoors. This thesis, however, addresses the specific problem of obstacle detection and recognition for autonomous farming vehicles. To some extent, agricultural vehicles must deal with the same scenarios as autonomous cars. They must thus be able to detect other vehicles and pedestrians as well as static objects such as trees, buildings, fences and poles. However, the perception system of an agricultural vehicle does not need to detect and recognize traffic lights, lane lines, and traffic signs. As a farming vehicle should only operate in a closed and known environment, the range of possible scenarios thus seems rather limited. On the other hand, simple traversability assessment based on height differences may suffice for an autonomous car on a paved road, but prove insufficient in agriculture. Tall grass or crops may thus be traversable and processable although protruding from the ground, while objects obscured by or hidden within vegetation are not.

The methods proposed in the study all focus on point- and pixel-wise classification, either directly in sensor frames or in a global map. This is commonly referred to as semantic segmentation and serves as a generic representation that allows for subsequent clustering, tracking, or further fusion with other modalities. For object detection and obstacle avoidance, 2D or 3D bounding box representations are typically used to describe object location and size. However, some structures such as vegetation, a fence, or the sky cannot be represented with bounding boxes. For these categories, semantic segmentation is capable of describing both position and shape. However, semantic segmentation, in local sensor frames or global maps, does not describe if a system or algorithm will ensure safety. To address the important issue of safety, actual use cases of agricultural machines need to be taken into account. For some tasks such as fruit harvesting, it is normal to have humans operating close by automated vehicles, whereas for other tasks such as crop harvesting, any human in close vicinity is considered a high risk. Similarly, in some environments such as grass fields, a vehicle may pass detected obstacles, whereas in others such as row crops, the vehicle must stop in front of obstacles, as deviations from the planned path can damage plants and crops. By recognizing object classes, vehicle behavior can further distinguish between static obstacles (that must be passed) and dynamic obstacles (that may be told to move aside).

Operating speeds and braking distances further impose requirements on minimum detection distances and update frequencies. For instance, working speeds up to $25\,\mathrm{km\,h^{-1}}$ and working widths up to 12 m are common for grass mowing. Braking distance d can be calculated as $d = \frac{v^2}{2\mu g}$ with v, μ, and g denoting velocity, coefficient of friction, and gravitational acceleration (Noon, 1994). A worst-case stopping distance for locked-wheel braking on wet grass ($\mu = 0.2$) is thus 12.3 m, when zero reaction time is assumed for the perception system. For downhill operation, the distance will be even larger. This means that all obstacles must be detected at least 12.3 m in front of the vehicle at the full working width of 12 m. Methods such as object tracking require even larger detection dis-

tances, whereas an intelligent system may choose to slow down the vehicle (and thereby decrease its braking distance) in case of uncertainty. These concerns have all been a part of the SAFE project, in which other work packages have dealt with risk management, tracking, behavior, automation, and control. This thesis, however, only deals with the problem of detecting obstacles by post-processing recorded scenarios.

1.2 Contributions

This thesis presents a number of methods for lidar-based obstacle detection and recognition in agricultural environments. Two approaches classify point clouds from a lidar alone, while four other approaches investigate multi-modal fusion of lidar with color camera, thermal camera, and radar. The proposed methods and their experimental results are summarized in part II-V of the thesis and documented in detail in the scientific publications listed in Table 1.1. Primary publications from the list are attached at the back of the thesis in part VII.

The presented data material and proposed methods all include specific contributions and novelties that are outlined in the publications, individually. The combined work, however, addresses three core problems in lidar-based obstacle detection in agriculture. The main contributions of the study are:

- A multi-modal dataset for obstacle detection in agriculture including GPS-based annotations of static and dynamic obstacles. The dataset, called FieldSAFE, has been made publicly available at https://vision.eng.au.dk/fieldsafe/.
- An online-applicable method for joint semantic segmentation on 3D point clouds and 2D color images. The method fuses lidar and camera data with a conditional random field by introducing spatial, temporal, and multi-modal relationships. Compared to individual sensor performances, the fusion method has shown to increase classification performance in both modalities on multiple agricultural datasets.
- A semi-automated procedure for producing large-scale GPS-based annotations. The procedure assigns class labels to georeferenced lidar points based on manually annotated drone-acquired orthophotos. The method has been shown to enable training of a deep neural network for multi-modal fusion of lidar, color camera, and thermal camera.

Table 1.1: List of publications. Primary publications are attached in part VII, secondary publications are project-relevant but not attached, whereas tertiary publications have been published during the course of the study but are unrelated to the project.

Primary publications (attached)	**Type**	**Author**	**State**
Advanced sensor platform for human detection and protection in autonomous farming (Paper 1; Christiansen et al., 2015)	Conference	Second	Published
Platform for evaluating sensors and human detection in autonomous mowing operations (Paper 2; Christiansen et al., 2017)	Journal	Second	Published
FieldSAFE: Dataset for Obstacle Detection in Agriculture (Paper 3; Kragh et al., 2017)	Journal	First (joint)	Published
Object Detection and Terrain Classification in Agricultural Fields using 3D Lidar Data (Paper 4; Kragh et al., 2015)	Conference	First	Published
Multi-Modal Obstacle Detection in Unstructured Environments with Conditional Random Fields (Paper 5; Kragh and Underwood, 2017)	Journal	First	Submitted 1st rev.
Multi-Modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture (Paper 6; Kragh et al., 2016b)	Conference	First	Published
Multi-Modal Detection and Mapping of Static and Dynamic Obstacles in Agriculture for Process Evaluation (Paper 7; Korthals et al., 2018)	Journal	First (joint)	Accepted
Towards Inverse Sensor Mapping in Agriculture (Paper 8; Korthals et al., 2017b)	Conference	Second	Published
Multi-Modal Semantic Segmentation in 3D with Range Images (Paper 9; Kragh et al., 2018)	Journal	First	Draft
Secondary publications (project-related)			
Self-Supervised Traversability Assessment in Field Environments with Lidar and Camera (Kragh et al., 2016c)	Conference (poster)	First	Published
Towards a DSL for Perception-Based Safety Systems (Ingibergsson et al., 2015)	Conference	Third	Published
Tertiary publications (unrelated)			
3D impurity inspection of cylindrical transparent containers (Kragh et al., 2016a)	Journal	First	Published
Automatic behaviour analysis system for honeybees using computer vision (Tu et al., 2016)	Journal	Second	Published

1.3 Reading Guide

The remainder of this thesis is organized as follows:

Part II presents the data material used for developing and evaluating methods. Two multi-modal research perception platforms are described along with associated datasets.

Part III presents two methods for point-wise classification of lidar-acquired 3D point clouds that both address varying point density and local point neighborhoods.

Part IV presents three multi-modal fusion methods for combining lidar point clouds with color and thermal images to improve classification performance.

Part V describes an approach for fusion, localization, and global mapping of obstacle detections from multiple modalities.

Part VI discusses and concludes the contributions of the study, relates to the end-goal of full autonomy, and suggests future work.

Part VII includes scientific papers published and submitted during the course of the study. These are the primary publications from Table 1.1.

Data Material Part II

It is unlikely that a single sensor can acquire all the necessary information required to perform robust and accurate detection, recognition, and positioning of obstacles. Therefore, sensing modalities are often combined with sensor fusion to ensure overlapping field of views as well as complementing detection capabilities. Objects that protrude from the ground can likely be detected with range sensors such as a lidar or radar, whereas objects residing or hiding in high grass or crops may require the use of color cameras and thermal cameras to be detected. Therefore, geometry alone may not suffice to distinguish traversable areas from non-traversable areas. Similarly, appearance cues may help in classifying specific object categories, but can also fail in detecting visually ambiguous structures such as grass, trees, and bushes, or animals appearing visually camouflaged in their natural habitat. Additionally, most sensors are affected by varying weather and illumination conditions and may easily lead to false or missing detections. Multi-modal perception systems are therefore necessary to address all possible conditions and safety scenarios.

For data-driven approaches such as machine learning, annotated datasets are crucial for both training and testing methods and models. State-of-the-art approaches within image classification, object localization and recognition, and semantic segmentation have all been trained on datasets specifically relevant for their purpose. Therefore, agricultural obstacle detection methods must be trained and tested on representative datasets that include not only relevant objects, but also realistic environments.

Today, a number of publicly available datasets exist within research on urban autonomous driving. Some focus on behavioral cloning (reproducing vehicle control actions from perception input), whereas others explicitly address object detection, recognition, and localization. In scientific research, the KITTI dataset (Geiger et al., 2013) is considered the standard for benchmarking object detection and localization methods using both single- and multi-modal approaches. Since its release in 2013, more and more annotated data have been made available for researchers to train and test their methods. Recently, a number of simulated datasets have further been published that use popular computer graphic engines to generate synthetic sensor data (e.g. images and point clouds) including automated annotations for all scenes and objects (Ros et al., 2016; Gaidon et al., 2016; Yue et al., 2017).

Agricultural environments, however, deviate significantly from urban environments. Where urban driving often involves planar surfaces, lane lines, and traffic signs, an agricultural environment is typically unstructured or semi-structured. Here, tall grass or crops may actually be traversable and processable although protruding from the ground, whereas objects residing or hiding within vegetation are non-traversable. Urban driving datasets thus do not include the necessary scenarios to fully cover all agricultural use cases. Datasets that address obstacle detection in agriculture are therefore needed.

The Marulan datasets (Peynot et al., 2010) provide multi-modal sensor data from rural environments with various obstacles present. The datasets further include a wide range of challenging environmental conditions such as rain, smoke, and dust. However,

the datasets primarily include static obstacles, and limited ground truth data are available. The National Robotics Engineering Center (NREC) Agricultural Person-Detection Dataset (Pezzementi et al., 2017) includes images of human obstacles in orange and apple orchards. The dataset contains ground truth annotations of multiple image sequences with pedestrians, but only includes a single perception sensor (stereo camera). A need therefore still exists for multi-modal datasets in various agricultural environments with both static and moving obstacles. Table 1.2 lists and compares existing datasets in autonomous driving and agriculture. A more detailed review of existing datasets is further available in Paper 3 (Kragh et al., 2017).

Table 1.2: Existing datasets in robotics and agriculture. Adapted from Kragh et al. (2017).

Dataset	Environment	Length	Localization	Sensors	Obstacles	Annotations
KITTI (Geiger et al., 2013)	urban	6 h	✓	stereo camera, lidar	cars, trucks, trams, pedestrians, cyclists	2D + 3D bounding boxes
Oxford (Maddern et al., 2017)	urban	1000 km	✓	stereo camera, lidars, color cameras	cars, trucks, pedestrians, cyclists	none
Marulan (Peynot et al., 2010)	rural	2 h	✓	lasers, radar, color camera, infra-red camera	humans, box, poles, bricks, vegetation	none
NREC (Pezzementi et al., 2017)	orchards	8 h	✓	stereo camera	humans, vegetation	bounding boxes (only humans)

In this part of the study, two research perception platforms and their associated multi-modal datasets are presented. The first chapter presents the SuperSensorKit platform which was developed as part of this thesis work and used for data acquisition in various agricultural fields in Denmark. The second chapter presents the robotic platform Shrimp which was developed at the Australian Centre for Field Robotics in Sydney, Australia and used for data acquisition in orchards and fields during 2013.

The datasets acquired with the two platforms all facilitate development and evaluation of multi-modal obstacle detection algorithms in a broad range of realistic agricultural environments. Common obstacles in agriculture such as humans, animals, vegetation, vehicles, and buildings are thus widely represented throughout the datasets.

Both platforms include multiple sensing technologies (modalities) that each have their own strengths and weaknesses for use in an obstacle detection system. A sensor can either be exteroceptive or proprioceptive. Exteroceptive sensors perceive and measure the environment (e.g. color, temperature, distance, and material), whereas proprioceptive sensors measure characteristics internal to the platform or robot (e.g. motion, heading, and vibrations). A comprehensive discussion and evaluation of advantages and disadvantages of the different exteroceptive sensing technologies is available in Paper 1 (Christiansen et al., 2015) and Paper 2 (Christiansen et al., 2017). Here, lidar, radar, and imaging sensors (color, thermal, and stereo) are compared with respect to range, resolution, cost, and robustness towards changes in illumination and weather conditions.

2 SuperSensorKit

The content of this chapter partly appears in the following three publications:

> Paper 1: *Christiansen et al. (2015). Advanced sensor platform for human detection and protection in autonomous farming. Conference presentation at the 10th European Conference on Precision Agriculture (ECPA).*
>
> Paper 2: *Christiansen et al. (2017). Platform for evaluating sensors and human detection in autonomous mowing operations. Precision Agriculture, June 2017.*
>
> Paper 3: *Kragh et al. (2017). FieldSAFE: Dataset for Obstacle Detection in Agriculture. MDPI Sensors, Special Issue: Sensors in Agriculture, November 2017.*

2.1 Sensors

Figure 2.1 shows the SuperSensorKit recording platform mounted on a grass mowing implement behind a tractor. The platform was designed for easy and flexible mounting on both a calibration rig and various agricultural vehicles such as tractors, harvesters, and all-terrain vehicles (ATVs). It was therefore also equipped with rubber suspensions to minimize vibration noise in images, and a tiltable camera frame for adjusting the camera pitch angle according to the height of the mount point.

Table 2.1 lists the exteroceptive sensors of the platform, whereas Table 2.2 lists the proprioceptive sensors. All sensors were interfaced in the Robot Operating System (ROS) (Quigley et al., 2009) with a Conpleks Robotech Controller 701. The lidar sensor is a Velodyne HDL-32E, rotating at 10 Hz with 32 vertically oriented laser beams operating at a wavelength of 905 nm. It generates 3D point clouds with up to 70,000 points per frame at 10 fps. The two tables describe the final recording setup used for acquiring the last (and most comprehensive) dataset. However, the platform has been improved incrementally throughout the thesis work, with multiple versions of stereo cameras, thermal cameras, and GPS systems. Below, the previous sensor versions and their strengths and weaknesses are described in detail.

Figure 2.1: Recording platform. Reprinted from Kragh et al. (2017).

Table 2.1: Exteroceptive sensors. Reprinted from Kragh et al. (2017).

Sensor	Model	Resolution	field of view (FOV)	Range	Rate
Stereo camera	Multisense S21 CMV2000	1024 × 544	85° × 50°	1.5–50 m	10 fps
Webcam	Logitech HD Pro C920	1920 × 1080	70° × 43°	-	20 fps
360° camera	Giroptic 360cam	2048 × 833	360° × 292°	-	30 fps
Thermal camera	FLIR A65, 13 mm	640 × 512	45° × 37°	-	30 fps
Lidar	Velodyne HDL-32E	2172 × 32	360° × 40°	1–100 m	10 fps
Radar	Delphi ESR	16 targets/frame	90° × 4.2°	0–60 m	20 fps
		16 targets/frame	20° × 4.2°	0–174 m	20 fps

Table 2.2: Proprioceptive sensors. Reprinted from Kragh et al. (2017).

Sensor	Model	Description	Rate
GPS	Trimble BD982 GNSS	Dual antenna RTK GNSS system. Measures position and horizontal heading of the platform.	20 Hz
IMU	Vectornav VN-100	Measures acceleration, angular velocity, magnetic field, and barometric pressure.	50 Hz

Stereo Camera

According to plan, a stereo camera developed specifically for agricultural use by one of the SAFE project partners was supposed to be mounted on the sensor platform. However, due to problems accessing raw data from the sensor with reasonable frame rates, the sensor was never used for recordings. Instead, three different versions of existing technologies were applied and tested on the platform as shown in Figure 2.2:

S1. Two NSC1003 logarithmic and global shutter CMOS sensors from New Imaging Technology providing 1280 × 1024 pixels at 25 fps. The camera baseline was 5 cm.
S2. Two Flea3/FL3-GE-28S4C-C global shutter cameras from Point Grey providing 1928 × 1448 pixels at 15 fps. The camera baseline was 24 cm.
S3. MultiSense S21 global shutter camera from Carnegie Robotics providing 1024 × 544 pixels at 10 fps. The camera baseline was 21 cm.

(a) T1 (b) T2 (c) T3

Figure 2.2: Stereo camera versions tested on the SuperSensorKit.

Initially, S1 was chosen as an ideal candidate for use in changing illumination conditions, due to its high dynamic range. However, the logarithmic intensity scale resulted in small typical bit depths and made stereo matching difficult. Additionally, the small baseline gave imprecise range estimates at far distances.

S2 was chosen as a research solution for post-processing only such that stereo matching was done offline after data collection. A small hardware circuit was developed to synchronize the triggering of the cameras, and the setup was manually calibrated. S2 was a clear improvement to S1, but suffered from frame loss and occasional errors that required manual resets.

With additional funding, S3 was chosen as an off-the-shelf solution that performs stereo matching in high resolution online using an FPGA. The camera was designed for robotic use in long-range outdoor applications and therefore integrated well with the ROS software platform. It further included an internal IMU.

Thermal Camera

Figure 2.3 shows the three different versions of thermal cameras that were applied and tested on the platform.

T1. FLIR A320 thermal camera from FLIR systems providing absolute temperature images of 380 × 240 pixels at 7 fps.
T2. HawkVision analog thermal camera from Tonbo Imaging Inc providing relative temperature images of 640 × 480 pixels at 25 fps. An iPORT Analog-Pro frame grabber from Pleora was used to convert from analog to digital signals.
T3. FLIR A65 13 mm lens thermal camera from FLIR systems providing absolute temperature images of 640 × 512 at 30 fps.

T1 was used in the first field trial for initial sensor exploration. However, the interfacing software had to run in Windows and could not easily be integrated with ROS. Therefore, the camera was not synchronized with the other sensors and could thus not be used effectively for sensor fusion. Even further, the image resolution and frame rate were rather low.

(a) T1 (b) T2 (c) T3

Figure 2.3: Thermal camera versions tested on the SuperSensorKit.

The second thermal camera T2 provided a much higher frame rate and a considerably better image resolution. However, it could not provide absolute temperatures and included a normalization step for each image. This made it difficult to utilize the temperature information, as intensities for the same object depended on general weather and illumination conditions at the specific time.

With additional funding, T3 was chosen as an ideal solution for the problem. The camera had a high frame rate, high resolution, and provided absolute temperatures. It could further be triggered externally, allowing exact hardware synchronization with the stereo camera.

GPS

Two different versions of Real Time Kinematic (RTK) GPS systems were applied and tested on the platform:

GPS1. AG GPS361 RTK GNSS system from Trimble.
GPS2. BD982 RTK GNSS system from Trimble. With a dual-antenna system, the GPS provides absolute heading angles with a standard deviation of <0.5°.

The first GPS system GPS1 was provided by one of the SAFE project partners and is a standard RTK GPS system for agricultural vehicles. It was used for most field trials with GPS carrier measurements providing heading estimates of the platform (Bevly and Cobb, 2010). However, the heading estimates were imprecise and noisy at slow velocities, which made localization difficult during headland turns and reverse driving.

GPS2 was borrowed by another research institution for the last field trial, as its dual-antenna RTK GPS system could provide accurate heading estimates even at low velocities and during reverse driving. The system made both latitude and longitude positioning and yaw angle estimates of the platform highly accurate, whereas altitude positioning as well as pitch and roll estimates were less precise.

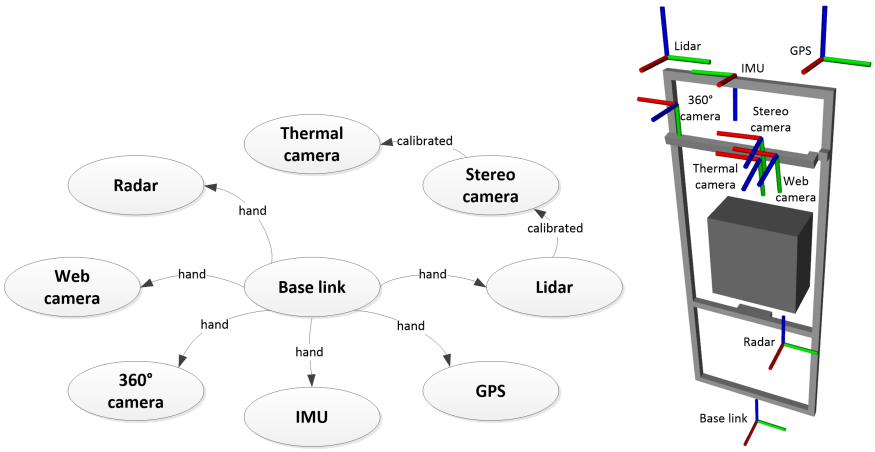


Figure 2.4: Sensor frames on the SuperSensorKit. (a) Registration chain. "Hand" denotes a manual measurement by hand, whereas "calibrated" indicates that an automated calibration procedure was used to estimate the extrinsic parameters. (b) Sensor frames overlaid on platform model. Adapted from Kragh et al. (2017).

2.2 Manual Calibration, Registration and Synchronization

Figure 2.4a illustrates the chain of transformations in relation to the common reference frame on the platform called "base link", whereas Figure 2.4b shows the physical placement of the frames on the sensor platform. Extrinsic parameters (position and orientation) of all sensors on the SuperSensorKit were first measured by hand. Calibration and registration procedures were then used to refine the estimates for the cameras and the lidar. The stereo camera and thermal camera were calibrated and registered using a calibration checkerboard and the Camera Calibration toolbox from MATLAB. A custom-made visual-thermal checkerboard was designed and constructed for this purpose to enable both the stereo camera and the thermal camera to distinguish "black" tiles from "white" tiles. In this way, both cameras could be calibrated and registered using the same procedure as shown in Figure 2.5. For more information on the specific procedure, see Paper 2 (Christiansen et al., 2017).

As the depth image from the stereo camera could be converted to a 3D point cloud, the lidar and the stereo camera were registered using the Iterative Closest Point (ICP) algorithm on recordings of multiple static scenes with various objects and structures. An average over all the static scenes was used as a final estimate for the transformation between the two sensors.

All sensors were interfaced and synchronized using a best-effort approach in ROS. That is, all sensor messages were timestamped at their arrival by the ROS system time. The

(a) Stereo camera (b) Thermal camera

Figure 2.5: Calibration of stereo camera and thermal camera with custom-made visual-thermal checkerboard.

best-effort message delivery, however, did not guarantee specific delivery times, and the actual time delays therefore depended on internal sensor processing times, transmission times, network traffic load, software drivers, and the kernel scheduler in the operating system (Lütkebohle, I., 2017). For more information on synchronization and typical sensor latencies, see Paper 3 (Kragh et al., 2017).

The lidar, stereo camera, and thermal camera were further synchronized in hardware using a pulse per second (PPS) signal from the GPS system of the lidar. A microcontroller was used to derive synchronized 10 fps and 30 fps trigger signals from the PPS signal such that the two cameras were triggered at exactly the same time. Effectively, this resulted in synchronized data from the lidar, stereo camera, and thermal camera with 10 fps.

2.3 Automated Registration and Synchronization

As described above, all extrinsic parameters were measured by hand and refined, if possible, with different semi-automated calibration procedures. A few important parameters for enabling accurate localization and mapping, however, were not easily measured by hand with sufficient precision. These concern extrinsic parameters (position and orientation) for the IMU and GPS sensors as well as potential time delays between sensors. In order to accurately georeference 3D points from a lidar point cloud, an exact transformation must be defined between the coordinate frames of the lidar, IMU, and GPS. Positional errors during georeferencing can originate from any of the above parameters. However, whereas positional extrinsic errors map directly to georeferenced errors, angular extrinsic errors increase with distance (range measurements). An angular error of 1° thus introduces a positional error of $100\,\text{m} \cdot \sin(1°) = 1.75\,\text{m}$ at a distance of 100 m from the lidar sensor. It is therefore critical to estimate angular transforms with up to three or four decimal places when specified in radians.

In robotics, a common practice is to extract transforms from a CAD model of a robot. This will often provide more accurate estimates than transforms measured by hand. IMUs can be calibrated in isolation by utilizing the known gravitational magnitude and direction while the vehicle is stationary (Nebot and Durrant-Whyte, 1999). Odometry from wheel encoders can further easily be calibrated by adjusting the parameters such that the estimated path fits with GPS measurements of the same traversal (Bevly et al., 2002). In the literature, more advanced calibration methods have been proposed for semi or fully automated estimation of extrinsic sensor parameters. A number of methods exist for automatically estimating transforms between a lidar and a camera (Geiger et al., 2012; Levinson and Thrun, 2013; Pandey et al., 2015; Taylor and Nieto, 2016). The method by Levinson and Thrun (2013) is even online-applicable, making it capable of adapting to small changes over time. Underwood et al. (2007) presents a framework for calculating static transforms between a range sensor and a navigation system utilizing sensed data of a known and manually marked structure. In a follow-up study, the approach is extended to simultaneously optimize extrinsic parameters between a navigation system and multiple range sensors (Underwood et al., 2010). Levinson and Thrun (2014) propose a fully automated and unsupervised method for calibrating intrinsic parameters of a multi-beam lidar as well as an extrinsic transform to the robot's navigation frame. The method seeks to minimize an error defined as the distance between points from one laser beam to a surface defined by points from the other beams. Point-wise normal vectors are thus computed for N pairs of accumulated lidar frames with N being the number of lidar beams. The cost function is related to the energy function of the ICP scan matching method (Chen and Medioni, 1992). However, whereas ICP assumes rigid individual point clouds acquired at an instance in time, a point cloud from a rotating lidar is acquired over a time interval with multiple robot poses along the way if the vehicle is in motion. The optimization objective therefore is to align individual points instead of rigid point clouds.

In this section, an unsupervised calibration procedure is proposed to automatically tune both extrinsic parameters and time delays for a lidar, IMU, and GPS sensor during platform movement. The method optimizes the extrinsic parameters and time delays iteratively by computing and minimizing point cloud alignment errors across all frames. As opposed to the method of Levinson and Thrun (2014), we propose a simple and computationally cheaper cost function that favors a combined point cloud that occupies as few voxels as possible. The method extends easily to more parameters such as estimating covariance matrices in Kalman filtering for navigation and state estimation. It further allows for registration of more range sensors such as radar and stereo camera, simply by incorporating all measurements into the same data representation.

2.3.1 Optimization Parameters

Figure 2.6 illustrates a simplified version of the SuperSensorKit with only lidar, GPS, and IMU sensors. The transformation tree as defined in Figure 2.4a is further reduced to

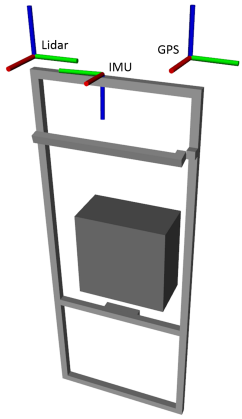


Figure 2.6: Simplified setup with only lidar, IMU, and GPS sensor frames.

include only the three sensors with the GPS as the common frame of reference. This simplification reduces the number of parameters and thus simplifies the optimization problem considerably. Table 2.3 lists the parameters chosen for optimization, as these parameters all influence localization and global mapping of lidar point clouds. The position of the IMU is left out, as its estimates of angular velocity and acceleration are not used in the simplified model. Time delays are further assumed constant, although higher order approximations could include constant and varying drifts over time.

Table 2.3: Optimization parameters including transformations and time delays. DOF denotes the degrees of freedom.

Parameter	Unit	DOF
$\text{Lidar}_{\text{position}}$	m	3
$\text{Lidar}_{\text{orientation}}$	rad	3
$\text{IMU}_{\text{orientation}}$	rad	3
$\Delta t_{\text{GPS}-\text{lidar}}$	s	1
$\Delta t_{\text{GPS}-\text{IMU}}$	s	1

Equation 2.1 presents the transformation of a 3D point $\vec{P}_{\text{lidar}} = [x, y, z, 1]^T$ in the lidar frame to a 3D point $\vec{P}_{\text{UTM}} = [x, y, z, w]^T$ in a global Universal Transverse Mercator (UTM) frame, both specified as homogeneous coordinates. Each transformation matrix has a subscript S or D, indicating whether the transform is static or dynamic (defined by IMU or GPS measurements).

$$\vec{P}_{\text{UTM}} = \mathbf{T}_{\text{GPS},D}\,\mathbf{R}_{\text{GPS},D}\,\mathbf{R}^{-1}_{\text{IMU},S}\,\mathbf{R}_{\text{IMU},D}\,\mathbf{R}_{\text{IMU},S}\,\mathbf{R}_{\text{lidar},S}\,\mathbf{T}_{\text{lidar},S}\,\vec{P}_{\text{lidar}} \tag{2.1}$$

First, a static translation $\mathbf{T}_{\text{lidar},S}$ and a static rotation $\mathbf{R}_{\text{lidar},S}$ are applied to transform the point from the lidar frame to the GPS frame. These are directly defined from the x, y, and z translations contained in the parameter vector $\text{Lidar}_{\text{position}}$ and the roll,

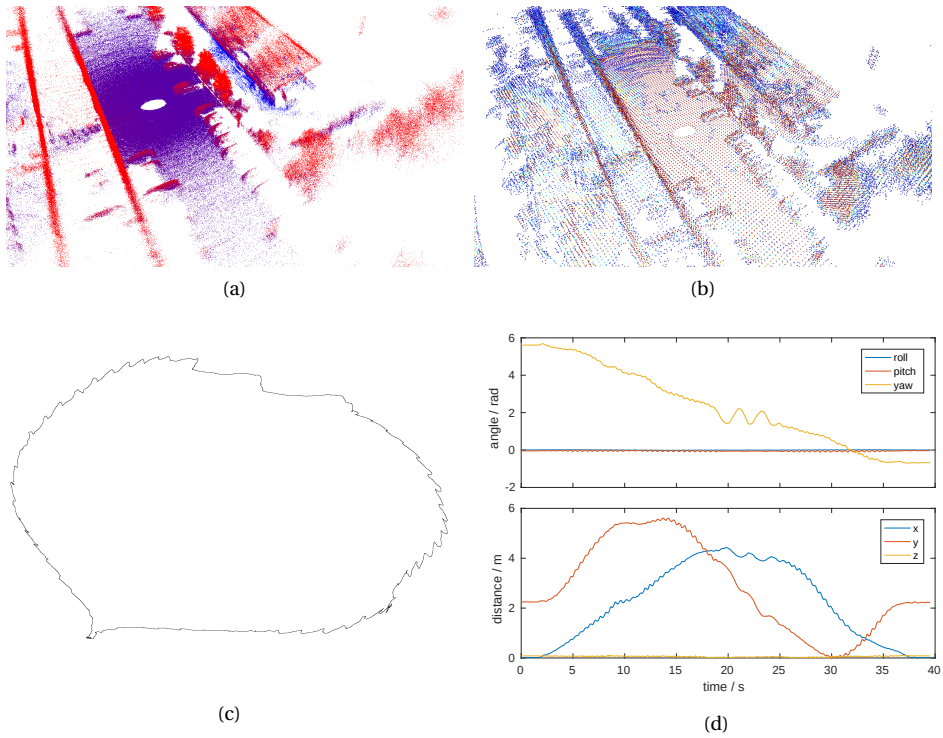


Figure 2.7: Georeferenced point clouds from subsequent lidar frames. (a) Accumulated points colored by height. (b) Voxelized point cloud colored by number of points inside each 0.5 m × 0.5 m × 0.5 m voxel. (c) Traversed circular path from above (d) Distance and angle coordinates as a function of time.

pitch, and yaw rotations contained in the parameter vector $\text{Lidar}_{\text{orientation}}$. The term $\mathbf{R}_{\text{IMU},S}^{-1}\mathbf{R}_{\text{IMU},D}\mathbf{R}_{\text{IMU},S}$ applies first a static rotation to the IMU frame based on the parameter vector $\text{IMU}_{\text{orientation}}$, then uses the current pitch and roll measurements from the IMU, and finally undoes the first static rotation. In practice, the combination transforms the point to a coordinate frame with the z-axis parallel to the direction of the gravitational force (i.e. xy-plane parallel to the ground plane). $\mathbf{R}_{\text{GPS},D}$ uses the current yaw measurement (heading with respect to true north) from the GPS to align the coordinate frame with the UTM frame. And $\mathbf{T}_{\text{GPS},D}$ finally applies the measured GPS position through a translation, thus providing 3D points in a global UTM frame. The two time delays, $\Delta t_{\text{GPS}-\text{lidar}}$ and $\Delta t_{\text{GPS}-\text{IMU}}$ are included in $\mathbf{T}_{\text{GPS},D}$, $\mathbf{R}_{\text{GPS},D}$, and $\mathbf{R}_{\text{IMU},D}$ such that the GPS and IMU measurements are delayed accordingly.

Figure 2.7 shows an example of a circular traversal in a parking lot, in which Equation 2.1 was used to georeference 3D points from subsequent lidar frames. The objective of the optimization process is to align subsequent point clouds optimally. The proposed approach does this by maximizing the point density in occupied areas. Specifically,

the accumulated point cloud is voxelized as shown in Figure 2.7b. Here, the points are pseudocolored by the number of points $N_{i,j,k}$ inside each voxel with indices i, j, and k.

2.3.2 Objective Functions

Based on the voxelized point cloud, four different cost functions are proposed and evaluated. The first and most simple cost counts the number of occupied (non-empty) voxels:

$$\text{cost}_1 = \sum_{i,j,k} \text{sign}\left(N_{i,j,k}\right) \tag{2.2}$$

By minimizing the number of occupied voxels, the point density is maximized in occupied areas. Another closely related cost computes the mean density in occupied voxels, where a minus is introduced to effectively maximize the density.

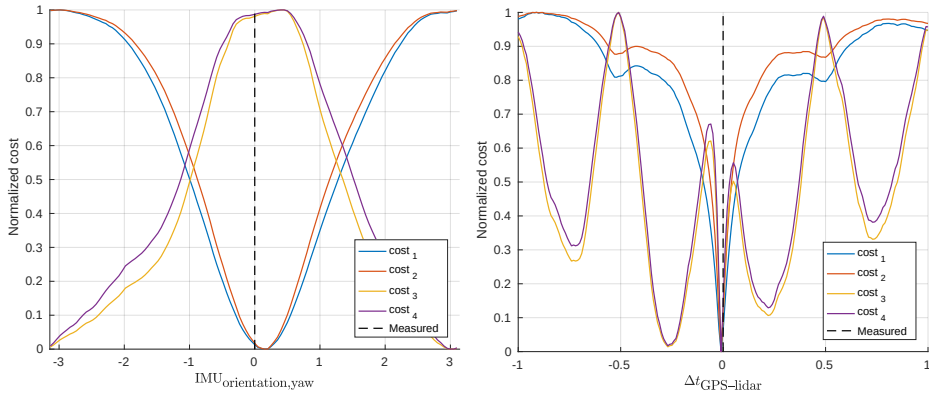
$$\text{cost}_2 = -\frac{1}{\text{cost}_1} \sum_{i,j,k} N_{i,j,k} \tag{2.3}$$

A downside of cost_1 and cost_2 is that they will potentially both favor a stationary case in which the navigation introduces no movement. That is, a stationary lidar will at most occupy all voxels within its field of view and range limits. With the parameters in Table 2.3, this scenario is not possible as long as the time delays are limited to a fixed range. However, with a more complex model with more degrees of freedom including e.g. covariance matrices in Kalman filtering, the problem can theoretically arise. Therefore, two additional cost functions are proposed that maximize the point density while also maximizing the smallest volume containing all points. In this way, the cost functions favor dense points distributed over a large area/volume. A volume V is defined as the volume of the 3D convex hull of the occupied voxel centroids. cost_1 and cost_2 are then extended:

$$\text{cost}_3 = \frac{\text{cost}_1}{V} \tag{2.4}$$

$$\text{cost}_4 = \text{cost}_2 V \tag{2.5}$$

The four cost functions have been evaluated and compared on individual parameters from Table 2.3 using a cubic voxel resolution of 0.1 m. Figure 2.8a illustrates the cost for each function normalized to the range $[0, 1]$ when varying the yaw-angle in $\text{IMU}_{\text{orientation}}$ while fixing all other parameters to the initially measured values. Similarly, Figure 2.8b shows the same costs when varying the time delay $\Delta t_{\text{GPS}-\text{lidar}}$. For both examples, cost_1 and cost_2 had global minima near the measured values, whereas cost_3 and cost_4 only had local minima and experienced local maxima near the measured values. The reason for this may be that a small change in e.g. the GPS time delay near the correct solution affected the convex hull volume more than it affected the number of occupied voxels. That is, cost_3 and cost_4 favored a small error in the time delay since it increased the



(a) (b)

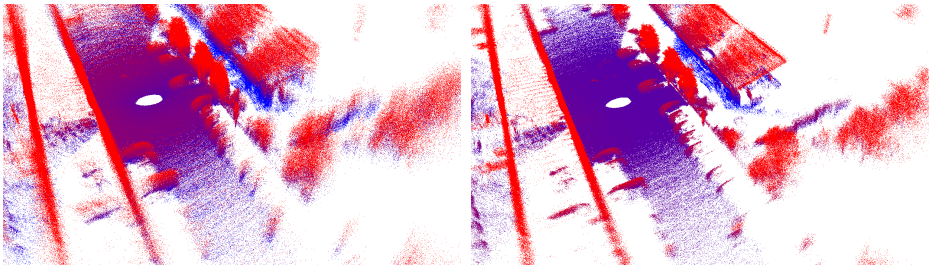
Figure 2.8: Comparison of normalized costs while varying two optimization parameters individually.

volume more than it decreased the number of occupied voxels. As expected, cost_1 and cost_2 showed similar tendencies with cost_1 experiencing a slightly larger gradient. In the following optimization procedure, cost_1 is therefore used.

2.3.3 Optimization

Although the objective function experienced multiple local minima as exemplified in Figure 2.8, a local search technique for numerical optimization is used to estimate the optimal parameters. The Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm is a common quasi-Newton optimization algorithm that approximates the Hessian based on its previous gradient evaluations/estimations (Liu and Nocedal, 1989). This greatly limits the number of function evaluations and thus reduces computation time. On an average laptop, each iteration takes 67 sec for an 40 sec traversal consisting of 401 frames from the lidar. A variant of the algorithm is the limited-memory BFGS with box constraints (L-BFGS-B) (Byrd et al., 1995), which allows upper and lower bounds for each parameter. This is useful for the parameters in Table 2.3, since they express physical quantities such as distances and angles that can be bounded to ranges near their hand-measured values.

Table 2.4 shows the measured parameter values along with optimized results after applying the L-BFGS-B algorithm. With measured parameter values, a cost of 3,337,065 was calculated. This means that 3,337,065 voxels were occupied after accumulating point clouds from the entire traversal. With zero-initialization, all parameters were initialized to zero before running the optimization. This resulted in a cost of 2,923,501 after optimization, slightly below the calculated cost for the hand-measured parameter values. With fine-tuning, all parameters were initialized to their measured values before running the optimization. This resulted in an optimized cost of 1,823,541, which is nearly half



(a) (b)

Figure 2.9: Georeferenced point clouds before and after optimization. (a) Hand-measured parameters. (b) Optimized parameters.

the initial cost. Figure 2.9 illustrates the combined georeferenced cloud before and after parameter optimization. Clearly, the points are much better aligned after optimization.

Table 2.4: Optimization parameters including transformations and time delays.

Parameter	Measured	Bounds	Optimized	
			Zero-initialized	Fine-tuned
$\text{Lidar}_{\text{position}}$	$[0.140, -0.635, -0.07]$	$[[-1,1],[-1,1],[-1,1]]$	$[0.279, -0.568, -0.067]$	$[0.106, -0.635, -0.038]$
$\text{Lidar}_{\text{orientation}}$	$[0.00, 0.00, 0.06]$	$[[-\pi,\pi],[-\pi,\pi],[-\pi,\pi]]$	$[-0.008, -0.163, -0.033]$	$[-0.002, 0.072, 0.053]$
$\text{IMU}_{\text{orientation}}$	$[3.142, 0.00, 0.00]$	$[[-\pi,\pi],[-\pi,\pi],[-\pi,\pi]]$	$[-0.305, 0.223, 0.110]$	$[3.142, -0.097, 0.096]$
$\Delta t_{\text{GPS-lidar}}$	0.0	$[-0.5, 0.5]$	-0.04	-0.01
$\Delta t_{\text{GPS-IMU}}$	0.0	$[-0.5, 0.5]$	-0.01	-0.01
cost_1	3337065		2923501	1823541

2.4 Data Collection

The SuperSensorKit recording platform described in section 2.1 was used to record 7 agricultural obstacle detection datasets at various locations in Denmark. Figure 2.10 shows the locations, shapes, and appearances of the fields that were traversed during acquisition. Table 2.5 further provides details for each dataset such as the season of recording, area of the field, and included obstacle types.

In the following subsections, each of the 7 datasets are further described and related to the scientific papers that use them. Furthermore, the progressive development and improvement of the sensor platform over time is shown and discussed. The main contribution of the section is the FieldSAFE dataset (DK6) that has been made publicly available and is described in Paper 3 (Kragh et al., 2017).

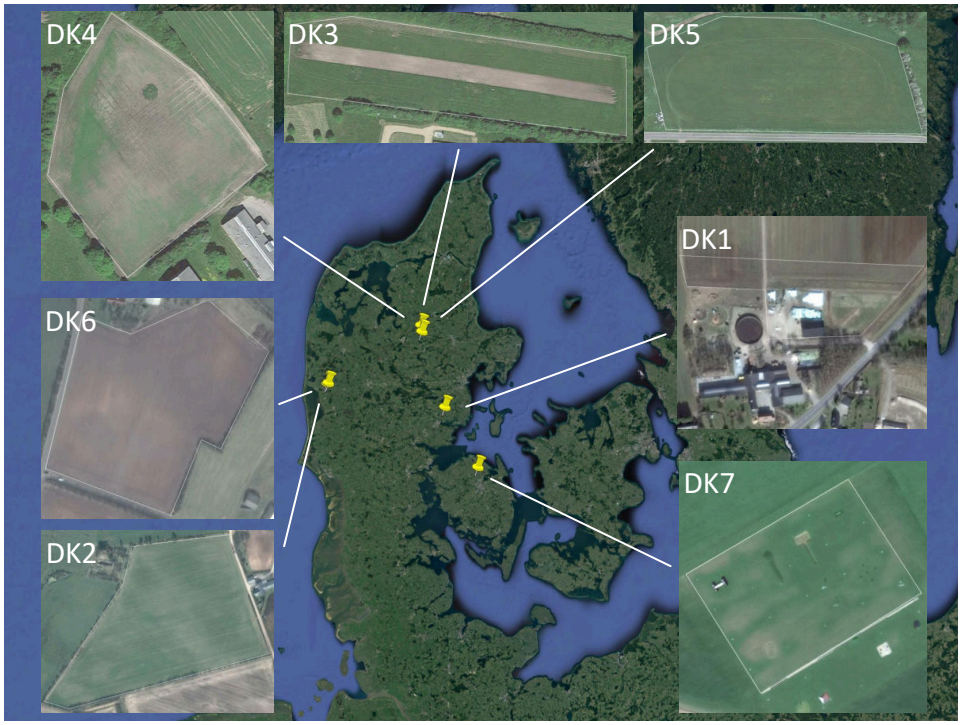


Figure 2.10: Overview of datasets collected with the SuperSensorKit in Denmark.

Table 2.5: Details for datasets recorded in Denmark. All recordings include vegetation (trees and bushes) as obstacles. (✓) indicates that mannequin dolls or toy animals were used as obstacles.

ID	Dataset	Setting	Season	Area (ha)	Length (hours)	Labeled frames	Obstacles			
							Human	Animal	Vehicle	Building
DK1	**Children's farm** *November 2014*	Field	Autumn	1.7	1.6	15	✓	✓		
DK2	**Lem** *June 2015*	Field	Summer	7.8	1.8	0	✓			✓
DK3	**Foulum grass** *June 2015*	Field	Summer	1.2	0.7	All (GPS)	✓		✓	✓
DK4	**Foulum row crop** *September 2015*	Row crop	Autumn	0.8	0.5	0	(✓)	(✓)		
DK5	**Tjele** *June 2016*	Field	Summer	3.5	2.0	0	✓			
DK6	**FieldSAFE** *October 2016*	Field	Autumn	3.3	2.2	All (GPS)	✓		✓	✓
DK7	**HCA Airport** *September 2017*	Field	Autumn	1.7	0.4	All (GPS)	✓		✓	✓

(a) (b)

Figure 2.11: DK1: Children's farm. (a) Platform mounted on ATV. (b) Children playing in high grass.

DK1: Children's farm

The purpose of the first field trial was to test the sensor platform outside the lab, in a realistic agricultural environment. Adults and children were used as obstacles along with two dogs, a rabbit, and a hen. The early version of the SuperSensorKit included lidar, radar, webcam, stereo camera (S1), thermal camera (T1), IMU, and single-antenna RTK GPS. The platform was mounted on an ATV to ease transportation and ensure flexibility during recording. Figure 2.11 shows the early version of the platform along with an example of children playing and acting as moving obstacles. The dataset was used in Paper 1 (Christiansen et al., 2015) and Paper 2 (Christiansen et al., 2017) for initial sensor exploration and for comparing advantages and disadvantages of the different sensing technologies.

DK2: Lem

The purpose of the second field trial was to record an entire traversal of a field during normal grass mowing with realistic operation speeds (up to 18 $\mathrm{km\,h^{-1}}$). The SuperSensorKit was updated with a new thermal camera (T2) and a new stereo camera (S2), while the lidar was moved to the top of the sensor frame to provide an unobstructed 360° field of view. The platform was further improved mechanically with rubber suspensions minimizing vibration noise in images and with a standardized A-frame mount for flexible installation on tractors and implements.

The platform was mounted on a mowing implement to a tractor as shown in Figure 2.12a. A number of static obstacles were placed in the field along the path of the tractor as shown in Figure 2.12b. Immediately before driving into the obstacles, the tractor was stopped to prevent collision. For safety reasons, mannequin dolls (adult and children) were used instead of humans. Furthermore, green barrels were used to represent small

(a) (b)

(c)

Figure 2.12: DK1: Lem. (a) Platform mounted on implement. (b) Obstacle locations in the field. (c) Example obstacles recorded with the webcam.

and short obstacles that were hard to see, even for a human operator. These barrels have since then been included in an ISO-standard for obstacle detection in agriculture (ISO/FDIS 18497, 2017). At the end of the recording, the mower was turned off, and actual humans were recorded in various postures. Figure 2.12c shows examples of both static obstacles and humans in various postures partly occluded by high grass.

The dataset was used in Paper 2 (Christiansen et al., 2017) for evaluating sensor calibration and registration and for comparing advantages and disadvantages of the different sensing technologies.

DK3: Foulum grass

The purpose of the third field trial was to record static obstacles in a grass field, while acquiring a high-resolution orthophoto of the field. A DJI Phantom 2 drone was used to record bird's-eye view images of the field that were stitched to an orthophoto using Pix4D (Pix4D, 2014). The orthophoto was used to obtain ground truth GPS positions and class labels for all obstacles by subsequent manual pixel-wise labeling of the image. The sensor platform was mounted directly on the front of a tractor without implements. Obstacles included vegetation, a walking human, mannequin dolls, barrels, a vehicle, and in-field water wells.

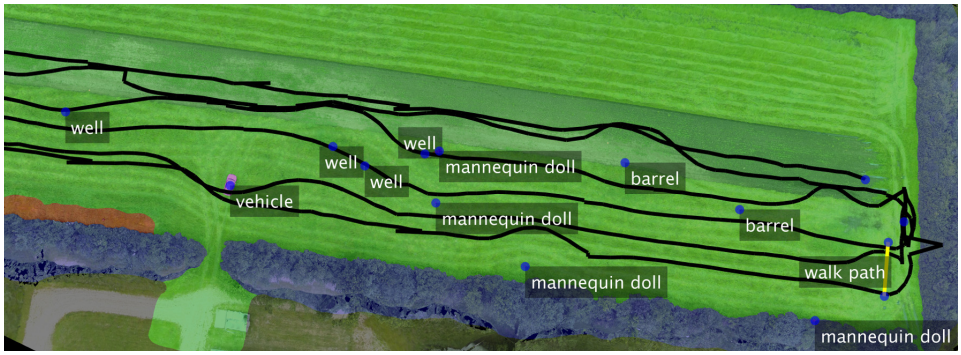


Figure 2.13: DK3: Foulum grass. Ground truth labels overlaid on orthophoto. The trajectory of the tractor is shown with a black line, whereas the walking path of a human is shown with a yellow line. The ground truth labels illustrate vegetation (blue), ground (green), and non-traversable ground (red). Adapted from Kragh et al. (2016b).

Prior to traversing the field, GPS markers were placed along the edge of the field, and their positions were measured with a handheld RTK GPS device. By manually pointing out the pixel locations of the markers, a nonreflective similarity transformation was estimated to convert from GPS coordinates (in UTM format) to pixel coordinates. Figure 2.13 illustrates the pixel-wise ground truth annotations overlaid on the orthophoto of the field.

The dataset was used in Paper 6 (Kragh et al., 2016b) for multi-modal obstacle detection and occupancy grid mapping.

DK4: Foulum row crop

The purpose of the fourth field trial was to record static obstacles in a maize row crop under different illumination conditions. The sensor platform was mounted on the front of a tractor, and mannequin dolls, barrels, and toy animals were used as obstacles. Different illumination conditions were captured by traversing the field in two directions, three times during the day.

The dataset was used by Steen et al. (2016) and Christiansen et al. (2016b), but has not been used in any of the papers included in this thesis.

DK5: Tjele

Every year, numerous young animals are killed by agricultural machines during the first annual harvests. Roe deer fawns, hares, pheasants, and partridges either attempt to escape through or hide within tall vegetation such as grass, when farming vehicles approach them. The purpose of the fifth field trial therefore was to record roe deer fawns hiding in a grass field while mowing it.

The sensor platform was mounted on a mowing implement to a tractor. An area of 3.5 ha was covered, mowing from the outside and inwards, thus maximizing the likelihood of seeing the animals. However, unfortunately no animals were detected during the field trial. Therefore, the dataset has not been used in any scientific publications.

DK6: FieldSAFE

The purpose of the sixth field trial was to record both static and dynamic (moving) obstacles in a grass field while mowing. The field trial was an extension of DK3 with an improved sensor setup and a more advanced method for acquiring ground truth data of both static and moving obstacles. The dataset, FieldSAFE, has been made publicly available at https://vision.eng.au.dk/fieldsafe/ and is described in Paper 3 (Kragh et al., 2017).

The dataset was used in Paper 7 (Korthals et al., 2018) for multi-modal detection and mapping of static and dynamic obstacles, and in Paper 9 (Kragh et al., 2018) for multi-modal semantic segmentation in 3D.

Data

After recording DK5, the SuperSensorKit was updated with a new thermal camera (T3), a new stereo camera (S3), a 360° camera, and a new dual-antenna RTK GPS. The sensor platform therefore resembled the exact setup described in section 2.1 above. Figure 2.14 illustrates an example from the dataset of synchronized frames from all modalities.

The recordings were split into two sessions: one for recording static obstacles only, and one for including moving obstacles as well. For the first session, static obstacles were placed along the edge of the field prior to recording. Static obstacles included vegetation, mannequin dolls, barrels, vehicles, buildings, and rocks as exemplified in Figure 2.15. The field was then traversed while mowing grass in a regular pattern by first cutting the headland. For the second session, four humans walked in random patterns in a subset of the field, crossing the path of the tractor multiple times at both close and long range. Various postures were represented such as standing, sitting, and lying in the grass. Figure 2.16 illustrates stereo camera images of the humans along with their traversed paths on the field.

Ground Truth

As for DK3, a number of GPS markers were placed along the edge of the field and measured with exact GPS coordinates using a handheld RTK GPS device. A static, high-resolution orthophoto of the field was acquired with a DJI Phantom 4 drone and stitched using Pix4D (Pix4D, 2014). Figure 2.17 shows the orthophoto along with a manually labeled version, assigning each pixel to either *grass*, *ground*, *road*, *vegetation*, *building*, *GPS marker*, *barrel*, *human*, or *other*. Using corresponding pairs of pixel and GPS coordinates for all GPS markers, a nonreflective similarity transform was estimated to convert from GPS coordinates (in UTM format) to pixel coordinates.

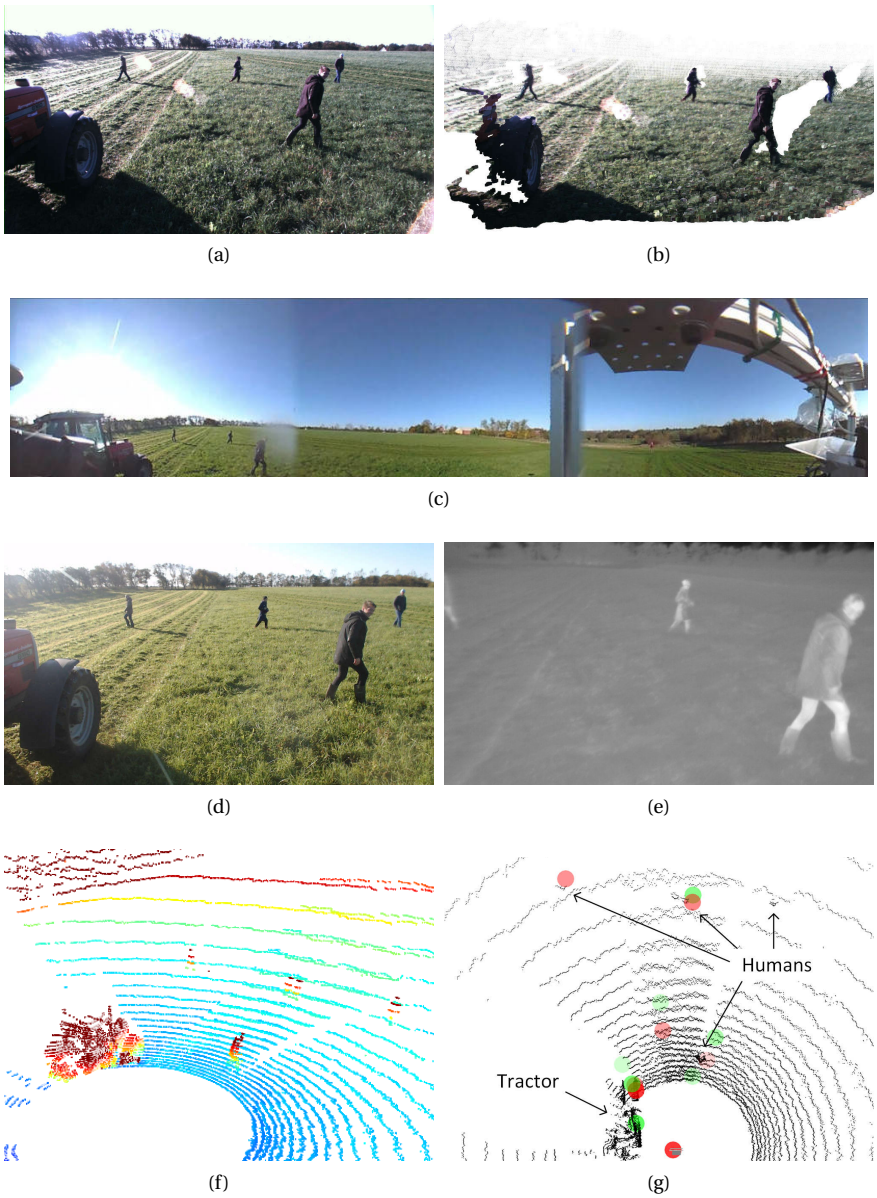
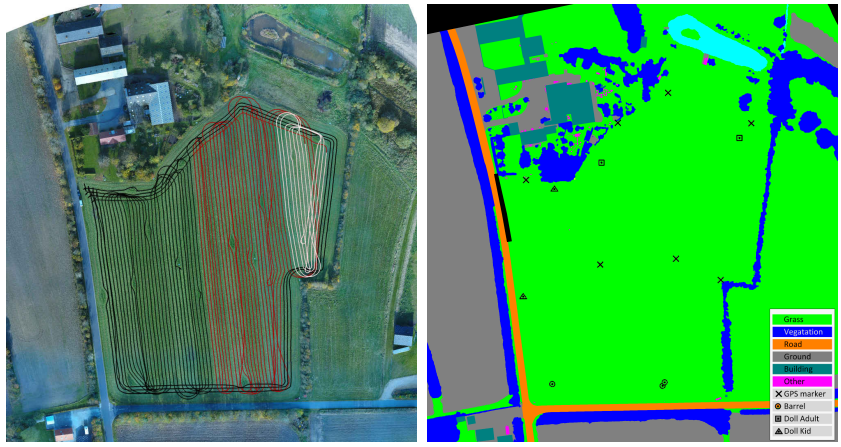


Figure 2.14: Example frames from the FieldSAFE dataset. (a) Left stereo image. (b) Stereo point cloud. (c) 360° camera image (cropped). (d) Webcam image. (e) Thermal camera image (cropped). (f) Lidar point cloud (cropped and colored by height). (g) Radar detections overlaid on lidar point cloud (black). Green and red circles denote detections from mid- and long-range modes, respectively. Reprinted from Kragh et al. (2017).

Figure 2.15: Examples of static obstacles. Reprinted from Kragh et al. (2017).

(a) Human 1 (b) Human 2 (c) Human 3 (d) Human 4

Figure 2.16: Examples of moving obstacles (from the stereo camera) and their paths (black) overlaid on the tractor path (gray). Reprinted from Kragh et al. (2017).



(a) (b)

Figure 2.17: Colored and labeled orthophotos. (a) Orthophoto with tractor tracks overlaid. Black tracks include only static obstacles, whereas red and white tracks also have moving obstacles. Currently, red tracks have no ground truth annotations for moving obstacles. (b) Labeled orthophoto. Reprinted from Kragh et al. (2017).

For capturing ground truth information on the moving obstacles, a DJI Matrice 100 was used to hover above the subset of the field, in which the human subjects were walking. The drone recorded video of the walking paths and had multiple GPS markers within its field of view. This enabled estimation of a frame-wise transform between GPS and pixel coordinates, similar to the one described above. However, since the camera on the drone was tilted, a projective transform was used instead of a similarity transform. The video was manually annotated frame-wise using the vatic video annotation tool (Vondrick et al., 2013). This provided point-wise GPS positions of all dynamic obstacles for each frame. Finally, it was manually synchronized to the SuperSensorKit recordings, thus providing temporal ground truth information across the entire field. The annotated human tracks are shown beneath each subject in Figure 2.16.

DK7: HCA Airport

The purpose of the seventh field trial was to extend the scenarios captured in DK6 with another environment. The dataset was recorded during a demonstration of the SAFE project at a grass field located at HCA Airport in Odense, Denmark. The stereo camera, thermal camera, and lidar from the SuperSensorKit were mounted on a robot developed by the project partner Conpleks Innovation. The robot itself provided accurate localization from combined IMU, RTK GPS, and wheel encoders. Figure 2.18a illustrates the sensors mounted on the robot, while Figure 2.18b shows a drone image of the scenario from above.

The system was used to demonstrate real-time human obstacle detection and avoidance using stereo vision and lidar. While running detection algorithms, the raw sensor data and localization outputs were recorded for post-processing. A ground truth map of static obstacles was available from the airport in which the dataset was recorded, whereas the moving human obstacles were captured and georeferenced from drone videos as described for DK6.

The dataset has not yet been used in any scientific publications. However, according to plan, the data will be used as a test set in a finalized version of Paper 9 for multi-modal semantic segmentation in 3D.

(a) (b)

Figure 2.18: DK7: HCA Airport. (a) Sensors mounted on robot. (b) Drone image of robot and obstacles from above.

3 Shrimp

The content of this chapter partly appears in the following publication:

> Paper 5: *Kragh and Underwood (2017). Multi-modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. Submitted to International Journal of Robotics Research, February 2017.*

3.1 Sensors

The robotic platform in Figure 3.1, Shrimp, has been developed at the Australian Centre for Field Robotics (ACFR) in Sydney, Australia. The platform is based on a Segway RMP 400 module and includes multiple sensing modalities such as a thermal camera, stereo camera, lidars and a panospheric camera (360° view). In addition to these, it has an advanced Novatel INS localization system, providing accurate 6 degrees of freedom position and orientation estimates from combined RTK GPS and IMU. Table 3.1 and Table 3.2 list the exteroceptive and proprioceptive sensors available on Shrimp. In this thesis, however, only the Ladybug 360° camera, the Velodyne HDL-64E S2 lidar, and the Novatel localization system are used. The Velodyne HDL-64E S2 lidar rotates at 20 Hz with 64 vertically oriented laser beams operating at a wavelength of 905 nm. It generates 3D point clouds with up to 70,000 points per frame at 20 fps.

Table 3.1: Exteroceptive sensors on Shrimp.

Sensor	Model	Resolution	FOV	Range	Rate
Stereo camera	Point Grey Bumblebee XB3	1280 × 960	66° × 50°	-	15 fps
360° camera	Ladybug 3	6 × 1600 × 1200	360° × >280°	-	5 fps
Thermal camera	Raytheon Thermal-Eye 2000B	320 × 240	46° × 35°	-	25 fps
2D Lidar × 2	SICK LMS200	764 × 1	180°	0–80 m	75 fps
3D Lidar	Velodyne HDL-64E S2	2172 × 64	360° × 26.3°	1–100 m	10 fps

Table 3.2: Proprioceptive sensors on Shrimp.

Sensor	Model	Rate
GPS	Novatel SPAN OEM3	50 Hz
IMU	Honeywell HG1700	50 Hz

Figure 3.1: Robotic platform “Shrimp”. Reprinted from Kragh and Underwood (2017).

3.2 Calibration and Registration

All extrinsic parameters defining transformations between sensors were extracted from the CAD model of the platform. The transformations between the Ladybug cameras and the Velodyne lidar, however, were refined using an unsupervised calibration method for cameras and lasers (Levinson and Thrun, 2013). The calibration method was performed for each of the 6 cameras in the Ladybug camera system.

The Ladybug camera system was calibrated from the factory. Exact focal lengths, principal points, and distortion parameters were thus available for all 6 cameras. Similarly, the Velodyne lidar was calibrated from factory for both laser offsets and reflectances.

3.3 Data Collection

The datasets from Australia were all recorded by the ACFR team, whereas the ground truth annotations were made as part of this thesis work. The raw data acquisition does therefore not represent a contribution of this thesis. However, similar to the FieldSAFE dataset (DK6), the datasets from Australia have been made publicly available at http://data.acfr.usyd.edu.au/ag/2017-orchards-and-dairy-obstacles/ and are described in Paper 5 (Kragh and Underwood, 2017).

The Shrimp recording platform described in section 3.1 was used to record 5 datasets at various locations in Australia during 2013. Figure 3.2 shows the locations, shapes, and appearances of the fields that were traversed during acquisition. Table 3.3 provides details for each dataset such as the season of recording, area of the field/orchard, and included obstacle types. For all recordings, Shrimp was remotely controlled to traverse the field or orchard while recording localization data for post-processing. Figure 3.3 illustrates examples of images, point clouds, and annotations from each of the five datasets. The datasets present a large variation in both appearance and geometry with four different orchards and a dairy grass field. Various obstacles including humans, cows, buildings, vehicles, trees, and hills are represented across the datasets.

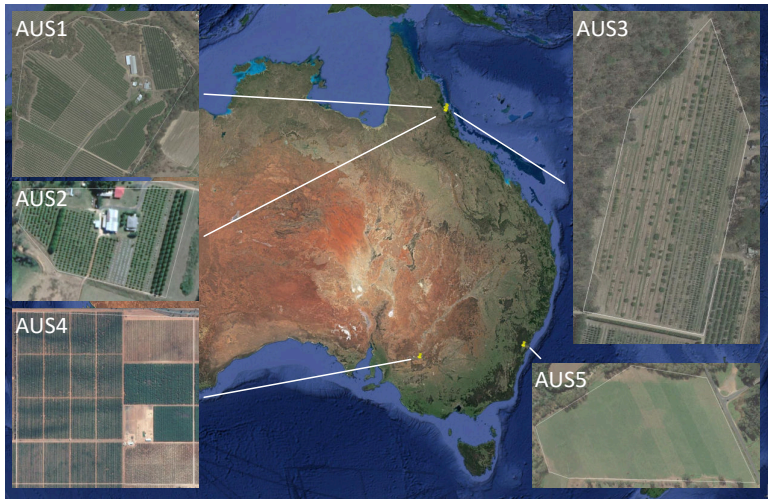


Figure 3.2: Overview of datasets collected with Shrimp in Australia.

A total of 120 pairs of synchronized images and point clouds have been manually annotated pixel- and point-wise into 9 different classes: *ground*, *sky*, *vegetation*, *building*, *vehicle*, *human*, *animal*, and *other*. The *sky* class, however, is only present in the images due to the physics of the lidar.

Table 3.3: Overview of datasets recorded in Australia.

ID	Dataset	Setting	Season	Area (ha)	Length	Labeled frames	Obstacles			
							Human	Animal	Vehicle	Building
AUS1	**Mangoes** *December 2013*	Orchard	Summer	32.6	408 m (359 s)	36	✓		✓	✓
AUS2	**Lychees** *December 2013*	Orchard	Summer	3.5	122 m (121 s)	15	✓		✓	✓
AUS3	**Custard apples** *December 2013*	Orchard	Summer	5.8	159 m (128 s)	23	✓		✓	
AUS4	**Almonds** *August 2013*	Orchard	Spring	183.0	258 m (212 s)	31	✓		✓	✓
AUS5	**Dairy** *May 2013*	Field	Winter	13.6	91 m (106 s)	15	✓	✓		

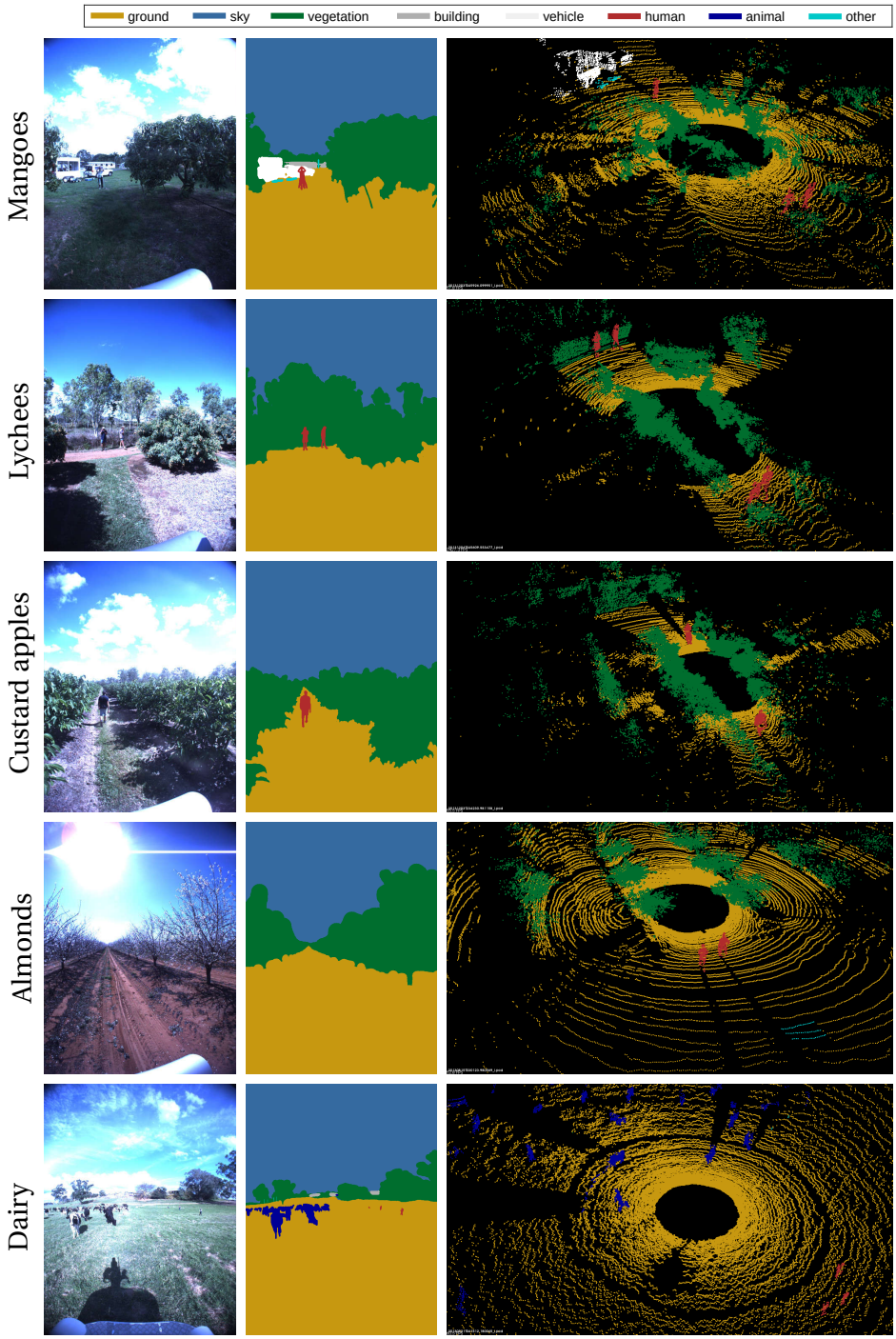


Figure 3.3: Examples of images, point clouds, and annotations from the Australian datasets AUS1-5.

4 Concluding Remarks

In this part of the study, two research perception platforms were described along with associated multi-modal datasets. The platforms both included a 3D lidar, a stereo camera, a thermal camera, a 360° camera, and localization from combined IMU and accurate GPS. Calibration and registration procedures were presented for enabling sensor fusion across modalities. An automated procedure was further proposed for optimizing localization. The platforms have been used to acquire multiple datasets including various obstacles in different agricultural environments.

The acquired datasets represent a wide range of realistic agricultural environments including grass fields, row crops, and orchards with mangoes, lychees, apples, and almonds. As such, they are useful for evaluating methods in their ability to handle different obstacle appearances and illumination conditions, and in their ability to transfer across domains. Despite great efforts, no recordings of wild animals in their natural habitat were acquired. For applications targeting animal safety, such datasets would be invaluable. Furthermore, no recordings were carried out at night-time, nor in bad weather such as heavy rain, dense fog, or strong and dusty wind. Therefore, future work on data acquisition should focus on capturing more seasons and weather conditions and include more variation in obstacle appearances.

Ground truth object labels have only been annotated sparsely across the multiple datasets and sensing modalities. Manual annotation is both tedious and time-consuming, especially when annotating the same obstacles in multiple modalities simultaneously. Therefore, a method for obtaining multi-class ground truth annotations of both static and dynamic obstacles in global GPS coordinates was proposed. Orthophotos captured with drones were georeferenced and manually annotated pixel-wise such that 3D sensor data could be labeled automatically using the localization system of the sensor platform. Although multiple error sources caused slight misalignments in the annotations, the semi-automated annotation procedure has shown that GPS-based labeling of multiple modalities on large-scale is possible. The dataset has been made publicly available to facilitate future research on multi-modal obstacle detection in agricultural environments.

Point Cloud Classification Part III

Autonomous vehicles performing obstacle detection and avoidance often use lidar sensors to measure range data in front of the vehicles. A 3D point cloud of the environment is ideal for detecting non-traversable areas and can further be used for path planning and vehicle control. Obstacle detection is commonly accomplished by segmenting the point clouds into traversable and non-traversable areas using ground plane extraction. Ideally, this provides segmented 3D point structures of all elements protruding from the ground. For an autonomous system to make informed and reasonable decisions, however, a subdivision of non-traversable structures may be needed. Some structures may represent static obstacles that can easily and safely be passed at close distance, whereas others may represent dynamic objects such as pedestrians, cyclists, and vehicles that can make sudden and unexpected movements. Point cloud classification deals with this issue by discriminating point structures based on their shape and neighborhoods.

Traditional pipeline

A common and traditional approach to object recognition and point cloud classification is to apply a pipeline of segmentation, feature extraction, and classification (Douillard et al., 2011).

Segmentation The point cloud is first segmented using ground plane segmentation. For dense point clouds, ground plane segmentation is usually carried out by voxelizing the point cloud into a grid followed by clustering based on height differences (Thrun et al., 2006; Douillard et al., 2010b). Other approaches use more advances methods such as constructing probabilistic elevation maps (Lang et al., 2007) or estimating complex surface functions (Hadsell et al., 2010). After ground segmentation, all remaining voxels are clustered to form non-ground partitions. For sparse point clouds, however, voxelizing with a constant grid size results in a high number of empty cells. For Velodyne lidar data for instance, the number of empty cells increases with distance. Instead of voxelization, other methods such as mesh construction followed by gradient-based region growing can therefore be used (Moosmann et al., 2009).

Feature extraction Often, the segmentation step uses a number of low-level features such as estimated normals, voxel means and variances, and local gradients as mentioned above. However, for actual classification of individual points or segments, more advanced features are typically applied. Local feature descriptors are extracted before segmentation, whereas global descriptors are extracted after segmentation at the non-ground partitions. Popular local feature descriptors include Point Feature Histograms (PFH) (Rusu et al., 2008), the closely related Fast Point Feature Histograms (FPFH) (Rusu et al., 2009), the Signature of Histograms of OrienTations (Salti et al., 2014), and the Normal Aligned Radial Feature descriptor (NARF) (Steder et al., 2011) for 2.5D point clouds (range images). Popular global feature descriptors include Spin Images (Johnson and Hebert, 1999) and Viewpoint Feature Histograms (VFH) (Rusu et al., 2010). Local

descriptors use a neighborhood around each point to calculate the features. The neighborhood of a point can include either the k nearest points or all points within a fixed radius.

For unstructured environments with rough terrain and high vegetation, custom-made features have been proposed for distinguishing man-made structures from vegetation (Vandapel et al., 2004; Lalonde et al., 2006). These all use principal component analysis (PCA) to describe local point neighborhoods as either planar, linear, or scattered. As man-made structures tend to be either planar or linear, and point distributions from vegetation tend to be scattered, PCA features are useful for discrimination. Laser reflectance values have further been shown to provide some discrimination abilities for e.g. vegetation (Wellington and Stentz, 2004; Wellington et al., 2006), however only if carefully calibrated across laser beams (Levinson and Thrun, 2014).

Classification After feature extraction, a classifier can be trained to distinguish a number of classes. For point-wise feature extraction, the classifier provides a class label for each point, whereas for voxel-based feature extraction, all points within a voxel are assigned the same label. Support vector machine (SVM) classifiers have been used widely for both point-wise and segment-wise classification (Himmelsbach et al., 2009; Zhu et al., 2010; McDaniel et al., 2010). However, other approaches such as naive Bayes classification (Vandapel et al., 2004; Lalonde et al., 2006), k-nearest neighbors (KNN), and boosting algorithms have also been applied (Douillard et al., 2009; Golovinskiy et al., 2009). Moreover, Markov random fields (Wellington et al., 2005; Anguelov et al., 2005; Häselich et al., 2013) and conditional random fields (Lim and Suter, 2009) have been shown to provide both accurate and smooth predictions, as they seek to infer optimal class labels of all points jointly.

Deep Learning

A recent trend within point cloud classification uses deep learning methods to jointly learn new, hierarchical features and perform classification with multi-layered neural networks. For image recognition and semantic segmentation in 2D, deep learning has outperformed traditional approaches and even surpassed human performance on certain tasks (LeCun et al., 2015). Convolutional neural networks (CNNs) utilize the grid-based data representation in images by convolving 2D filter kernels across the input space. This generates a feature map, which, after more series of convolutions and pooling operations, contributes to a hierarchical feature representation. The concept easily generalizes to 3D data, as 2D filters can be replaced by 3D filters. However, 3D point clouds are rarely represented in grids and therefore need to be voxelized beforehand. The Voxnet network proposed by Maturana and Scherer (2015) voxelizes both 3D computer-aided design (CAD) data from the ModelNet dataset (Wu et al., 2015) and point clouds from the Sydney Urban Objects Dataset (Quadros et al., 2012) before successfully applying a 3D CNN for object detection. The network outperforms existing methods,

but can only handle input resolutions of 32^3 voxels due to comprehensive increases in memory consumption and computational complexity when going from 2D to 3D. The Octnet network has since then increased the input resolution to 256^3 by utilizing an efficient octree data representation that avoids redundant computations and storage of empty voxels (Riegler et al., 2017). However, for point-wise classification of complete 3D scans from e.g. a Velodyne lidar, a much higher input resolution is needed. Therefore, voxelized approaches typically require prior region extraction to provide relevant segments for the 3D CNN to process (Maturana and Scherer, 2015).

While voxelization may work well for dense point clouds that have an approximately constant point density, it poses a number of problems for sparse data. Point clouds from a multi-beam rotating lidar such as the Velodyne experience point densities that decrease with distance. For these, voxelization throws away detailed information at close distance and results in an increasing number of empty voxels at far distance. Therefore, other data representations have been proposed to handle sparse point clouds. The Pointnet network samples a number of points from a point cloud into an unordered set (Qi et al., 2017a). By only applying permutation-invariant operations, the network extracts both point-wise and global features that are successfully used for object detection and semantic segmentation. As it does not incorporate local point context, other variants have extended the approach with hierarchical feature learning using local neighborhoods (Qi et al., 2017b; Shen et al., 2017). Another recent trend for deep learning on irregular domains applies convolutions on graph structures. These are typically applied in the spectral domain, however, with various challenges concerning weight sharing and compatibility of different graph structures in the same model (Yi et al., 2017). Simonovsky and Komodakis (2017) therefore propose a variant that avoids the spectral domain representation and thus allows for arbitrary graph structures. They report results on the Sydney Urban Objects point cloud dataset that surpass state-of-the-art. Another approach is to represent 3D objects by 2D rendered views. Shi et al. (2015) apply a 2D CNN on a single panoramic view, whereas Su et al. (2015) combine multiple 2D CNNs on a collection of 2D rendered views. De Deuge et al. (2013) propose a 2D range image representation for 3D point clouds acquired with a rotating multi-beam lidar. The representation provides a 2D grid with neighborhoods defined by the horizontal and vertical laser sampling and not the actual range measurements. 2D CNNs using range images have been proposed for vehicle detection on the KITTI (Geiger et al., 2013) autonomous driving dataset (Li et al., 2016; Wu et al., 2017). And more recently, range images have been combined with bird's-eye view 2D grid maps on the same dataset to provide multiple 2D views of the point clouds, thus increasing classification accuracy (Chen et al., 2017).

In this part of the study, two methods for point-wise classification of lidar-acquired 3D point clouds are presented. In the first chapter, a "traditional" approach is presented using hand-crafted feature extraction followed by point-wise classification with SVMs. In the second chapter, a deep learning approach is proposed, converting 3D point clouds to 2D range images, followed by 2D semantic segmentation with a CNN.

5 Object Detection and Terrain Classification with SVM

The content of this chapter partly appears in the following two publications:

> Paper 4: *Kragh et al. (2015). Object Detection and Terrain Classification in Agricultural Fields using 3D Lidar Data. Proceedings of International Conference on Computer Vision Systems 2015 (ICVS2015), Vol. 9163 Springer, p. 188-197.*
>
> Paper 5: *Kragh and Underwood (2017). Multi-modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. Submitted to International Journal of Robotics Research, February 2017.*

In the automotive industry, lidar sensing is used widely to detect and localize objects from vehicles in urban environments. Popular approaches include ground plane detection and extraction, which allows for subsequent segmentation of protruding objects (Douillard et al., 2011; Luettel et al., 2012). In agriculture, however, rough terrain and high vegetation complicates ground plane segmentation and challenges the distinction between traversable and non-traversable terrain. Depending on the agricultural context, a subdivision of non-traversable ground into vegetation and objects may further be needed.

In this chapter, point-wise classification of 3D lidar point clouds using an adaptive neighborhood radius is investigated. Each point is classified as either *ground*, *vegetation*, or *object*. The *ground* class denotes traversable terrain, whereas the *object* class denotes obstacles. The *vegetation* class includes crops, bushes, and trees and may be either processable (and traversable) or non-traversable depending on the specific agricultural task.

The proposed method works on individual scans from a multi-beam lidar generating 3D point clouds with each N points. For each point, 13 different features are extracted based on local point neighborhood statistics. The features describe height, shape, orientation, distance, and reflectance of the point neighborhoods and are used for supervised classification into three classes using an SVM classifier.

The method consists of three steps: preprocessing, feature extraction, and classification. In the following sections, each of the three steps is described in detail.

5.1 Preprocessing

A preprocessing step first transforms the point cloud with a translation and rotation such that the z-axis is approximately vertical. This is done by aligning the xy-plane with a globally estimated plane. A minimum filter with a fixed radius of 1.0 m is used to handle the varying point density, and the RANSAC algorithm (Fischler and Bolles, 1981) is used for estimating plane coefficients. The normal vector of the plane defines the transformation necessary to align the ground plane and the xy-plane.

5.2 Feature Extraction

3D point clouds can originate from a wide range of sources. They can be sampled from CAD models, derived from camera depth images, accumulated from push-broom laser range scanners, or be output directly by multi-beam lidars. For some sources, the point distribution is roughly constant, and the point cloud can be denoted as dense. For other sources, the point distribution varies with distance and dimensions (e.g. azimuth and elevation), in which case the point cloud can be denoted as sparse. When calculating point features using a local neighborhood around each point, the point density is crucial in determining a reasonable neighborhood radius. A large radius is useful for low point densities, but results in coarse estimates. A small radius, on the other hand, is useful for high point densities, but results in noisy and possibly undefined features at places of low density.

A point cloud from a rotating, multi-beam lidar is sparse with its point density decreasing with distance. Figure 5.1 illustrates a single laser beam pointing towards a flat ground plane from a certain height with a certain angle. Figure 5.1a illustrates the scan pattern generated on the ground with equally distributed points along a circle, while Figure 5.1b shows a top-down view of the same scenario. The circle radius $\|\mathbf{p}\|_{xy}$ denotes the ground plane distance between the sensor and the point $\mathbf{p}$. Let θ_H denote the angular resolution of the laser horizontally. Then the distance between two neighboring points is $2\|\mathbf{p}\|_{xy}\sin\frac{\theta_H}{2}$. In order to achieve a constant number of neighborhood points M regardless of the point distance, an adaptive neighborhood radius r must be:

$$r = 2\|\mathbf{p}\|_{xy}\sin\frac{M\theta_H}{4}. \tag{5.1}$$

The neighborhood radius scales linearly with the horizontal distance $\|\mathbf{p}\|_{xy}$. For multi-beam lidars, this is only an approximation, since a point neighborhood will also include points from the beams above and below point $\mathbf{p}$. However, since the angular resolution

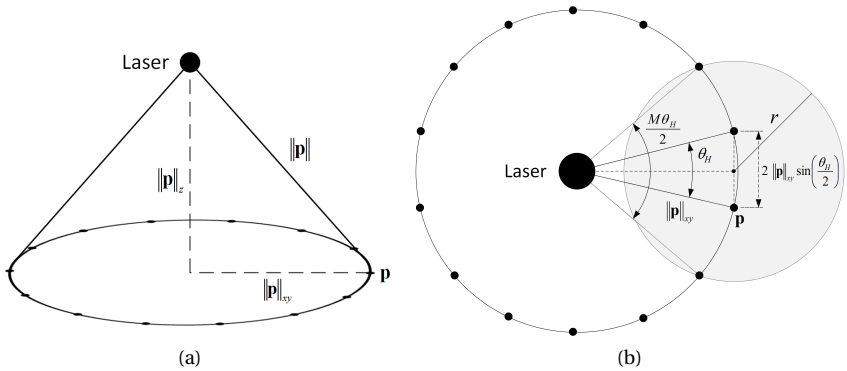


Figure 5.1: Example of adaptive neighborhood radius for single-beam lidar with $M = 4$. (a) Circular scan pattern on flat ground surface. (b) Overlaid adaptive neighborhood radius. Adapted from Kragh and Underwood (2017).

is normally much higher horizontally than vertically ($\theta_H < \theta_V$), Equation 5.1 serves as a good approximation.

The adaptive scaling allows for fine feature estimates close to the sensor and coarse estimates at long distances, while benefiting from computational simplicity due to the linear relationship. However, it also requires features to be scale-invariant such that the features themselves are not influenced by the neighborhood radius. Therefore, features are designed explicitly for scale-invariance by normalizing with respect to r when appropriate. Table 5.1 lists the 13 proposed features that are calculated for each evaluated point i having k neighbors. The features are divided into five groups: height, shape, orientation, distance, and reflectance.

f_1-f_4 are height features and describe statistics of the z-coordinates of the point and its neighborhood. f_5-f_7 are shape features derived from a principal components analysis (PCA) of a 3×3 covariance matrix of the point neighborhood. The eigenvalues $\lambda_1 < \lambda_2 < \lambda_3$ describe the neighborhood point distribution in terms of its scatteredness, linearity, and planarity. These measures are useful for distinguishing e.g. vegetation from planar objects, since vegetation typically has a more scattered point distribution than planar-like objects such as a vehicle or building. f_8 is another measure of planarity proposed by McDaniel et al. (2010) that uses the 3D vector $\vec{p}_j$ of a neighborhood point j as well as the neighborhood mean/centroid $\overline{\vec{p}}$. f_9-f_{11} are orientation features derived from the eigenvector $\vec{v}_1$ corresponding to the smallest eigenvalue λ_1. The eigenvector $\vec{v}_1$ defines the normal vector of a locally estimated plane. Its orientation can thus be used to distinguish e.g. horizontal from vertical surfaces. f_{12} is a distance feature proposed by Wellington and Stentz (2004). Although the adaptive neighborhood radius compensates for distance, f_{12} enables the subsequent classifier to further characterize points by their neighborhood density. f_{13} denotes the reflectance intensity, which is a value directly provided by the lidar sensor.

Table 5.1: Scale-invariant point features.

Type	Feature	Description	Definition
Height	f_1	Point height	z_i
	f_2	Minimum neighborhood height	$min(z_1 \dots z_k)$
	f_3	Mean neighborhood height	$\overline{z} = \frac{1}{k}\sum_{j=1}^{k} z_j$
	f_4	Neighborhood height standard deviation	$\frac{1}{r}\sqrt{\frac{1}{k}\sum_{j=1}^{k}\left(z_j - \overline{z}\right)^2}$
Shape	f_5	Scatteredness	$\frac{\lambda_1}{\lambda_3}$
	f_6	Linearity	$\frac{\lambda_2 - \lambda_1}{\lambda_3}$
	f_7	Planarity	$\frac{\lambda_3 - \lambda_2}{\lambda_3}$
	f_8	Normalized orthogonal residual sum of squares (RSS)	$\frac{1}{k}\sum_{j=1}^{k}\left(\left(\vec{p}_j - \overline{\vec{p}}\right)\cdot\vec{v}_1\right)^2$
Orientation	f_9	Normal vector x	$\vec{v}_1 \cdot (1,0,0)$
	f_{10}	Normal vector y	$\vec{v}_1 \cdot (0,1,0)$
	f_{11}	Normal vector z	$\vec{v}_1 \cdot (0,0,1)$
Distance	f_{12}	Point distance	$\|\vec{p}_i\|$
Reflectance	f_{13}	Reflectance intensity	intensity_i

5.3 Classification

The 13-dimensional feature vectors of annotated points are used to train an SVM with probability estimates using a one-against-one approach with the libsvm library (Chang and Lin, 2011). A radial basis function (RBF) kernel is used along with the default values $C = 1$ and $\gamma = \frac{1}{\#features} = \frac{1}{13}$. In order to handle class imbalance, an equal amount of points are drawn at random from each class. All features are further normalized by subtracting the mean and dividing by the standard deviation for each dimension across the training set. The normalization parameters are stored and reused at test time.

5.4 Results and Discussion

The method has been evaluated on the DK1 dataset (page 24) with point-wise annotations into three classes: *ground*, *vegetation*, and *object*. The dataset includes 15 annotated lidar frames from a Velodyne HDL-32E. Leave-one-out cross-validation was applied to split training and testing across 7 different trials (recordings).

Table 5.2 shows a confusion matrix generated by accumulating point predictions across the cross-validation folds. The method classifies *ground* points accurately, whereas it quite often confuses *vegetation* and *object* points. Figure 5.2 further shows example frames from the same dataset with ground truth annotations on the left side and classifier

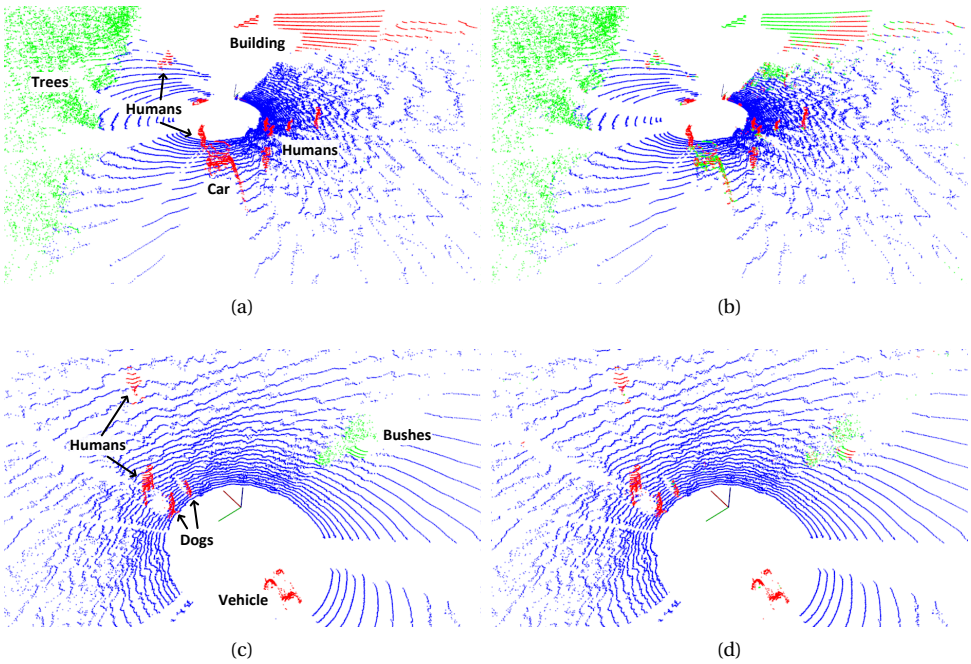


Figure 5.2: Examples of classification results. a) and b) respectively show ground truth and classification results of a scene with ground, trees, humans, a car, and a building. c) and d) respectively show ground truth and classification results of a scene with ground, bushes, humans, and dogs. Blue denotes ground, green denotes vegetation, and red denotes objects. Reprinted from Kragh et al. (2015).

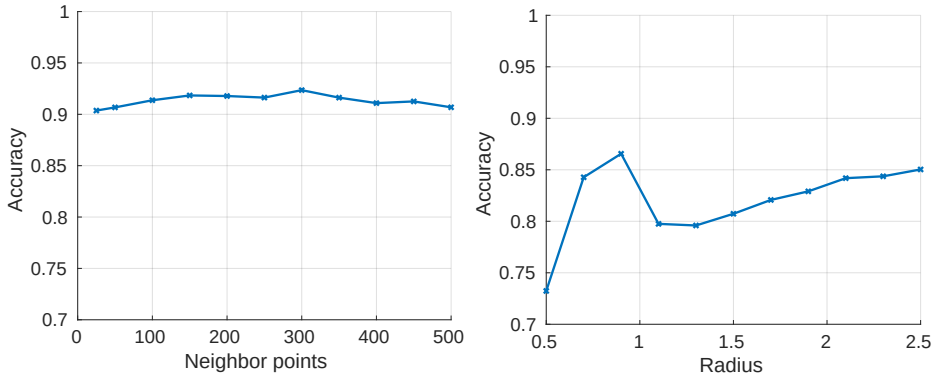
predictions on the right side. The *object-vegetation* confusion is clearly seen by the car and the building in Figure 5.2b.

The adaptive neighborhood radius in Equation 5.1 includes a single parameter, the approximate number of neighborhood points M. Figure 5.3a shows a graph of the accuracy evaluated with different values of M. The accuracy was above 90% for all evaluated values, with a maximum of 92.4% when $M = 300$. The method was compared to the Fast Point Feature Histograms (FPFH) descriptor (Rusu et al., 2009) followed by SVM classification. Figure 5.3b shows the accuracy with different values of the constant sized neighborhood radius r required by the FPFH feature descriptor. For all evaluated radii, FPFH gave inferior results and involved considerable longer computation times.

The list of features in Table 5.1 indicates a potentially high correlation between certain features. The four height features, for instance, may not all be necessary in order to distinguish structures based on height. Two feature selection techniques were therefore applied to examine the relative importance of the 13 proposed features. The two techniques both evaluate feature combinations and use the test set accuracy after SVM

Table 5.2: Confusion matrix relating predictions (columns) to ground truth (rows). Adapted from Kragh et al. (2015).

	Ground	Vegetation	Object
Ground	94.1%	3.2%	2.7%
Vegetation	6.4%	81.5%	12.1%
Object	3.3%	7.5%	89.2%

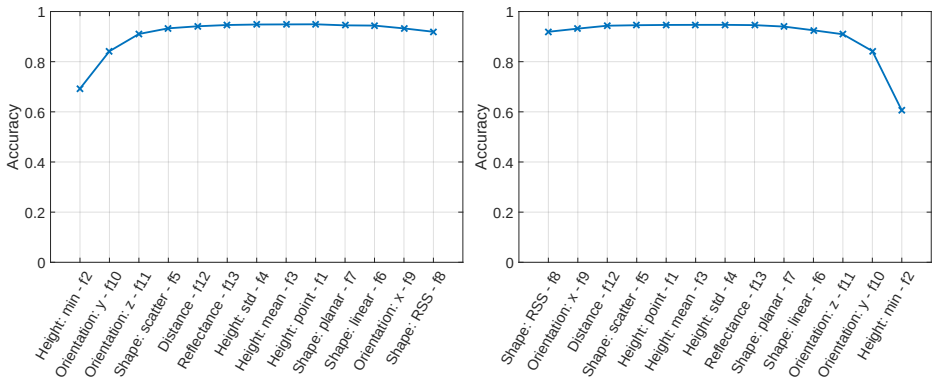


(a) Proposed features (b) FPFH features

Figure 5.3: Comparison of proposed features and FPFH descriptors. (a) Accuracy with proposed features as function of M. (b) Accuracy with FPFH descriptor as function of r.

classification as a common metric for comparison. As an exhaustive search would include $\sum_{f=1}^{13}\binom{13}{f} = 8191$ combinations, only a subset of all combinations is evaluated.

Greedy forward selection starts with an empty list of features. The list is grown incrementally by continuously adding the single feature among the remaining that gives the highest combined accuracy. Greedy backward selection, on the other hand, starts with a list of all features. The list is then reduced incrementally by continuously removing the single feature that, when removed, causes the smallest decrease in accuracy. Figure 5.4 shows the feature relevance sorting of the 13 proposed features using greedy forward and backward selection. The two approaches both had one feature from each of the categories height, shape, and orientation among their four most important features. The approaches further agreed that f_2 (minimum height) was the most important feature and that f_8 (RSS) was the least important feature. The graphs show that using more than 5 features did not significantly increase performance. In fact, both graphs show small decreases when using more than 10 features, which suggests that a low number of features should be used when the amount of annotated data is limited. A low number of features will further decrease the computational complexity and execution time.



(a) Forward selection (b) Backward elimination

Figure 5.4: Feature selection using greedy forward selection and backward elimination. The evaluation was conducted on the DK1 dataset. Reprinted from Kragh et al. (2015).

6 Semantic Segmentation in 3D with Range Images

The content of this chapter partly appears in the following draft of a journal paper:

> Paper 9: *Kragh et al. (2018). Multi-Modal Semantic Segmentation in 3D with Range Images. Draft, February 2018.*

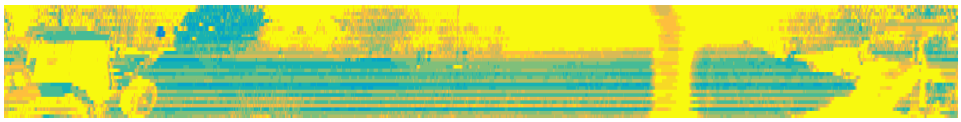
In the previous chapter, a "traditional" approach was presented for point-wise classification of 3D point clouds. A number of hand-crafted features were extracted for each point based on its point neighborhood, and an SVM classifier was used to distinguish a number of classes based on the features. Until recently, this was the preferred approach for object detection and semantic segmentation in both 2D images and 3D point clouds. However, recent advances in deep learning have outperformed traditional approaches in numerous research fields within computer vision and machine learning (LeCun et al., 2015). Multi-layered convolutional neural networks automatically learn features that are relevant for the specific classification tasks. Both feature representations and subsequent classifier decision boundaries are thus made data-driven.

In this chapter, point-wise classification of 3D lidar point clouds using deep learning is investigated. Each point cloud is converted to a 2D range image, and a state-of-the-art fully convolutional network (FCN) is applied for semantic segmentation. The method is evaluated on the FieldSAFE dataset DK6 (page 27) that was acquired in an agricultural grass field. A semi-automated annotation process is used to label the lidar frames point-wise into 6 classes: *grass*, *vegetation*, *human*, *road*, *building*, and *object*.

The proposed method is applied on individual frames from a multi-beam lidar generating 3D point clouds. Each point cloud is converted into a 2D range image with the x- and y-axes describing horizontal and vertical laser angles and with intensities corresponding to range measurements. An FCN for semantic segmentation on images is trained to predict pixel-wise class labels, thus providing point-wise predictions for each lidar frame.

In the following, the range image conversion and network architecture are described individually. This is followed by a description of the semi-automated annotation process and a presentation of preliminary results.

(a) Range (pseudo-colored)



(b) Intensity (pseudo-colored)

Figure 6.1: Channel examples represented in range image format. Blue represents low range/intensity, whereas yellow represents high range/intensity. Adapted from Kragh et al. (2018).

6.1 Range Image Representation

A 3D point cloud from a stationary lidar is often referred to as 2.5D, since it is acquired from a single view point. Such a point cloud is easily converted to a range image representation, in which the first and second axes describe azimuth and elevation of the laser beam, and the intensity describes the range measurement. Range, azimuth, and elevation are computed from Cartesian x, y, and z coordinates in the local sensor frame:

$$r = \sqrt{x^2 + y^2 + z^2} \tag{6.1}$$

$$\theta = \text{atan2}\left(y, x\right) \tag{6.2}$$

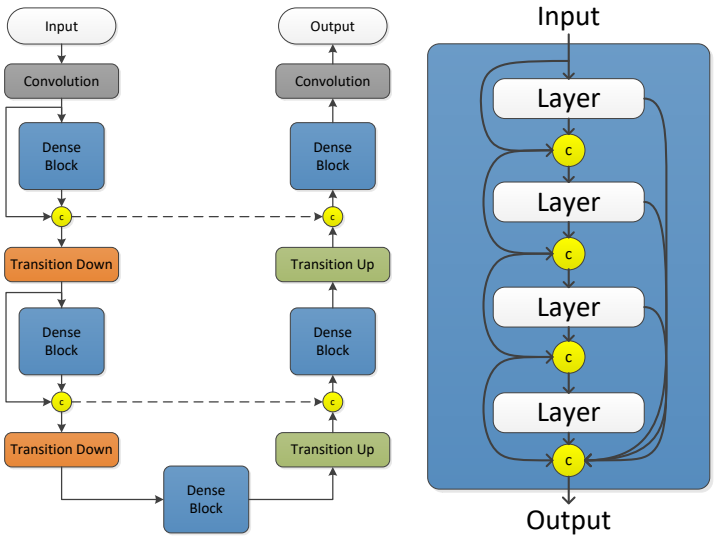
$$\phi = \text{atan2}\left(z, \sqrt{x^2 + y^2}\right) \tag{6.3}$$

The corresponding pixel coordinate has a column position of $\theta/\Delta\theta$ and a row position of $\phi/\Delta\phi$, where $\Delta\theta$ and $\Delta\phi$ are the azimuth and elevation resolutions of the lidar. Subsequent nearest neighbor interpolation ensures that all pixels are defined by assigning each pixel by its nearest projected range value.

Figure 6.1 illustrates an example of a range image acquired with a Velodyne HDL-32E rotating lidar. Along with the range channel, an intensity channel is generated using the reflectances available for each laser return. As input to the neural network, the two channels are converted to floating points and normalized to the range $[0, 1]$. More details on the normalization process are available in Paper 9.

6.2 Network Architecture and Training

Figure 6.2a illustrates the network architecture of a state-of-the-art FCN proposed by Jégou et al. (2017). The network is an extension of the DenseNet architecture (Huang



(a) (b)

Figure 6.2: Network as proposed by Jégou et al. (2017). (a) Network architecture with concatenations denoted by ©. (b) Dense block consisting of $l = 4$ densely connected composite layers. Adapted from Kragh et al. (2018).

et al., 2016). It includes an encoding path that extracts hierarchical features through downsampling, and a decoding path that combines feature representations during upsampling. This is achieved through skip-connections (dashed arrows) that connect the encoding and decoding paths (Long et al., 2015).

The encoding path consists of an initial 3×3 convolution followed by repeated sequences of dense blocks (DBs), concatenations, and transition downs (TDs). A DB internally consists of l densely connected composite layers that each output k feature maps. A composite layer is made up by batch normalization (Ioffe and Szegedy, 2015), a rectified linear unit (ReLU), a 3×3 convolution followed by a dropout layer (Srivastava et al., 2014). Figure 6.2b illustrates how the composite layers in a DB are connected and combined by concatenation to $l * k$ feature channels. The output of a DB is concatenated with its input such that low-level features are continuously forwarded through the encoding path, thus allowing feature reuse. A TD reduces the spatial dimension using batch normalization, a ReLU, 1×1 convolution, dropout, and 2×2 maximum pooling. The encoding path thus increases the number of feature channels while reducing the spatial dimension.

The decoding path consists of repeated sequences of transition ups (TUs), concatenations, and DBs followed by a final classification made up by a 1×1 convolution and a softmax layer. A TU increases the spatial dimension using a 3×3 transposed convolution with a stride of 2 (Long et al., 2015). Feature maps from the encoding path with the same resolution are then concatenated. This effectively combines low- and high-level features

and allows the network to perform accurate and smooth pixel-wise predictions. The DBs compress the information by reducing the number of feature channels. Contrary to the encoding path, their inputs and outputs are not concatenated. The decoding path thus decreases the number of feature channels while increasing the spatial dimension.

The network is trained for 50 epochs with a batch size of 8 using the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.0001. A weight decay factor of 1×10^{-4} and a dropout rate of 0.2 are used for regularization. As no pre-trained networks are publicly available for 2-channel range images, the network is trained with zero-initialized weights. To fit the network onto a GeForce Titan X GPU with 12 GB RAM, random crops of size 256×256 pixels corresponding to the full height are sampled from the range images and used as input. Horizontal wrap-around at the left and right image boundaries is used, as the range image corresponds to a 360° field of view. Random horizontal flipping further provides data augmentation.

As shown in Figure 6.1, the tractor and a part of the recording platform are visible in all range images. To avoid biasing the network with these observations, the ground truth range images are masked such that no loss is backpropagated for these regions during training. To handle class imbalance in the dataset, a custom loss function is further introduced:

$$H(p,q) = \sum_{x,y} \sum_{c} -p_c(x,y)\, w_c \log\left(q_c(x,y)\right) \tag{6.4}$$

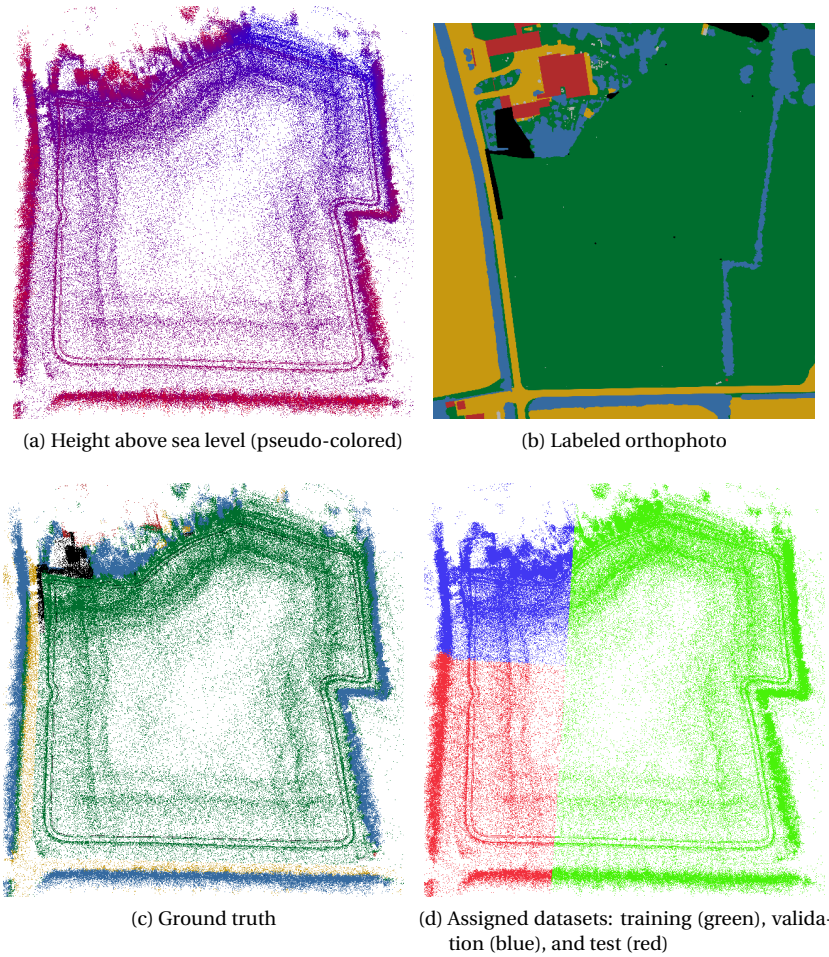
Here, $p_c(x,y)$ denotes the ground truth probability for class c at pixel location (x,y), while $q_c(x,y)$ denotes the predicted probability after the softmax layer. p is 1 for the correct class label and 0 for all other classes. w_c denotes a class weight that makes backpropagated losses depend on the class that was misclassified.

Class imbalance is handled using the number of occurrences within each class. With N_c denoting the number of pixels in the training set with label c, the relative class frequency is defined as $\frac{N_c}{\sum_k^C N_k}$. Three different strategies were evaluated for class weighting. The first used equal weights for all classes, the second used inverse relative class frequencies, and the third used $\log_2$ of the inverse relative class frequencies. For all scenarios, the third method gave the best performance across all classes:

$$w_c = \log_2 \frac{\sum_k^C N_k}{N_c} \tag{6.5}$$

6.3 Results and Discussion

The method has been evaluated on the FieldSAFE dataset DK6 (page 27) that includes different obstacles in a grass field with ground truth annotations in global GPS coordinates. To obtain point-wise class labels for all lidar frames, each point cloud was first georeferenced using the procedure presented in section 2.3. Figure 6.3a shows the resulting accumulated point cloud pseudo-colored with the height above sea level.



(a) Height above sea level (pseudo-colored)

(b) Labeled orthophoto

(c) Ground truth

(d) Assigned datasets: training (green), validation (blue), and test (red)

Figure 6.3: Georeferenced points colored by different channels. For technical reasons, only 10% of all points are shown. Adapted from Kragh et al. (2018) and Kragh et al. (2017).

All points were then projected to the annotated 2D drone orthophoto in Figure 6.3b using a transformation between georeferenced UTM coordinates and orthophoto pixel coordinates as described in section 2.4. Figure 6.3c illustrates the resulting point-wise labels.

Since the dataset only included a single field, it was split geographically into training, validation, and test subsets. Figure 6.3d shows a geographical split which ensured that all classes were represented in all subsets. The split allows the same range image to appear in both the training, validation, and test subsets. However, using the split in a similar way to the masking of range image regions covered by the tractor, no pixels in the ground truth images appeared in more than a single subset. This ensured that the network was

only trained on range image pixels from the training set. However, some unintended correlations may still have existed between subsets.

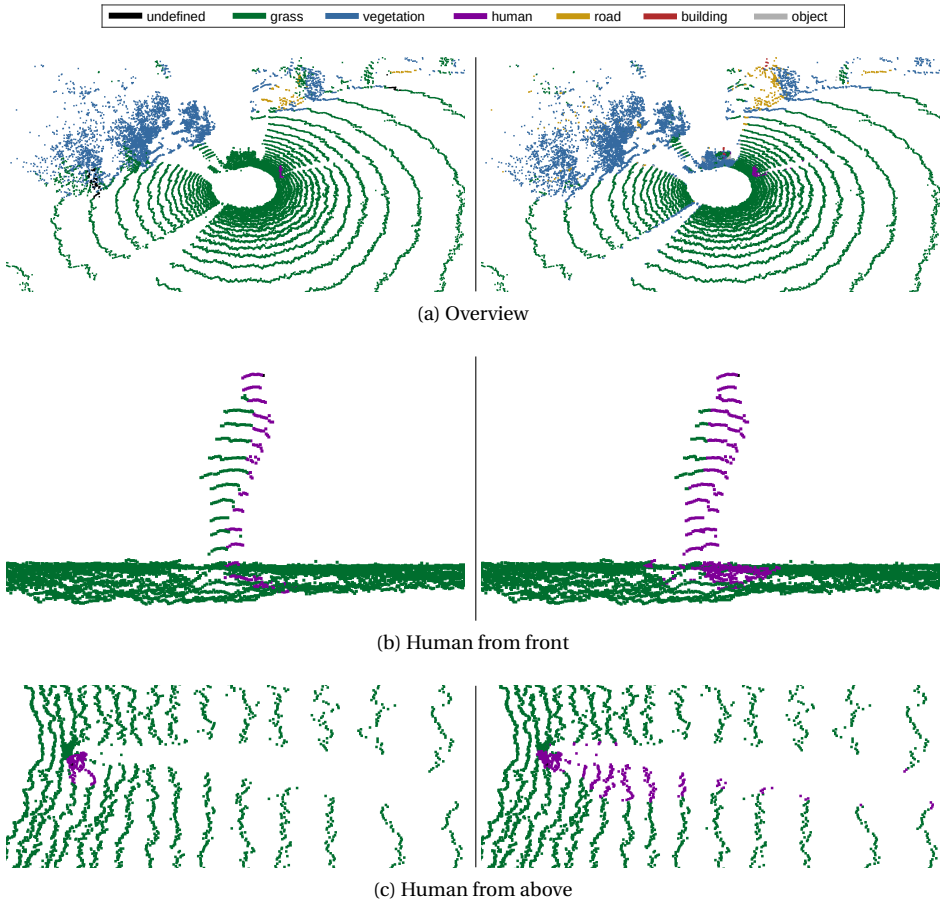
A total of 9,168 range images were generated based on a three lap traversal around the field. The training, validation, and test splits were applied as described above, and labeled images with less than 1000 defined pixels were excluded from each subset. This resulted in 7,837 frames for training, 4,092 frames for validation, and 3,359 frames for testing.

Table 6.1 lists the class-wise results across the entire test set when range and intensity channels were used as input for the FCN. The table reports class-wise intersection over union (IoU) as well as the overall classification accuracy. As is clear from the table, the best performance was achieved when using both the range and intensity channels. The addition of intensity cues thus increased the mean IoU from 0.355 to 0.426. The rather small mean IoU was caused by a performance varying significantly over classes. The *human* and *building* classes had IoUs close to 0, whereas *grass*, *vegetation*, and *road* had IoUs of 0.985, 0.905, and 0.545, respectively. This may be caused by severe class imbalance, as the accuracy for both models were above 97%. Another source of error could be label misalignments, as annotation errors were inevitably introduced during the semi-automated annotation procedure.

Table 6.1: Class-wise classification results comparing the use of only range measurements with the use of both range and intensity. Adapted from Kragh et al. (2018).

	IoU							accuracy
	grass	vegetation	human	road	building	object	mean	
Range	0.979	0.888	0.000	0.246	**0.000**	0.014	0.355	0.975
Range, intensity	**0.985**	**0.905**	**0.020**	**0.545**	**0.000**	**0.103**	**0.426**	**0.981**

Figure 6.4a shows an example of a predicted lidar frame (visualized as a point cloud) along with its ground truth labels. Despite the mixed quantitative results from Table 6.1, the network seemed to correctly predict many of the structures in the environment. Figure 6.4b and 6.4c show two different views of a human in the scene. From these, it is clear that the ground truth annotations (left side) suffered from misalignment, which was likely caused by errors during georeferencing and projection to the annotated orthophoto. However, the network was able to mitigate parts of these errors as seen from the predictions (right side). Possibly, this was due to the class weighting in the custom loss function that favored misclassified *grass* points over misclassified *human* points. In practice, the network thus corrected some misalignments in the ground truth annotations.



(a) Overview

(b) Human from front

(c) Human from above

Figure 6.4: Three views on a single frame from the test set. The left column shows ground truth annotations, whereas the right column shows predictions using a network trained on range and intensity channels. Reprinted from Kragh et al. (2018).

7 Concluding Remarks

In this part of the study, two approaches for point-wise classification of 3D point clouds were proposed. The methods both targeted sparse point clouds from a rotating multi-beam lidar. The first method built on a traditional pipeline including ground plane identification, feature extraction, and classification. Sparsity was handled during feature extraction using an adaptive neighborhood radius that depended on distance and thus also point density. For point-wise distinction of ground, vegetation, and objects, the proposed method outperformed a frequently used 3D feature descriptor on both accuracy and computation time. The second method investigated recent advancements in deep learning on images. How to transfer these from 2D to 3D, however, is not obvious, as local neighborhoods in a sparse point cloud are not well-defined. Therefore, the second method proposed a 2D range image representation of lidar-acquired point clouds to perform semantic segmentation with a state-of-the-art 2D CNN. The range image representation implicitly solved sparsity problems and directly provided 2D grid-based point neighborhoods.

The proposed methods both perform semantic segmentation in 3D. That is, point-wise classification into multiple classes. Although actual obstacle avoidance systems may favor an object detection approach encapsulating detected objects with 3D bounding boxes, point-wise classification serves as a generic representation that allows subsequent clustering, tracking, and even fusion with other modalities. In the domain of autonomous robots, a binary distinction is often made between occupied and unoccupied regions. However, in agriculture, this simplification falls short when the objective is to interact with vegetation e.g. during grass mowing or when cutting tree branches. Therefore, multiple classes may be needed to represent traversable, processable, and non-traversable areas. And for an intelligent systems that reacts according to obstacle types, non-traversable areas may even be subdivided into static (buildings, fences, trees) and dynamic (humans, animals, vehicles) obstacles.

Future work should focus on investigating and comparing other data representations for deep learning on 3D points clouds. A 2D CNN on range images should thus be compared with state-of-the-art voxelized 3D networks (Riegler et al., 2017) and permutation invariant networks such as PointNet (Qi et al., 2017a).

Multi-Modal Fusion Part IV

In order for autonomous vehicles to be safe, the perception system reporting obstacle detections and classifications must be both robust and redundant. The system must be robust towards changes in illumination and weather conditions such that direct sunlight, heavy rain, fog, or dust does not degrade or fully obstruct its abilities to detect obstacles. It must be able to handle perceptive ambiguities between obstacles and non-obstacles such as animals that can be visually camouflaged to look like vegetation, or humans that when lying down can be geometrically similar to the ground. The system must further avoid all single points of failure by introducing redundancies. If one sensor or detection algorithm fails, other duplicates or replacements must be able to compensate. For these reasons, multiple sensors and complementary sensing modalities are needed.

Multi-modal fusion deals with the issue of combining sensor data from different domains in order to increase robustness and confidence. Combining multiple sensors should thus result in reduced uncertainty compared to individual sensor performances. In the scientific field of obstacle detection for autonomous vehicles, typical perception sensors include color cameras, thermal cameras, lidars, and radars (Luettel et al., 2012). All combinations of these sensors are possible, however, the primary focus in this thesis is on fusion between lidar and other modalities.

In some of the pioneering work on autonomous vehicles, proposed during the DARPA Grand Challenge, multiple single-beam lidars were fused with a color camera using self-supervised learning (Dahlkamp et al., 2006). Here, the lidars were used as robust and reliable sensors at close range to continuously supervise the color camera for traversability assessment at far range. Similar approaches have been used for fusing radar and color camera (Milella et al., 2014, 2015), radar and stereo vision (Reina et al., 2016a), and to extend the fusion approach of lidar and color camera (Zhou et al., 2012). Self-supervised learning, however, does not truly belong to the category of sensor fusion, as it does not improve the performance of the supervising sensor (e.g. lidar).

Different stages of sensor fusion are often used to describe at what level the data are combined. Low-level or early fusion refers to the combination of raw data from different sensors, whereas high-level or late fusion refers to the integration of information at decision level. Feature-level fusion is occasionally used to represent intermediate levels where some higher-level information has been extracted from both modalities before fusion. At low- and feature-level, a lidar has been fused with both monocular color, stereo, and thermal cameras for obstacle detection in agricultural environments (Dima et al., 2004; Wellington et al., 2005; Häselich et al., 2013; Mao et al., 2015; Benet et al., 2016). 3D points are projected onto 2D image planes by utilizing known extrinsic and intrinsic parameters of the lidar and cameras. By concatenating 3D coordinates with color or thermal intensities, features that utilize multiple modalities can be extracted. The same concept has been applied for scene analysis, in which 3D point clouds and 2D images are labeled point- and pixel-wise (Namin et al., 2014; Posner et al., 2009; Douillard et al., 2010a; Cadena and Košecká, 2016). Another low-level fusion approach exploits the high reliability and precision of a lidar with the high point density of a stereo

camera to generate a combined reliable and dense point cloud (Huber et al., 2011; Suvei et al., 2018).

At high-level, 2D grid map representations have been used to probabilistically fuse lidar and color camera (Laible et al., 2013; Reina et al., 2016b) and lidar and radar (Ahtiainen et al., 2015) for traversability assessment. For scene analysis, conditional random fields (CRFs) have been used at decision level to fuse intermediate predictions for each modality by including both spatial, temporal, and multi-modal relationships to improve initial classifications (Namin et al., 2015; Xiao et al., 2015; Zhang et al., 2015; Munoz et al., 2012).

Recently, the distinction between low-, feature-, and high-level fusion has been slightly blurred with the use of deep learning. Chen et al. (2017) fuse lidar and camera with a "deep fusion" approach for vehicle detection on the KITTI dataset (Geiger et al., 2013). This is done by adding layer-wise connections between parallel subnetworks for each modality, with the connections based on region proposals in 3D that are projected onto the 2D image. Another variant uses high-level fusion for the same task and dataset on separate CNNs trained on each modality (Asvadi et al., 2017). In other fields of research, a CNN has been used to fuse camera images with remote sensing data on high-level for estimating ocean depth with an autonomous underwater vehicle (Rao et al., 2017). Results showed that sensor fusion improved the accuracy compared to single-modality classifiers, even when one of the modalities was not available during inference. And Eitel et al. (2015) have fused color and depth images on high-level with a CNN for recognizing household objects. Here, sensor fusion helped handle missing or incomplete sensor data from one of the modalities.

In this part of the study, three methods for multi-modal obstacle detection including lidar sensing are presented. In the first chapter, a self-supervised classification system is presented using a lidar for continuously supervising a visual classifier of traversability. In the second chapter, a lidar and a camera are fused at decision level using conditional random fields. In the third chapter, the deep learning approach from chapter 6 is extended with color and temperature channels from a stereo and a thermal camera, thus fusing lidar and camera data with deep learning on range images.

8 Self-Supervised Traversability Assessment

The content of this chapter partly appears in the following publication:

> *Kragh et al. (2016c). Self-supervised Traversability Assessment in Field Environments with Lidar and Camera. Poster presentation at the International Conference on Agricultural Engineering 2016 (CIGR2016).*

Self-supervised learning is a concept in machine learning in which one classification system (the supervisor) outputs labeled data as training examples to another (supervised) classification system. The supervisor is assumed to be a reliable classifier that is able to classify only a part of the input space, whereas the supervised classifier spans the entire input space, but also requires labeled training data. The concept became popular in the 2005 DARPA Grand Challenge as a near-to-far approach (Dahlkamp et al., 2006). Here, a reliable laser sensor was used at near-distance to distinguish and label ground and non-ground areas. By projecting the labeled areas to a camera image, the laser could continuously supervise/train a visual classifier. Since the camera image covered a much wider area than the laser, the visual classifier was effectively trained to distinguish ground and non-ground at both near and far distances.

The near-to-far approach builds on the assumption that a robust and reliable sensor and associated algorithm can find traversable ground regions and feed these into e.g. a visual classifier as training data. This can be done in an online fashion, allowing the visual classifier to continuously adapt to environmental and illumination changes. The concept has been applied in various domains and with various supervisors and supervised classifiers. A subdivision can be made into proprioceptive and exteroceptive learning. Proprioceptive learning typically uses vibration data, whereas exteroceptive learning uses perceived environmental data from exteroceptive sensors such as a camera, laser, or radar. Proprioceptive learning has explored the use of vibration-based terrain classification on both planetary rovers and off-road vehicles to label previous camera images of the same areas (Brooks and Iagnemma, 2012; Kim et al., 2006). It has further been used to distinguish vegetation and ground in various outdoor environments (Wurm et al., 2014). Exteroceptive learning, on the other hand, has been used for traversability assessment both indoors and outdoors with laser and camera (Maier et al., 2011), with stereo and monocular cameras (Hadsell et al., 2009), and with a reverse optical flow

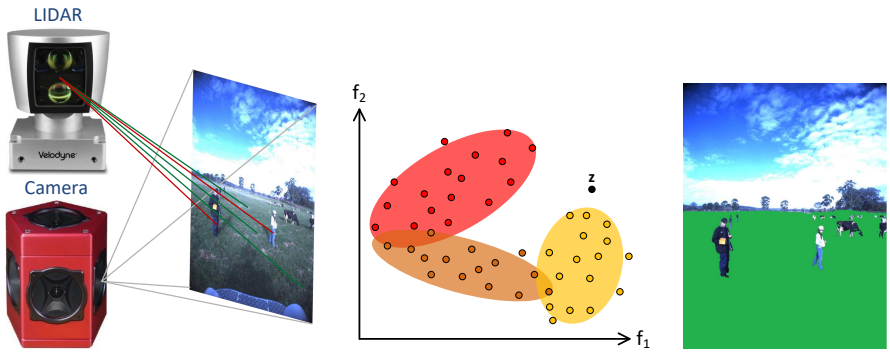


Figure 8.1: Camera-based traversability assessment supervised by lidar. (a) Lidar-classified ground and non-ground 3D points are projected onto 2D image. (b) Gaussian mixture model of normal ground appearance. (c) Ground truth traversable area overlaid on image.

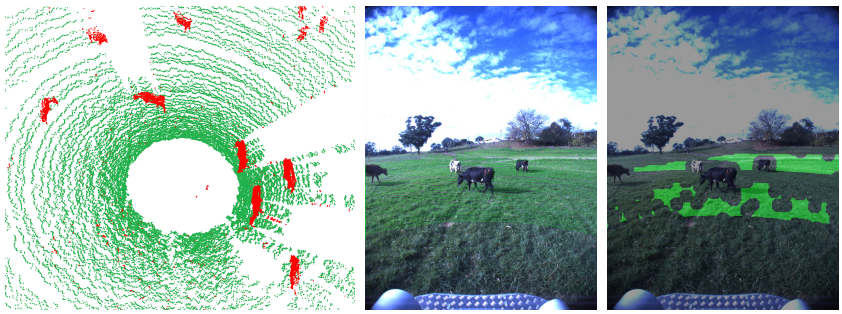
mechanism on a single monocular camera (Lookingbill et al., 2007). In agricultural contexts, the near-to-far approach has been applied by Milella et al. (2015) using radar and camera in rural environments and by Zhou et al. (2012) using lidar and camera in forested terrain.

In this work, a 64-beam lidar is used to continuously supervise a visual classifier with online learning for traversability assessment in an agricultural grass field environment. Although the lidar itself is capable of detecting far-distance objects, its vertical resolution and field of view is smaller than that of the camera. The use of a multi-beam lidar further increases reliability and allows for an evaluation of how the number of lasers affects the accuracy.

Figure 8.1a illustrates the proposed approach. A simple geometric classifier detects flat, traversable ground areas (green rays in the figure). The traversable areas are projected from 3D onto a corresponding 2D image from a camera and used as training data to update a visual model of normal ground appearance. Figure 8.1b illustrates such as visual model of normal ground appearance. The visual classifier applies the model on the entire image to detect non-traversable image patches as outliers from the model. The desired output (ground truth) is illustrated in Figure 8.1c as an overlay of traversable pixels.

8.1 3D Ground Segmentation

A ground segmentation algorithm for lidar point clouds by Douillard et al. (2011) is used to extract 3D points of traversable areas in front of a vehicle. Figure 8.2a shows the output from the segmentation as ground (green) and non-ground (red) 3D points,



(a) (b) (c)

Figure 8.2: 3D ground segmentation and projection to 2D images. (a) ground and non-ground 3D points. (c) 3D ground points projected on 2D image. (c) resulting pixel areas used for supervising the visual classifier.

whereas Figure 8.2c shows the projection of the ground points onto the 2D camera image using a perspective transformation. A morphological dilation connects the ground pixels to form traversable pixel areas. This is followed by a morphological erosion with a slightly larger structuring element in order to prevent potential calibration or synchronization inaccuracies from overlaying non-traversable areas. Figure 8.2c shows the resulting traversable ground points used for supervising the visual classifier.

8.2 Visual Classifier

A Gaussian mixture model (GMM) is used for modeling and maintaining the normal, visual appearance of traversable ground. Each cluster in the model describes a ground component (e.g. grass or dirt) and is represented by a mean vector and a covariance matrix. Figure 8.1b shows an example of a two-dimensional GMM in 2D with three components ($K = 3$). These are estimated with the Expectation Maximization (EM) algorithm (Dempster et al., 1977) by clustering all the training data provided by the lidar (Figure 8.2c).

As visual features, we use rg chromaticity, Gray-Level Co-Occurrence Matrix (GLCM) features (energy, homogeneity, and contrast) (Haralick et al., 1973), as well as densely extracted SIFT features on multiple scales (Lowe, 2004). For each frame, all ground pixels are added to the GMM by maintaining a constant sized sliding window of all supervised ground points within the past 20 frames. The visual classifier is then applied to all pixels in the image by calculating the Mahalanobis distance between their feature vectors and the K different components in the GMM. If the minimum distance of these is beyond a certain threshold, the pixel is considered to be non-traversable. The method can thus be seen as an anomaly detector, as it models only normal ground appearance and thus detects anomalies as outliers.

(a) Frame 1 (b) Frame 40 (c) Frame 100 (d) Frame 256 (e) Frame 340

Figure 8.3: Examples of online visual ground segmentation from a 2 minute traversal with overlaid ground model inliers colored by their closest Gaussian components.

8.3 Results and Discussion

The method has been evaluated on the Australian dairy dataset AUS5 (page 32), which includes a grass field with cows and humans. 374 randomly sampled image patches (one for each frame) have been manually annotated with per-pixel labels for a 2 minute traversal on the field. A 64-beam Velodyne HDL-64E lidar provided point clouds used for supervising the visual classifier.

Figure 8.3 illustrates examples of how the visual classifier detected ground pixels. The pixels are overlaid with colors corresponding to which of the $K = 3$ GMM components had the smallest Mahalanobis distance. Distances above 12 standard deviations were considered non-traversable. The decision boundary was chosen to maximize the accuracy on the same dataset.

Figure 8.4 shows a graph of the classification accuracy over a 2 minute traversal on the field. Results were averaged with a sliding window of 25 frames to reduce noise introduced by the annotation of randomly sampled image patches. The figure compares three strategies for the learning process. With online learning, the visual classifier was continuously updated using supervised ground points from a sliding window of 20 frames. With offline learning, the visual classifier was trained on the first 20 frames only. And with batch training, the visual classifier was trained on ground points from all 374 frames. The graphs show that initially, online and offline learning performed equally well as they shared the first 20 frames of training. However, as online learning was able to continuously update its visual model, it generally performed better thereafter. Especially after a 180° platform turn from frame 130-160, rapid changes in illumination caused the performance of offline learning to degrade, whereas online learning was able to adapt after a while. Batch training performed somewhere in between. With a mean accuracy well below online learning, the batch-trained model clearly did not incorporate the necessary variation in appearance and illumination.

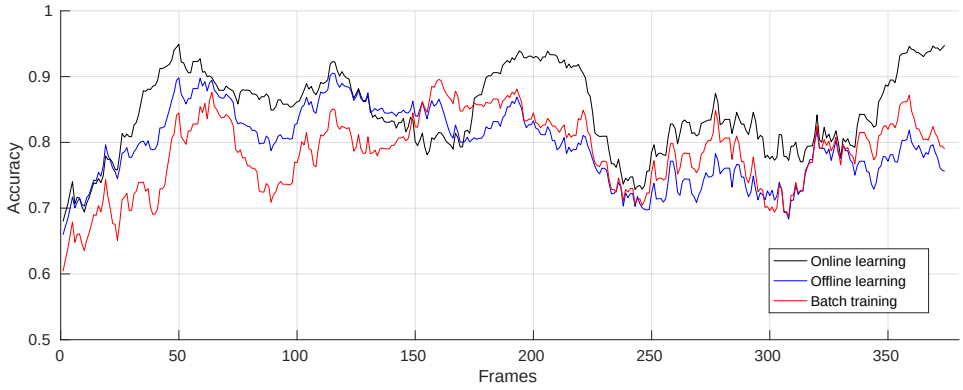


Figure 8.4: Accuracy over time with three learning strategies.

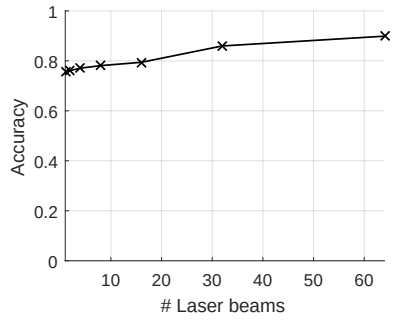


Figure 8.5: Mean accuracy of online learning as function of the number of laser beams used for supervision.

Figure 8.5 shows an evaluation of the number of lasers utilized from the 64-beam supervising lidar. The accuracy was averaged over the entire traversal using the online learning strategy. As expected, using all 64 beams resulted in the best overall performance of 90% accuracy. Halving the number of beams to 32 gave a slightly worse performance of 86%, while the use of 16 lasers gave 79%. The worst performance was achieved using a single laser resulting in 76% accuracy. The relative difference can simply be caused by the availability of more training data with more lasers. However, it can also be caused by the presence of more view points, effectively training the visual classifier at different scales. This could be addressed in a similar way to proprioceptive self-supervised learning (Kim et al., 2006). For a single laser, image features of previously seen areas could thus be memorized and introduced into the model once the same areas were seen by the laser.

In the above examples, only $K = 3$ GMM components were used. This was partly for visualization purposes and partly for providing comparable results. Larger values of K would increase the capacity of the models and thus considerably improve non-adaptable models. However, for the batch training presented above with identical training and test sets, it would also mean an unfair comparison. To exemplify, using $K = 20$ GMM

components results in mean accuracies of 84%, 76%, and 89% for online learning, offline learning, and batch training. Therefore, batch training can not generally be considered worse than online learning. If the training set includes all relevant conditions of appearance and illumination, and the model is capable of containing and using the information, batch training may provide equivalent or better results with potentially even faster adaptation.

9 Lidar-Camera Fusion with Conditional Random Fields

The content of this chapter partly appears in the following publication:

> Paper 5: *Kragh and Underwood (2017). Multi-modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. Submitted to International Journal of Robotics Research, February 2017.*

In this chapter, appearance- and geometry-based detection methods are combined probabilistically with lidar and camera sensors using a conditional random field (CRF). A visual classifier provides information of visually distinctive areas, whereas a geometric classifier discriminates ground from non-ground and characterizes structures by their point distributions. The proposed method performs semantic segmentation in 2D and 3D simultaneously and fuses predictions to ensure consistent labels both spatially, temporally, and across modalities. The method is evaluated on a diverse agricultural dataset, comparing single- vs. multi-modality performance and domain adaptation vs. domain training.

Figure 9.1 illustrates the proposed fusion algorithm. Individual 2D and 3D classifiers are first trained to provide initial pixel- and point-wise class probability estimates. For each modality, these are then clustered into 2D and 3D segments in order to limit the number of nodes in the subsequent CRF graph. 3D segments are projected onto the 2D image, and a CRF is trained to jointly infer optimal class labels by fusing the two modalities. Finally, temporal links are added between subsequent frames, thus smoothing predictions both spatially, temporally, and across modalities.

The following two sections describe the initial 2D and 3D classifiers individually. This is followed by a detailed description of the proposed CRF graph including its unary and pairwise potentials.

9.1 2D Classifier

The visual classifier represents a pipeline of three steps: segmentation, feature extraction, and classification. The image is first segmented into superpixels using SLIC (Achanta

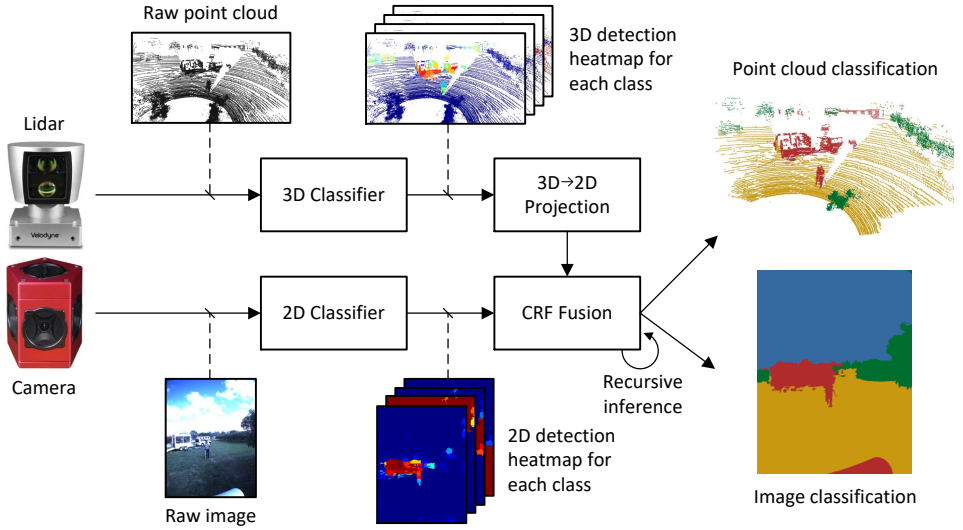


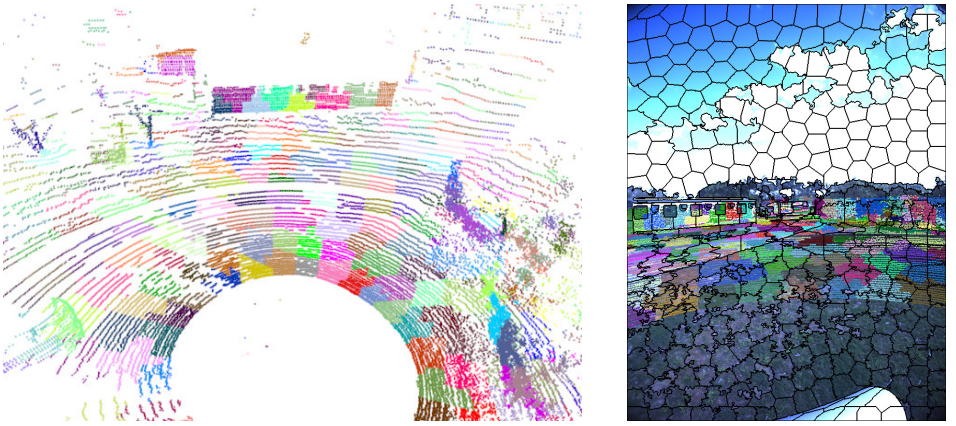
Figure 9.1: Schematic overview of proposed fusion algorithm. Reprinted from Kragh and Underwood (2017).

et al., 2012), and edges between neighboring segments are established. A number of features are then extracted from each superpixel. These include average RGB values, GLCM features (energy, homogeneity and contrast) (Haralick et al., 1973), and a bag-of-words histogram of densely extracted SIFT features (Lowe, 2004). The concatenated feature vector is normalized by subtracting the mean and dividing by the standard deviation across the entire training set. Finally, an SVM is used to classify each superpixel and assign probability estimates that sum to one across all classes (Chang and Lin, 2011). For superpixel i, this provides a probability estimate $P_{\text{initial}}\left(x_i^{\text{2D}} \mid \mathbf{z}_i^{\text{2D}}\right)$ of class label x_i^{2D} given the features $\mathbf{z}_i^{\text{2D}}$.

Paper 5 (Kragh and Underwood, 2017) compares the above "traditional" computer vision pipeline with a recent deep learning approach using a CNN for semantic segmentation (Long et al., 2015). However, as the traditional approach provided better results when fused with lidar, the deep learning approach and its results are omitted in this summary.

9.2 3D Classifier

The geometric classifier represents a similar pipeline to the visual classifier, although in a different order: feature extraction, classification, and segmentation. Point-wise features are first extracted using the method proposed in Paper 4 (Kragh et al., 2015) and described in chapter 5. That is, a number of hand-crafted features describing local point distributions are calculated for each point using an adaptive, distance-dependent neighborhood radius. In this work, however, only 9 of the proposed features are used: f_1-f_4 (height), f_5-f_7 (shape), f_{11} (orientation), and f_{13} (reflectance) (see Table 5.1 for



(a) (b)

Figure 9.2: Segmentation in 2D and 3D. (a) Supervoxels in 3D. (b) Superpixels in 2D overlaid with projected 3D supervoxels. Adapted from Kragh and Underwood (2017).

more information). An SVM classifier similar to the one used in 2D is used to produce class probabilities for each point.

After classification, a clustering procedure is used to segment the point cloud into a number of 3D supervoxels. The method uses the approach proposed by Papon et al. (2013), however, with a modified feature distance measure D between two segments:

$$D = \lambda D_s + \chi^2. \tag{9.1}$$

D_s denotes the spatial Euclidean distance between the segments, whereas χ^2 denotes the Chi-Squared histogram distance (Pele and Werman, 2010) between their mean histograms of probability estimates. $\lambda > 0$ is a weighting factor. During clustering, points are grouped together such that D in Equation 9.1 is minimized. The supervoxels thus consist of neighboring points with similar initial probability estimates. Finally, neighboring supervoxels are connected with edges, and the average of all point probabilities within each segment is computed. For supervoxel i, this provides a probability estimate $P_{\text{initial}}\left(x_i^{\text{3D}} \mid \mathbf{z}_i^{\text{3D}}\right)$ of class label x_i^{3D} given the features $\mathbf{z}_i^{\text{3D}}$.

In order to fuse the two modalities, the 3D superpixels are projected onto the corresponding image using a perspective projection defined from the extrinsic and intrinsic parameters of the lidar and the camera. Figure 9.2a shows an example of a point cloud segmented into supervoxels, whereas Figure 9.2b shows the projection of supervoxels onto a superpixel-segmented image. The overlap of 3D segments and 2D segments in image space defines the connections between the two modalities. As is clear from the figure, a single 3D segment can map to multiple 2D segments and vice versa.

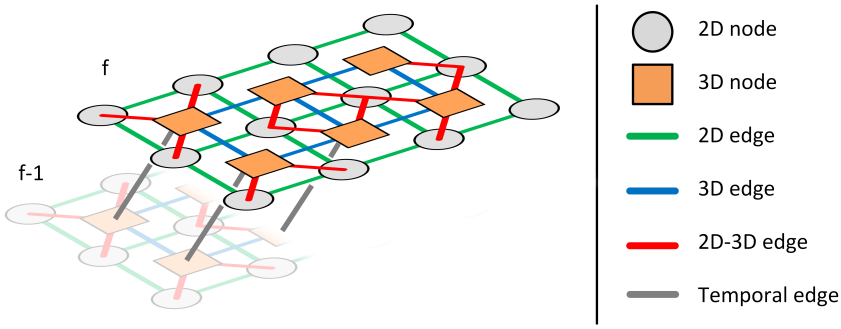


Figure 9.3: CRF graph with 2D nodes (superpixels), 3D nodes (supervoxels), and edges between them both spatially and temporally. Reprinted from Kragh and Underwood (2017).

9.3 Conditional Random Field

A CRF is trained to infer optimal class labels for both modalities simultaneously. Figure 9.3 illustrates an undirected graphical model defining the different nodes and edges available. Each 2D segment (superpixel) and 3D segment (supervoxel) is assigned a node in the graph. 2D edges are derived from neighboring 2D segments in image space, 3D edges from neighboring 3D segments in 3D space, and 2D-3D edges from overlapping 2D segments and projected 3D segments. Additional temporal edges link 3D segments between subsequent frames.

A CRF models the conditional probability distribution $p(\mathbf{x} \mid \mathbf{z})$ with $\mathbf{x}$ representing class labels of all nodes and $\mathbf{z}$ representing observations. In this work, the observations are the class probability estimates of the initial 2D and 3D classifiers. The conditional distribution $p(\mathbf{x} \mid \mathbf{z})$ can be expressed as:

$$p(\mathbf{x} \mid \mathbf{z}) = \frac{1}{Z(\mathbf{z})} exp(-E(\mathbf{x} \mid \mathbf{z})), \tag{9.2}$$

where $Z(\mathbf{z})$ is the partition (normalization) function and $E(\mathbf{x} \mid \mathbf{z})$ is the Gibbs energy. The Gibbs energy function of a pairwise CRF with the graph structure in Figure 9.3 can be written as:

$$E(\mathbf{x} \mid \mathbf{z}) = \sum_{i=1}^{N^{2D}} \phi_i^{2D} + \sum_{i=1}^{N^{3D}} \phi_i^{3D} + \sum_{i,j \in E^{2D}} \psi_{ij}^{2D} + \sum_{i,j \in E^{3D}} \psi_{ij}^{3D} \tag{9.3}$$

$$+ \sum_{i,j \in E^{2D\text{-}3D}} \psi_{ij}^{2D\text{-}3D} + \sum_{i,j \in E^{Time}} \psi_{ij}^{Time}. \tag{9.4}$$

Here, ϕ_i^{2D} and ϕ_i^{3D} are unary potentials, whereas ψ_{ij}^{2D}, ψ_{ij}^{3D}, $\psi_{ij}^{2D\text{-}3D}$, and ψ_{ij}^{Time} are pairwise potentials. N^{2D} and N^{3D} denote the number of 2D and 3D nodes, and E^{2D}, E^{3D}, $E^{2D\text{-}3D}$, and E^{Time} denote edges between nodes. For simplicity, function variables and

weights for both unary and pairwise potentials are omitted from the equation, but explained in more detail below.

Unary Potentials

The unary potentials incorporate the prior class probabilities of the initial 2D and 3D classifiers into the model. They are defined as the negative logarithm of the class probabilities, such that Equation 9.2 simplifies to initial probability distributions if no pairwise potentials are present:

$$\phi_i^{2D}\left(x_i^{2D}, \mathbf{z}_i^{2D}\right) = -log\left(P_{\text{initial}}\left(x_i^{2D} \mid \mathbf{z}_i^{2D}\right)\right), \tag{9.5}$$

$$\phi_i^{3D}\left(x_i^{3D}, \mathbf{z}_i^{3D}\right) = -log\left(P_{\text{initial}}\left(x_i^{3D} \mid \mathbf{z}_i^{3D}\right)\right). \tag{9.6}$$

Here, x_i^{2D} and x_i^{3D} are the class labels and $\mathbf{z}_i^{2D}$ and $\mathbf{z}_i^{3D}$ are the 2D and 3D features described in section 9.1 and 9.2 above. The unary potentials define the cost for assigning label x to 2D or 3D node i. A low cost is inferred if the initial probability is close to 1, whereas a high cost is inferred if the initial probability is close to 0. As class imbalance is handled by the initial classifiers, no CRF weights are introduced for unary potentials.

Pairwise Potentials

Four types of pairwise potentials exist: 2D edges linking superpixels spatially, 3D edges linking supervoxels spatially, 2D-3D edges linking overlapping superpixels and supervoxels, and recursive edges linking supervoxels temporally.

2D and 3D edges help smooth predictions spatially in both the image and the point cloud. The potentials define cost functions that depend on exponentiated distances between neighboring 2D or 3D nodes. That is, a small distance (in some space) between two nodes results in a high cost of assigning them with different labels, and vice versa. Inspired by (Boykov and Jolly, 2001; Krähenbühl and Koltun, 2012), 2D edges use a distance in RGB-space:

$$\psi_{ij}^{2D}\left(x_i^{2D}, x_j^{2D}, \mathbf{z}_i^{2D}, \mathbf{z}_j^{2D}\right) = w_p^{2D}\left(x_i^{2D}, x_j^{2D}\right)\delta\left(x_i^{2D} \neq x_j^{2D}\right)\exp\left(-\frac{\left|I_i - I_j\right|^2}{2\sigma_{2D}^2}\right). \tag{9.7}$$

Here, I_i is a vector of mean RGB-values for 2D segment i, σ_{2D} is a normalization factor trained with cross-validation, and $\mathbf{w}_\mathbf{p}^{2D}$ is a weight matrix. The Dirac delta function δ ensures that no cost is inferred for assigning neighboring nodes the same label. The symmetric weight matrix $\mathbf{w}_\mathbf{p}^{2D}$ is trained with the CRF and allows for different interactions between classes. For instance, adjacent *ground* and *object* pixels may be common (low weight), whereas adjacent *ground* and *sky* pixels may be less common (high weight), as they are usually separated by some sort of vegetation.

3D edges use a distance between local plane normals (Hermans et al., 2014; Namin et al., 2015):

$$\psi_{ij}^{\text{3D}}\left(x_i^{\text{3D}}, x_j^{\text{3D}}, \mathbf{z}_i^{\text{3D}}, \mathbf{z}_j^{\text{3D}}\right) = w_p^{\text{3D}}\left(x_i^{\text{3D}}, x_j^{\text{3D}}\right)\delta\left(x_i^{\text{3D}} \neq x_j^{\text{3D}}\right)\exp\left(-\frac{\left|\theta_i - \theta_j\right|^2}{2\sigma_{\text{3D}}^2}\right). \tag{9.8}$$

Here, θ_i is the angular difference between the z-axis and the local plane normal for 3D segment i, calculated as $\theta = cos^{-1}\left(f_{11}\right)$. And similar to 2D, σ_{3D} is a normalization factor trained with cross-validation, and $\mathbf{w}_{\mathbf{p}}^{\text{3D}}$ is a symmetric and class-dependent weight matrix trained with the CRF.

2D-3D edges connect 2D and 3D segments through a perspective projection. The potential defines a cost for assigning overlapping regions from the two modalities with different labels. As suggested by Namin et al. (2015), the cost depends on the area of overlap, which can be defined as:

$$\omega\left(S_i^{\text{2D}}, S_j^{\text{3D}}\right) = \left|S_i^{\text{2D}} \cap S_j^{\text{3D}\rightarrow\text{2D}}\right|. \tag{9.9}$$

Here, S_i^{2D} denotes the set of pixels in 2D segment i, S_j^{3D} denotes the set of points in 3D segment j, and $S_j^{\text{3D}\rightarrow\text{2D}}$ denotes the set of pixels intersected by the projection of 3D segment j onto the image. The weight in Equation 9.9 thus describes the cardinality (number of elements) of the intersection of the two segments. The pairwise potential normalizes this weight by the maximum weight across all 2D segments overlapped by the projected 3D segment j:

$$\psi_{ij}^{\text{2D-3D}}\left(x_i^{\text{2D}}, x_j^{\text{3D}}, \mathbf{z}_i^{\text{2D}}, \mathbf{z}_j^{\text{3D}}\right) = w_p^{\text{2D-3D}}\left(x_i^{\text{2D}}, x_j^{\text{3D}}\right)\delta\left(x_i^{\text{2D}} \neq x_j^{\text{3D}}\right)\frac{\omega\left(S_i^{\text{2D}}, S_j^{\text{3D}}\right)}{\max\limits_{k\in E_j^{\text{2D-3D}}}\omega\left(S_k^{\text{2D}}, S_j^{\text{3D}}\right)}. \tag{9.10}$$

Here, k denotes a 2D segment in the set of all edges $E_j^{\text{2D-3D}}$ that connect the projection of 3D segment j with 2D segments. With this potential, the cost for assigning different labels for connected 2D and 3D segments depends on their area of overlap, such that a large overlap results in a high cost and vice versa. The weight matrix $\mathbf{w}_{\mathbf{p}}^{\text{2D-3D}}$ is class-dependent and asymmetric, such that both class label and sensor technology can affect the interactions between nodes.

Recursive edges link 3D nodes from the current frame f_c to a previous frame f_p. The localization system of the robot is used to transform all 3D segments from frame f_p into the world frame, and from there into the current frame f_c. Here, they will likely represent the same structures from different view points and should thus be assigned the same class labels. The pairwise potential defines a cost that depends on the Euclidean distance

between a 3D node i in frame f_c and a transformed 3D node j in frame f_p:

$$\psi_{ij}^{\text{Time}}\left(x_{i,f_c}^{\text{3D}}, x_{j,f_p}^{\text{3D}}, \mathbf{p}_{i,f_c}^{\text{3D}}, \mathbf{p}_{j,f_p}^{\text{3D}}\right) = w_p^{\text{Time}}\left(x_{i,f_c}^{\text{3D}}, x_{j,f_p}^{\text{3D}}\right)\delta\left(x_{i,f_c}^{\text{3D}} \neq x_{j,f_p}^{\text{3D}}\right) \cdot \exp\left(-\frac{\overline{\text{diag}\left(\Sigma_{\text{Nav}}\right)}}{2\sigma_{\text{Nav}}^2}\right)\exp\left(-\frac{\left|\mathbf{p}_{i,f_c}^{\text{3D}} - T_{f_p}^{f_c}\left(\mathbf{p}_{j,f_p}^{\text{3D}}\right)\right|^2}{2\sigma_{\text{Time}}^2}\right). \tag{9.11}$$

Here, x_{i,f_c}^{3D} and x_{j,f_p}^{3D} denote the labels of 3D nodes i and j in frame f_c and f_p, respectively. $\mathbf{p}_{i,f_c}^{\text{3D}}$ and $\mathbf{p}_{j,f_p}^{\text{3D}}$ are the mean 3D coordinates of supervoxel i in frame f_c and supervoxel j in frame f_p. $T_{f_p}^{f_c}$ is the transformation from frame f_p to f_c, and $\overline{\text{diag}\left(\Sigma_{\text{Nav}}\right)}$ is the mean localization variance (position and orientation) reported by the localization system and averaged between frame f_p and f_c. Finally, σ_{Nav} and σ_{Time} are normalization factors trained with cross-validation, whereas $\mathbf{w}_{\mathbf{p}}^{\text{Time}}$ is a class-dependent, symmetric weight matrix trained with the CRF. With the recursive inference, 3D nodes from subsequent frames are linked, and the predictions are effectively smoothed spatially across frames. As 2D nodes are linked to 3D nodes in each frame, visual information is indirectly forwarded across frames. The use of localization variance in Equation 9.11 ensures that a cost is only inferred when the localization can be trusted.

Training and Inference

In the above subsections, a number of weight matrices were introduced. These are combined to a single weight vector $\mathbf{w} = \left[\mathbf{w}_p^{\text{2D}}, \mathbf{w}_p^{\text{3D}}, \mathbf{w}_p^{\text{2D-3D}}, \mathbf{w}_p^{\text{Time}}\right]$, which is extended with bias weights for all pairwise potentials. The weights are trained offline using maximum likelihood estimation. However, as the graph is cyclic, exact inference is unattainable, and loopy belief propagation is therefore used for approximate inference. For recursive edges, ground truth labels should ideally be available for both frame f_c and f_p. However, to limit the manual annotation work, only current frames f_c were annotated. All 2D and 3D nodes from frame f_p were instead given unknown labels (unobserved in the graph) and marginalized out during maximum likelihood estimation.

During inference at test time, a decoding procedure minimizes the energy $E(\mathbf{x} \mid \mathbf{z})$ using loopy belief propagation. That is, the most likely sequence of class labels $\mathbf{x}$ for all 2D and 3D nodes is estimated given all measurements $\mathbf{z}$.

9.4 Results and Discussion

The method has been evaluated on the Australian datasets AUS1-5 (page 32) that comprise a dairy paddock and various types of orchards. The datasets all include synchronized frames of Velodyne HDL-64E lidar data and Ladybug panospheric camera images along with accurate IMU- and GPS-based localization data. Pixel- and point-wise manual annotations are available for 120 frames.

In the following, the method is evaluated on multi-class classification with four classes $x_i = \{ground, sky, vegetation, object\}$. Due to the physics of the lidar, the *sky* class was only observed in the images. 5-fold cross-validation was applied, corresponding to the 5 individual datasets from different locations and environments. The results thus represent a multi-source domain adaptation approach, in which training data from multiple source domains (e.g. various orchards) were used to generalize to a different, unseen target domain (e.g. grass field).

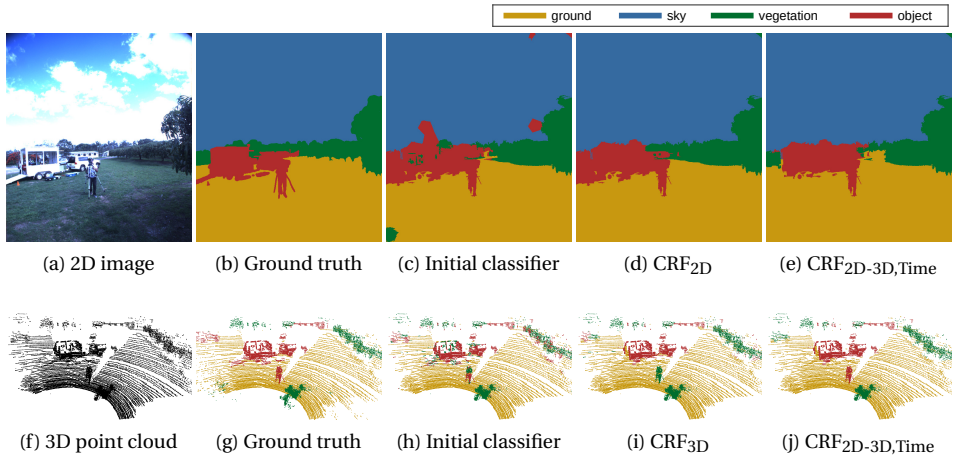
Table 9.1 presents the results across all datasets for 2D and 3D performance after applying the CRF and enabling progressively more pairwise potentials. The results are evaluated per-pixel in 2D and per-point in 3D in terms of intersection over union (IoU) and accuracy. *Initial* refers to the performance of the single-modal initial SVM classifier, whereas CRF_{2D} and CRF_{3D} refer to single-modal "smoothed" outputs using only 2D and 3D edges, respectively. $CRF_{2D\text{-}3D}$ refers to the performance after sensor fusion (multi-modal edges), while $CRF_{2D\text{-}3D,Time}$ refers to the final performance after adding recursive inference (temporal edges).

Table 9.1: Classification results for 2D and 3D. Reprinted from Kragh and Underwood (2017).

	IoU					accuracy
	ground	sky	vegetation	object	mean	
2D, Initial	0.847	0.933	0.729	0.233	0.685	0.900
2D, CRF_{2D}	0.893	**0.971**	0.763	0.342	0.742	0.937
2D, $CRF_{2D\text{-}3D}$	**0.907**	**0.971**	0.774	0.372	0.756	**0.943**
2D, $CRF_{2D\text{-}3D,Time}$	**0.907**	**0.971**	**0.775**	**0.379**	**0.758**	**0.943**
3D, Initial	**0.936**	-	0.735	0.365	0.678	0.881
3D, CRF_{3D}	0.933	-	0.846	0.466	0.748	0.923
3D, $CRF_{2D\text{-}3D}$	0.929	-	0.886	0.667	0.827	0.943
3D, $CRF_{2D\text{-}3D,Time}$	0.933	-	**0.897**	**0.697**	**0.842**	**0.948**

The table generally shows a gradual improvement as more terms in the CRF are enabled. A high performance increase was introduced with single-modal 2D and 3D edges in the CRF. This is commonly experienced for local feature extraction methods such as the ones presented here. Deep learning methods that use hierarchical feature representations generally experience a smaller improvement with spatial smoothing as shown in Paper 5. Introducing multi-modal links further increased performance, although most remarkably in 3D. The same applies to recursive inference which increased mean IoU in 3D with 1.5%, but only 0.2% in 2D.

Figure 9.4 shows a qualitative example of the gradual improvements. The initial 2D and 3D classifications in (c) and (h) show a number of misclassifications in both 2D and 3D. Some of these were handled by single-modal CRF smoothing in (d) and (i), with (i) showing significant improvements in label consistency. Finally, multi-modal fusion



(a) 2D image (b) Ground truth (c) Initial classifier (d) CRF_{2D} (e) $CRF_{2D\text{-}3D,Time}$

(f) 3D point cloud (g) Ground truth (h) Initial classifier (i) CRF_{3D} (j) $CRF_{2D\text{-}3D,Time}$

Figure 9.4: Example results. The two rows show 2D and 3D results, respectively. Reprinted from Kragh and Underwood (2017).

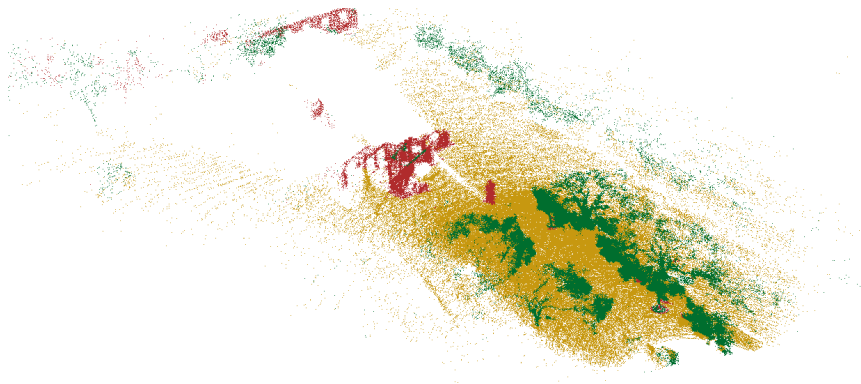
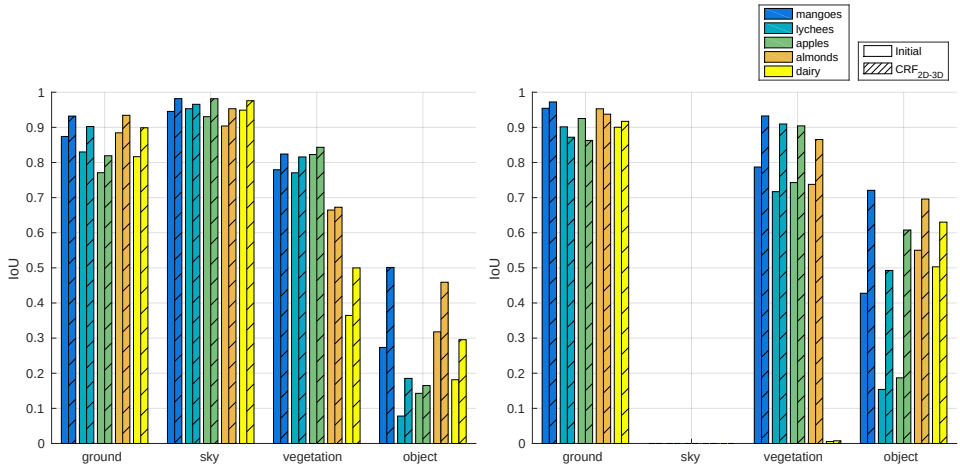


Figure 9.5: Example of accumulated classification results in 3D of a trajectory. Reprinted from Kragh and Underwood (2017).

and recursive inference helped correct misclassified *ground* and *vegetation* pixels in 2D around the trailer as seen in (e). And in 3D, a person in the front of the scene that was previously mistaken for *vegetation* was corrected to the *object* class in (j).

Figure 9.5 shows an example of a traversal along the end of an orchard row, in which the predicted 3D point clouds were accumulated over time. As subsequent predictions at the same physical location could be corrected over time, the most recent prediction within a radius of 0.5 m was used as label for each point.



(a) 2D (b) 3D

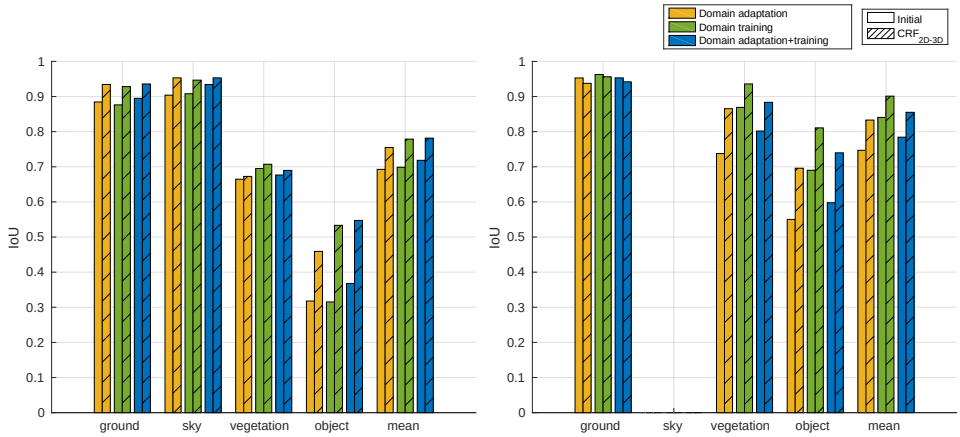
Figure 9.6: Classification results across object classes and datasets before and after sensor fusion. Reprinted from Kragh and Underwood (2017).

9.4.1 Domain Adaptation

The results presented in Table 9.1 were averaged across all datasets. However, as the datasets represent different domains and environments, features and classifiers transferred from other sources may not generalize equally well. Figure 9.6 therefore presents the class-specific 2D and 3D performances for the 5 datasets, individually. Filled bars denote results from initial classifiers, whereas hatched bars denote results after sensor fusion ($CRF_{2D\text{-}3D}$). Figure 9.6a shows that in 2D, the *ground* and *sky* classes transferred well, whereas the *object* class experienced considerable fluctuations, possibly due to a larger variation in object appearances. The *vegetation* class transferred rather well among orchards, but did not seem to generalize to the dairy dataset (grass field). Figure 9.6b shows that in 3D, *ground* transferred well, while *vegetation* transferred quite well among orchards, but not to the dairy dataset. Similar to 2D, the *object* class experienced considerable fluctuations, but transferred slightly better than 2D for most datasets. The hatched bars in Figure 9.6 show improvements for nearly all classes and datasets. The most significant improvements were seen for the *vegetation* class in 3D and the *object* class in both 2D and 3D.

9.4.2 Domain Training

Domain adaptation, as presented above, transfers features and classifiers trained on a source domain to a target domain. Unless the source and target distributions are identical, this will in general provide inferior results compared to models trained and tested on the same data distribution. In this subsection, domain adaptation is compared to domain training.



(a) 2D (b) 3D

Figure 9.7: Domain adaptation vs. domain training on the *almonds* dataset. Domain training includes training data from *almonds* only, whereas domain adaptation+training additionally includes training data from other domains. Reprinted from Kragh and Underwood (2017).

The AUS4 dataset (page 32) consists of recordings from an almonds orchard from two separate days. By splitting this dataset into two, domain training was evaluated. Figure 9.7 shows a comparison between three training strategies all tested on the *almonds* dataset. The first strategy, domain adaptation, refers to the exact same training strategy as described in the previous section. That is, the algorithms were trained on data from all domains except *almonds*. To ease comparison, these results are replicated from Figure 9.6. The second strategy, domain training, refers to a scenario, in which training data was available only from the *almonds* dataset. This was accomplished using 2-fold cross-validation across the two days of recordings in the *almonds* dataset. The third strategy, domain adaptation+training, refers to a combination of the first two. Again, 2-fold cross-validation was used across the two days of recordings, however with the addition of training data from all other domains. The graphs show relatively small variations between the three strategies in 2D. In 3D, however, domain training performs significantly better than domain adaptation on both the *vegetation* and *object* classes. Domain adaptation+training provided the best results in 2D, whereas in 3D, it gave worse results than domain training. This shows that domain adaptation can also deteriorate performance when the source and target data distributions are dissimilar. Classes with large inter-domain variation such as *object*, however, appeared to benefit from domain-specific training data, as the extended data examples helped span the space of possible object appearances and geometries.

10 Multi-Modal Semantic Segmentation in 3D with Range Images

The content of this chapter partly appears in the following draft of a journal paper:

> Paper 9: *Kragh et al. (2018). Multi-Modal Semantic Segmentation in 3D with Range Images. Draft, February 2018.*

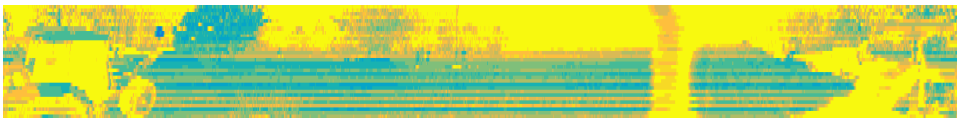
In this chapter, an extension of the point-wise classification approach with deep learning from chapter 6 is presented. The extension fuses lidar sensing with stereo and thermal cameras by projecting 3D lidar points onto the 2D images, thus appending lidar range images with color and temperature channels. The method is evaluated on the FieldSAFE dataset DK6 (page 27) which includes calibrated and synchronized frames from the three modalities.

10.1 Color and Temperature Channels

The method is a direct extension of the approach presented in chapter 6. It thus applies a state-of-the-art FCN on 2D range images to infer pixel-wise class labels. These can then easily be converted back to point clouds, thus providing semantic segmentation in 3D.

To extend the range images with color and temperature channels, 3D points from a single lidar scan were projected onto each of the images using known static transformations between the sensors. The transformations were found with the calibration procedure described in section 2.2. As all three sensors were synchronized in hardware, a unique correspondence between the points clouds and the images was guaranteed. Figure 10.1 illustrates an example of a range image with additional color and temperature channels. The color image provides three additional channels, each represented as unsigned 8-bit integers. The temperature image provides a single channel with unsigned 16-bit integers ranging from $-273\,^{\circ}\text{C}$ to $2348\,^{\circ}\text{C}$ with a resolution of $0.04\,^{\circ}\text{C}$. Since it is intended for obstacle detection only, the temperature intensities are scaled linearly in the interval $0\,^{\circ}\text{C}$ to $50\,^{\circ}\text{C}$.

(a) Range (pseudo-colored)



(b) Intensity (pseudo-colored)

(c) Color channel

(d) Temperature channel (pseudo-colored)

Figure 10.1: Channel examples represented in range image format. Adapted from Kragh et al. (2018).

As input to the neural network, the 6 channels are all converted to floats and normalized to the range $[0, 1]$. More details on the normalization process are available in Paper 9.

The same network and training strategies as in chapter 6 are used for semantic segmentation on the 2D range images. The network structure, however, deviates slightly as the number of input channels goes from 2 to 6. Furthermore, instead of randomly sampling 256×256 pixel crops from the entire range image, random crops are extracted in a limited horizontal range to ensure non-zero color and temperature channels.

10.2 Results and Discussion

As in chapter 6, the method was evaluated on the FieldSAFE dataset DK6 (page 27) with the same training, validation, and test splits.

Table 10.1 lists preliminary class-wise results for the test set as more range image channels were added. The table reports class-wise intersection over union (IoU) as well as the overall classification accuracy. Clearly, fusion with the color and thermal cameras did generally not improve classification results. In fact, adding color and temperature channels resulted in worse scores for both the *grass*, *vegetation*, and *road* classes. Only the *human*, *building*, and *object* classes were increased by a very small margin.

The discouraging results may have originated from two separate issues. Lidar-camera calibration inaccuracies could result in misaligned range images which would make it difficult if not impossible for the network to benefit from additional modalities. Another possible reason may be that the network overfitted to the training set, once more modalities were introduced. For instance, although the mannequin dolls in the dataset were geometrically similar, their visual appearances were remarkably different due to their individual clothes. When providing the network with color channels, it may have learned the exact appearance of the mannequin dolls in the training set. If so, this would not generalize to the test set, and multi-modal fusion would therefore decrease performance rather than increasing it. Moreover, due to the use of static mannequin dolls instead of real humans, the fusion with thermal camera did not help recognize humans based on their normal temperatures.

As the presented results are only preliminary, future work should focus on optimizing inter-sensor calibration to minimize misalignments. Furthermore, separate and larger datasets should be used for training and testing, with manual annotations for the test set. This would provide fair and reliable results and allow for an evaluation of how well the trained network generalizes to new environments.

Table 10.1: Class-wise classification results as more range image channels are added. Results for range and intensity channels are copied from Table 6.1 for comparison. Adapted from Kragh et al. (2018).

	IoU							accuracy
	grass	vegetation	human	road	building	object	mean	
Range	0.979	0.888	0.000	0.246	0.000	0.014	0.355	0.975
Range, intensity	**0.985**	**0.905**	0.020	**0.545**	0.000	0.103	**0.426**	**0.981**
Range, intensity, color	0.977	0.878	0.000	0.202	0.000	0.009	0.344	0.972
Range, intensity, thermal	0.980	0.878	**0.023**	0.426	**0.005**	0.074	0.398	0.976
Range, intensity, color, thermal	0.976	0.879	0.000	0.140	**0.005**	**0.113**	0.352	0.972

11 Concluding Remarks

In this part of the study, three separate methods were proposed for fusing 3D point clouds from a lidar with 2D images from color and thermal cameras. The first method presented a self-supervised classification system using a lidar for continuously supervising an online visual classifier of traversability. By only modeling normal ground appearance instead of all possible obstacle appearances, the method was used to detect anomalies. It was shown that online learning continuously adapted to changes in illumination and ground appearance. The method is directly applicable in new and unseen environments and does not require manual annotations.

The second method presented point- and pixel-wise high-level fusion of a lidar and a color camera using a conditional random field. Spatially and temporally consistent labels were inferred by including relationships between neighboring 2D pixels and 3D points as well as overlapping regions between the two modalities. It was shown that the introduction of spatial, multi-modal, and temporal relationships gave gradual improvements in classification performance. Evaluating domain adaptation further showed that features and classifiers that were trained on mango, lychees and apple orchards generalized well to unseen almond orchards in both 2D and 3D.

The third method presented an approach for fusing a lidar with thermal and color images on low-level using a convolutional neural network. The method extended the range image representation proposed in section 6 with additional color and temperature channels. Preliminary results showed that no improvements were seen when adding visual and thermal cues to the network. This may be due to calibration inaccuracies that caused inter-modality misalignment errors. Future work should therefore focus on ensuring optimal alignment and investigate other approaches for neural network fusion. Recent work by Chen et al. (2017) suggests hierarchical fusion of parallel subnetworks for each modality, which would enable both low- and high-level fusion while allowing the camera-based subnetwork to benefit from pre-trained models.

All three methods rely on accurate calibration, registration, and synchronization between the involved sensors. These parameters determine how well the data are aligned, and thus to a large extent how much sensor fusion can contribute. In the presented work, all methods were evaluated in their ability to improve classification performance. Multi-modal fusion can also help mitigate single points of failure by introducing redundancy with complementary views. In this study, however, these aspects were not evaluated.

Obstacle Mapping Part V

In Part III and IV, obstacle detection and classification was performed in local sensor frames. That is, images and point clouds were classified pixel- and point-wise without subsequent treatment of these in a robotics context. However, for an autonomous system to be safe, obstacle detection must be followed by obstacle avoidance. This includes, among other steps, a transformation of detections from local sensor frames to the vehicle frame, possibly followed by local or global mapping.

Occupancy grid mapping is widely used to generate obstacle maps from potentially noisy detections (Elfes, 1989). Using probabilistic inverse sensor models, occupancy grid maps (OGMs) are capable of fusing detections from multiple sensors probabilistically and recursively. They have been used for 2D mapping in numerous robotic applications and have shown great performance in high-level obstacle fusion (Thrun et al., 2002; Colleens and Colleens, 2007). Other variants focus on real-time aspects by only mapping obstacles locally (Jörg, 1995), or perform motion planning on local histogram grids representing obstacle detections (Borenstein and Koren, 1991). Occupancy grid mapping assumes known vehicle poses provided by a localization system. In this study, only the mapping problem is addressed, although simultaneous localization and mapping (SLAM) methods exist for solving the localization and mapping tasks concurrently.

In agriculture, global 2D object mapping has been proposed for mapping detected plants with a lidar (Weiss and Biber, 2011), and for mapping olive stems using a combination of lidar and stereo (Cheein et al., 2011). Similarly, a number of methods exist that map detected obstacles globally in 2D. Moorehead et al. (2012) detect and map obstacles in an orchard by maintaining three separate obstacle maps from 1) prior knowledge, 2) lidar detections, and 3) camera detections. Instead of fusing the three representations, a human operator is alerted if any of the three views report obstacles in front of the vehicle. Reina et al. (2016b) apply probabilistic fusion on lidar and stereo vision to generate 2D traversability maps. Other approaches use 2D obstacle or occupancy maps for autonomous control. Emmi et al. (2014) maintain a global occupancy map between a fleet of robots that contains robot positions and lidar-detected obstacles. The occupancy grid map is used to start and stop vehicles to avoid collisions on their predefined paths, similar to the method proposed by Fleischmann and Berns (2016). Rovira-Más and Reid (2004) use an A* path planner (Hart et al., 1968) on a stereo vision-based density grid, Thrun et al. (2006) apply local path planning on lidar-generated occupancy maps, whereas Ball et al. (2016) use global path planning on a 2D costmap generated with stereo vision. To represent obstacle positions accurately, all the above approaches use either Differential or RTK GPS combined with odometry from e.g. IMUs and wheel encoders.

In this part of the study, localization and mapping of obstacle detections from multiple modalities is addressed. In the following chapter, an architecture is presented for fusing multi-modal obstacle detections with a semantical occupancy grid map, thus representing different object categories concurrently. The maps include both static and dynamic obstacles and are used for extracting process-relevant information along the traversed trajectory of an agricultural vehicle.

12 Obstacle Detection and Mapping for Process Evaluation

The content of this chapter partly appears in the following three publications:

> Paper 6: *Kragh et al. (2016b). Multi-modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. Conference presentation at the International Conference on Agricultural Engineering (CIGR), June 2016.*
>
> Paper 7: *Korthals et al. (2018). Multi-Modal Detection and Mapping of Static and Dynamic Obstacles in Agriculture for Process Evaluation. Frontiers in Robotics and AI, Research Topic: Multi-modal Sensor Fusion, March 2018.*
>
> Paper 8: *Korthals et al. (2017b). Towards Inverse Sensor Mapping in Agriculture. Conference presentation at the International Conference on Intelligent Robots and Systems (IROS), Workshop on "Agricultural Robotics: learning from Industry 4.0 and moving into the future", September 2017.*

In this chapter, multi-modal detections of static and dynamic obstacles are localized and mapped globally using an OGM representation. Detection methods for stereo camera, thermal camera, radar, and lidar are fused both spatially and temporally using a common 2D grid map representation in their local sensor frames. Finally, properties relevant for processing an agricultural field such as traversability and yield information are extracted along planned vehicle trajectories. The proposed method is evaluated on a multi-modal obstacle detection dataset with ground truth annotations in global GPS coordinates.

Figure 12.1 illustrates the proposed architecture. A sensor platform captures multi-modal perception data. Exteroceptive sensors are used for detecting obstacles in the environment, whereas proprioceptive sensors are used for global localization. For each exteroceptive sensor, an inverse sensor model (ISM) performs obstacle detection in the local sensor frame and transforms detections into a local bird's-eye view 2D grid map with class-specific occupancy probabilities. A fusion and mapping step localizes ISMs globally and fuses them across sensor modalities, detection algorithms, and object classes using a semantical occupancy grid map (SOGM) representation. Finally, the planned trajectory of the vehicle is decoded such that process-relevant parameters are extracted.

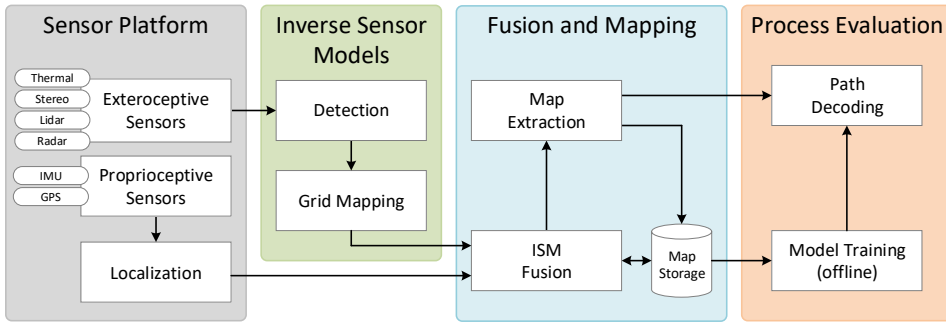


Figure 12.1: System architecture including information flow. Reprinted from Korthals et al. (2018).

The sensor platform was described in detail in section 2.1. The following subsections describe each of the remain three steps of the architecture: inverse sensor models, fusion and mapping, and process evaluation.

12.1 Inverse Sensor Models

The exteroceptive sensors include stereo camera, thermal camera, lidar, and radar. For each sensor, one or more ISMs are introduced, providing class-specific 2D OGMs in the local sensor frame.

For the **stereo camera**, a number of state-of-the-art methods for object detection and semantic segmentation on images are applied. The pedestrian detector LDCF by Nam et al. (2014) uses locally decorrelated channel features to detect bounding boxes of humans with fixed aspect ratios. The object detector YOLO by Redmon and Farhadi (2016) uses deep learning to detect bounding boxes of a variety of object classes. In this work, these are remapped to one of the classes *human*, *object*, or *unknown* as shown in Figure 12.2a. The anomaly detector DeepAnomaly by Christiansen et al. (2016a) uses features from a CNN to detect outliers from a model trained on normal appearance of agricultural environments. An example frame with overlaid anomaly detections (in red) is shown in Figure 12.2b. Finally, the FCN algorithm by Long et al. (2015) performs semantic segmentation with a fully convolutional neural network on 59 classes. In this work, these are remapped to one of the classes *human*, *object*, *grass*, *ground*, *vegetation*, and *undefined* as shown in Figure 12.2c.

For the **thermal camera**, the HeatDetection algorithm by Christiansen et al. (2014) is used to threshold pixels 3.0 °C above the median temperature of the bottom 80% part of image. An example frame with overlaid detections is shown in Figure 12.2d.

As the above image-based detection algorithms all operate in pixel-space, a transformation is required to provide 2D OGMs. For the object detection algorithms, the depth

image from the stereo camera is used to transform from local pixel coordinates to metric coordinates. For semantic segmentation and heat detection, inverse perspective mapping is used to approximate the transformation, assuming zero-height objects and a flat ground plane (Bertozzi and Broggi, 1998; Konrad et al., 2012). The global transformation is illustrated for the color camera in Figure 12.2 (e) and (f).

The applied **radar** outputs a preprocessed list of 32 targets for each frame. The targets are output in range and azimuth coordinates and can thus directly be mapped to a 2D grid representation. However, as the targets mostly represent noise, a tracking algorithm (Munkres, 1957) is used to filter out non-consistent samples temporally. Figure 12.3a illustrates noisy targets (red) and confirmed targets (green) overlaid on the lidar point cloud. A pseudo detection probability is estimated based on the track length of the target. The resulting OGM is shown in Figure 12.3b.

For the **lidar**, the point-wise classification method proposed in Paper 4 (Kragh et al., 2015) and described in chapter 5 is used. The classifier provides probabilities for each of the classes *ground*, *vegetation*, and *object* using an SVM as exemplified in Figure 12.3c. In order to provide 2D OGMs, the *ground* class is incorporated into the *vegetation* and *object* classes with Bayesian fusion. Figure 12.3d shows an example of the resulting OGM for the *object* class.

12.2 Fusion and Mapping

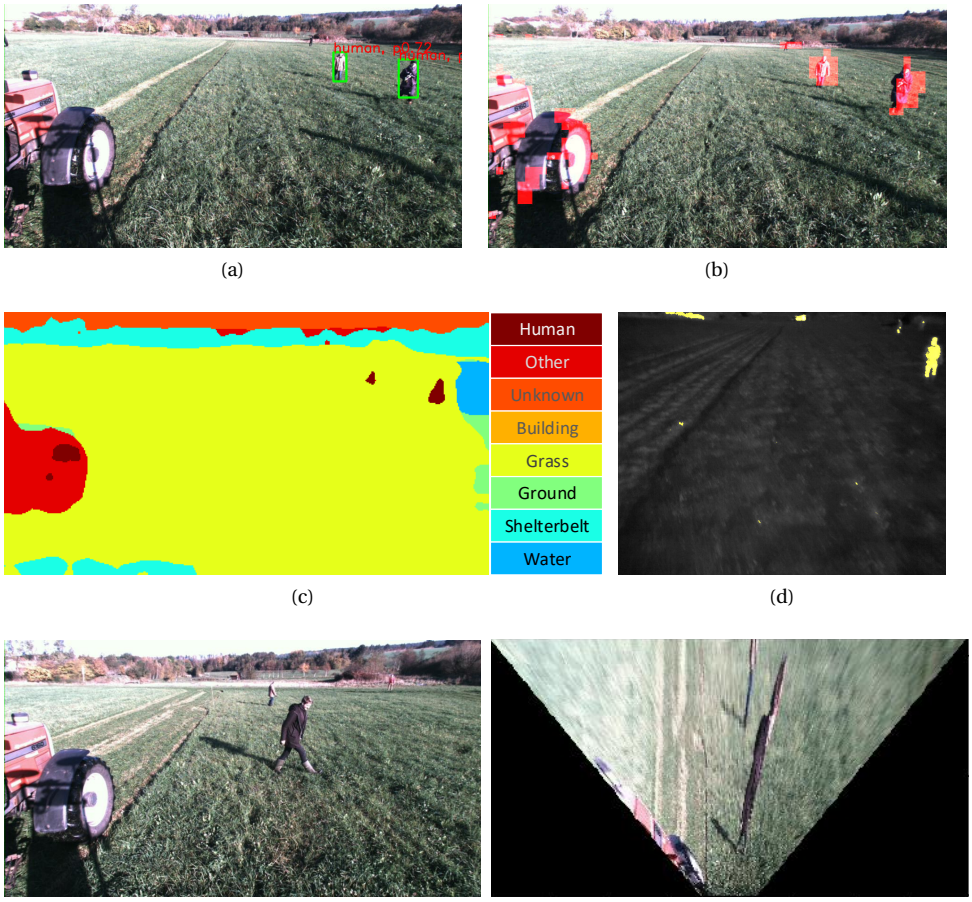
Occupancy grid maps are often used to generate and update static obstacle maps while traversing unknown areas (Thrun et al., 2005; Stachniss, 2009). The most common representation is a 2D OGM with probabilities of occupancy. A cell probability of 0 represents unoccupied space, whereas a cell probability of 1 represents occupied space. An OGM is typically initialized with 0.5 probabilities representing unknown states of occupancy.

An OGM M consists of a number of cells $m \in M$. Over time t, the map is updated to incorporate sensor measurements $z_{1:t} = z_1, \ldots, z_t$ in the local vehicle frame and vehicle poses $x_{1:t} = x_1, \ldots, x_t$. The vehicle poses are obtained with the robot_localization package (Moore and Stouch, 2014) in ROS by concatenating the world referenced position and orientations from the GPS and IMU sensors. The overall goal is to estimate the posterior distribution $P(M \mid z_{1:t}, x_{1:t})$. Using a Bayesian update rule (Hähnel, 2004), this can be done recursively for each cell m:

$$P(m \mid z_{1:t}, x_{1:t}) = \frac{P(m \mid z_t, x_t) \cdot P(z_t \mid x_t) \cdot P(m \mid x_{1:t-1}, z_{1:t-1})}{P(M) \cdot P(z_t \mid x_{1:t}, z_{1:t-1})} \tag{12.1}$$

Here, $P(m \mid z_t, x_t)$ is provided directly by the ISM from each sensor. By further exploiting that $P(\neg m) = 1 - P(m)$ and using the notation $\text{Odds}(x) = \frac{P(x)}{1-P(x)}$, we get:

$$\log\text{Odds}(m \mid x_{1:t}, z_{1:t}) = \log\text{Odds}(m \mid z_t, x_t) + \log\text{Odds}(m \mid x_{1:t-1}, z_{1:t-1}) \tag{12.2}$$



(a) (b)

(c) (d)

(e) (f)

Figure 12.2: Camera detections for stereo and thermal camera. (a) Object detection using YOLO. (b) Anomaly detections (highlighted with red) using DeepAnomaly. (c) Semantic segmentation using FCN. (d) Thermal camera detections (highlighted with yellow) using HeatDetection. (f) Raw unwarped image. (f) Inverse perspective mapping. Adapted from Korthals et al. (2018).

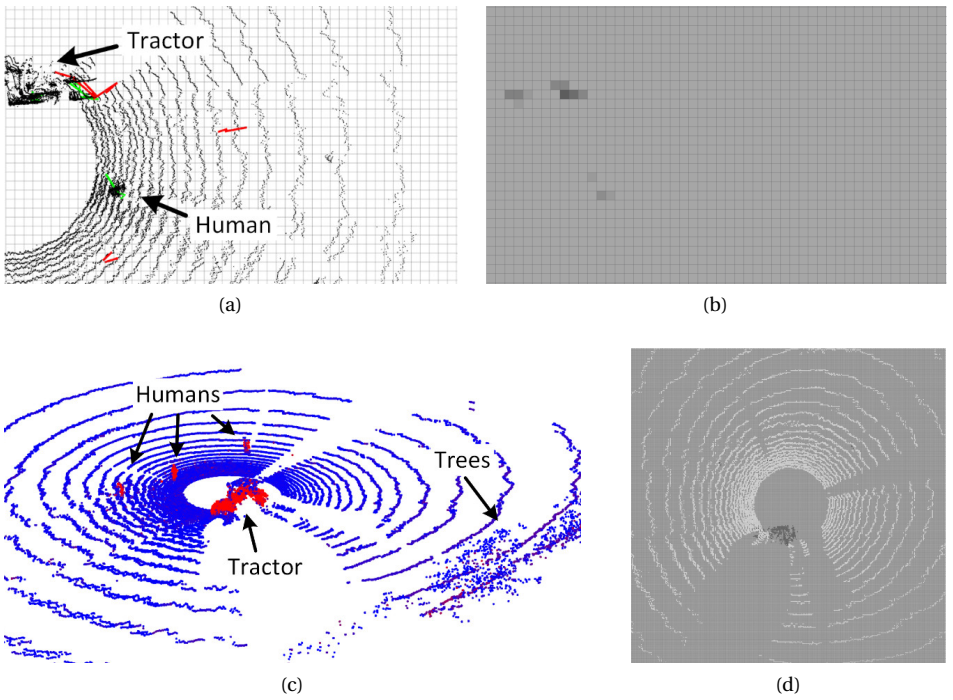


Figure 12.3: Lidar and radar detections and OGMs. (a) Radar detection example with confirmed (green) and unconfirmed (red) radar tracks overlaid on point cloud. (b) Resulting radar OGM. (c) Point cloud with pseudo-colored probability estimates of the *object* class. Blue and red denote low and high probabilities, respectively. (d) Resulting lidar OGM for the *object* class illustrating low (bright) and high (dark) probabilities. Adapted from Korthals et al. (2018).

The update formula is recursive and does not require increasing memory or computation as more measurements are introduced.

In this work, the binary OGM presented above is extended to a semantical occupancy grid map (SOGM) representation proposed by Korthals et al. (2017a). Here, N semantical map layers corresponding to the different object classes are maintained such that $m \in [0,1]^N$. Figure 12.4 illustrates the SOGM framework. Each sensor has its own SOGM and uses the Bayesian update formula in Equation 12.2 to update the individual layers over time. During evaluation, information is fused across layers and sensors using one of two strategies: Superbayesian Independent Opinion Pooling P_{B} (Pathak et al., 2007) and non-Bayesian maximum pooling P_{M}:

$$P_{\mathrm{B}}(m) = \frac{1}{1 + \prod_n \frac{1-P(m_n)}{P(m_n)}} \tag{12.3}$$

$$P_{\mathrm{M}}(m) = \max_n P(m_n) \tag{12.4}$$

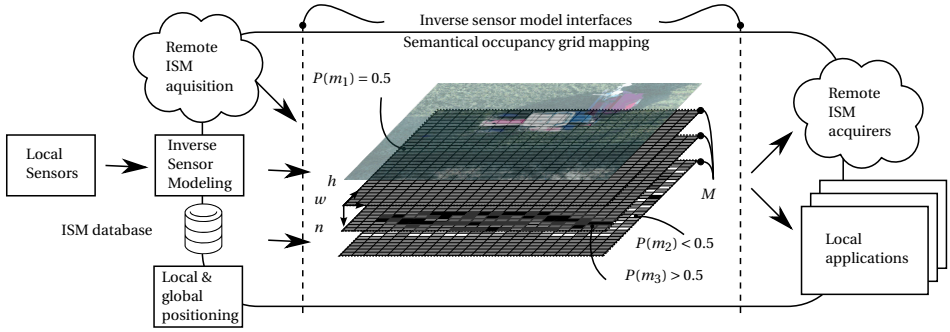


Figure 12.4: SOGM framework. Reprinted from Korthals et al. (2018).

where m_n denotes the n'th layer. The two methods represent competitive and complementary fusion strategies, respectively, and are used for evaluating binary traversability and class-specific obstacle mapping.

As the mapping and fusion methods assume a static environment, a concept known as recency weighting (Biber, 2005) is introduced to handle dynamic (moving) obstacles. A *ForgetValue* parameter defines how much of the previous information in the map is forgotten during an update. A value of 0 indicates no forgetting, corresponding to a static obstacle mapping approach, whereas a value of 1 effectively clears the map before each update. A *ForgetRate* parameter defines the update interval. Together, the two parameters introduce measurement decay over time, allowing the positions of moving obstacles to be updated continuously.

12.3 Process Evaluation

When traversing an agricultural environment during field operation, a number of parameters may influence the optimum actions of a vehicle. In addition to safety in terms of obstacle avoidance, process-relevant parameters include features such as traversability, processability, and crop quality. These parameters may control whether the vehicle should simply traverse an area or also perform an agricultural task such as mowing, spraying, or sowing while doing so.

The proposed method continuously queries the SOGM and decodes predictions along the planned trajectory of the vehicle. A hidden Markov model (HMM) for each property is used to model dependencies between classes both spatially and temporally, thus estimating the joint probability $P(\mathcal{O}, w; \lambda)$ for each step along the trajectory. $\mathcal{O}$ denotes observations extracted from the SOGM along the vehicle trajectory, w denotes one of the specific properties *traversable*, *non-traversable*, *processable*, and *moving obstacle*, and λ denotes a generative property model (another HMM) for $\mathcal{O}$. The observations $\mathcal{O}$ are extracted from the SOGM using a cell-clustering technique called Supercell Extracted Variance Driven Sampling (SEVDS) (Korthals et al., 2017a). This reduces the number of

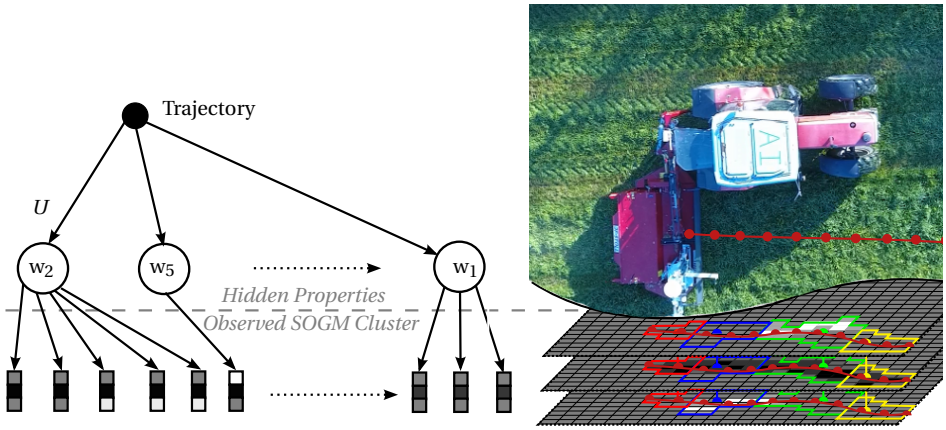


Figure 12.5: Conceptual representation of the proposed framework with the generative sampling on the left and a corresponding scenario with observations along the red tractor trajectory on the right. Reprinted from Korthals et al. (2018).

inputs and handles potential positional offsets between layers by clustering grid cells with similar predictions. The model and training procedures are further described in Paper 7 (Korthals et al., 2018).

Figure 12.5 illustrates the proposed framework conceptually. A given trajectory is evaluated by decoding hidden properties along it using observed SOGM clusters. For each step, the property w with the most likely model λ_w is found using the Viterbi algorithm (Rabiner, 1989).

12.4 Results and Discussion

The framework has been evaluated on the FieldSAFE dataset DK6 (page 27) that includes both static and dynamic obstacles in a grass field with ground truth annotations in global GPS coordinates. The dataset was recorded with the most recent version of the SuperSensorKit and therefore includes stereo camera, thermal camera, webcam, 360° camera, lidar, radar, IMU, and GPS. In this evaluation, however, only the stereo camera, thermal camera, lidar, and radar are used for exteroceptive perception, whereas fused IMU and GPS provide accurate localization.

Three evaluation scenarios were carried out, all with a grid size in the global map of 10 cm. A static scenario evaluated detection and mapping of static obstacles on both binary traversability assessment and class-specific classification. A dynamic scenario evaluated detection and mapping of moving obstacles by disregarding all static obstacles. Finally, a process evaluation scenario evaluated the proposed approach to decode process-relevant parameters along the trajectory of the vehicle.

For all scenarios, the number of true positives (TP), false positives (FP), and false negatives (FN) were calculated on cell-level across the entire map. From these, precision, recall, and F_1 scores (harmonic mean of precision and recall) were derived along with the entropy H describing information gain. More details on the metrics and how the evaluation was carried out specifically are available in Paper 7 (Korthals et al., 2018). In the following, each of the three scenarios is presented individually.

12.4.1 Static Obstacles

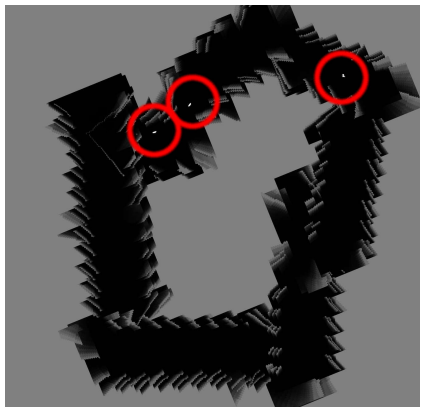
The static evaluation was split into two, such that the ability to map both class-specific obstacles and traversable areas were evaluated separately.

In evaluation **A**, four process-relevant classes were defined: *vulnerable obstacles*, *processable*, *traversable*, and *non-traversable*. Table 12.1 lists the relationships to individual ISMs along with detection results before and after fusion. Data fusion was performed first among classifiers within each sensor, and then across sensors. Results are presented for both competitive Bayesian fusion (Equation 12.3) and complementary max-pooling (Equation 12.4). Figure 12.6 shows four example obstacle maps from the same evaluation. (a) illustrates human detections by the YOLO algorithm, whereas (b) shows the Bayesian fusion across human obstacle maps generated by YOLO, LDCF, and FCN. Where YOLO managed to detect three mannequin dolls, the fused map includes detections of all four mannequin dolls. (c) and (d) show vegetation detections, first by the lidar alone, and then after max-pooling fusion with FCN semantic segmentation. FCN helped increase the recall and F_1 scores slightly, although most structures were captured accurately by the lidar itself.

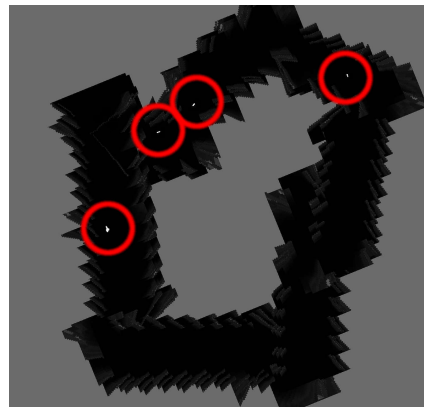
In evaluation **B**, binary traversability assessment was evaluated. In order to disregard unobserved areas, a third *unknown* class was added on top of *occupied* and *unoccupied*. Table 12.2 lists the detection results before and after fusion, with classifiers grouped by their object categories. In the second column, classifiers within each group were fused with competitive Bayesian fusion, as it increased precision while maintaining entropy. In the third column, complementary max-pooling fusion was applied across groups, as it increased recall while maintaining precision. As is clear from the table, the lidar itself provided rather accurate traversability predictions. However, the best results were obtained after fusing all algorithms and sensors. Figure 12.7 shows qualitative results for the two fusion steps. (a) illustrates the outcome of competitive Bayesian fusion of the first group of classifiers, all detecting humans. (b) shows the subsequent outcome of complementary max-pooling fusion across the three groups, effectively fusing information of all non-traversable structures.

Table 12.1: Evaluation **A**. Process-relevant object detection for single classifiers, classifier combinations, and sensor combinations. Vertical lines encapsulate groups of algorithms. Reprinted from Korthals et al. (2018).

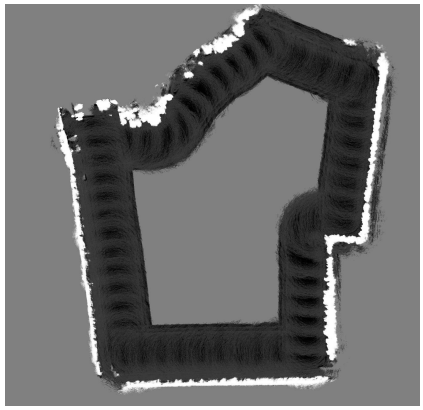
	Single classifiers				Fusion among classifiers					Fusion among sensors				
Classifier	F_1	Prec.	Rec.	H	Fus.	F_1	Prec.	Rec.	H	Fus.	F_1	Prec.	Rec.	H
Vulnerable Obstacles														
cam-LDCF-human	1.3	0.7	25.9	83.2	max.	3.2	1.6	73.4	86.2					
cam-FCN-human	3.4	1.7	73.6	75.6	bay.	12.6	7.1	57.4	84.3					
cam-YOLO-human	11.7	6.9	36.1	75.5										
Processable														
cam-FCN-grass	85.2	94.2	77.8	75.2										
Traversable														
cam-FCN-grass	83.4	96.3	73.6	75.2	max.	84.6	96.0	75.6	75.3	max.	90.1	89.2	91.0	92.3
cam-FCN-ground	24.0	96.8	13.7	75.1	bay.	82.0	97.2	71.0	75.2	bay.	87.7	90.8	84.8	92.2
lidar-SVM-ground	89.7	89.4	90.1	81.1										
Non-Traversable														
lidar-SVM-veg.	83.6	81.4	86.0	87.9						max.	84.3	80.1	89.1	92.3
cam-FCN-veg.	46.6	32.2	84.7	81.2						bay.	84.8	81.3	88.7	92.3



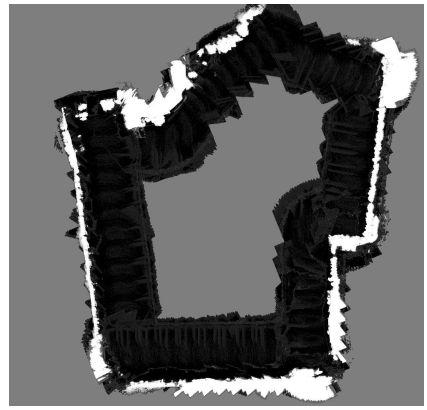
(a) cam-YOLO-human



(b) Bayesian fusion



(c) lidar-SVM-veg

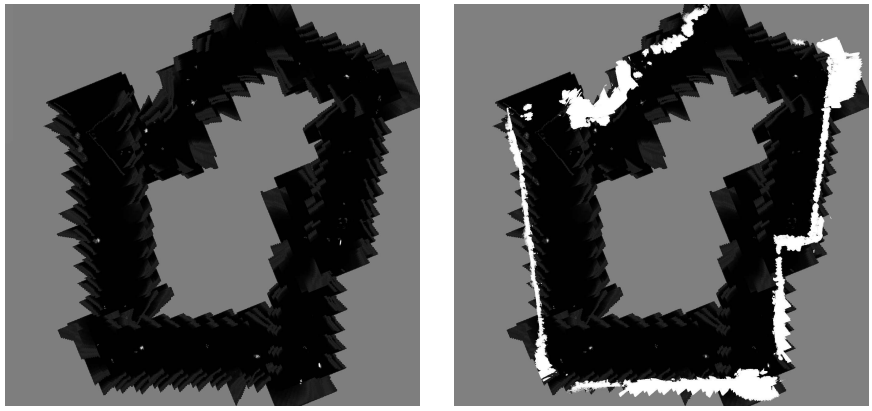


(d) Max-pooling fusion

Figure 12.6: Example obstacle maps from static evaluation **A**. Intensities indicate occupancy probabilities with white being occupied, black unoccupied, and gray unknown. Red circles highlight static mannequin dolls in (a) and (b). Adapted from Korthals et al. (2018).

Table 12.2: Evaluation **B**. Traversability assessment of static obstacles for single classifiers, classifier combinations, and sensor combinations. Vertical lines encapsulate groups of algorithms. Reprinted from Korthals et al. (2018).

	Single classifiers				Bayesian among classifiers				Max-pooling among groups			
Classifier	F_1	Prec.	Rec.	H	F_1	Prec.	Rec.	H	F_1	Prec.	Rec.	H
cam-FCN-human	3.8	25.3	2.1	75.6								
cam-LDCF-human	0.7	3.7	0.4	83.2								
cam-YOLO-human	1.2	6.8	0.7	75.5	13.0	67.4	7.2	89.2				
radar-tracking	2.6	3.5	2.1	15.9								
thermal-HeatDetection	7.3	16.6	4.7	88.6								
lidar-SVM-object	7.8	66.8	4.1	89.7								
cam-FCN-object	4.1	30.8	2.2	76.3					88.8	88.3	89.4	92.5
cam-YOLO-object	2.0	3.9	1.3	75.6								
cam-DeepAnomaly	2.0	3.8	1.4	75.6	22.3	72.3	13.2	89.5				
radar-tracking	2.6	3.5	2.1	15.9								
lidar-SVM-object	7.8	66.8	4.1	89.7								
lidar-SVM-veg.	83.5	81.4	85.8	87.9	84.6	88.3	81.6	92.3				
cam-FCN-veg.	46.7	32.2	84.4	81.2								



(a) Bayesian fusion among 1st classifier group (b) Max-pooling fusion across groups

Figure 12.7: Example obstacle maps from static evaluation **B**. Intensities indicate occupancy probabilities with white being occupied, black unoccupied, and gray unknown. Adapted from Korthals et al. (2018).

12.4.2 Dynamic Obstacles

When evaluating the detection of dynamic obstacles, recency weighting as introduced in section 12.2 was applied during map updates to disregard all static obstacles. As described in section 2.4, ground truth annotations for dynamic obstacles in the DK6 dataset were point-wise and not pixel-wise as for the static scenario. Therefore, cell-wise evaluation was infeasible. Instead, algorithm predictions were clustered, and TP, FP, and FN figures were calculated by comparing detected clusters with ground truth points for each timestamp as described in Paper 7 (Korthals et al., 2018).

Table 12.3 defines two sensor/algorithm setups that were evaluated individually. Setup 1 includes all sensors and algorithms, whereas setup 2 includes stereo camera-based algorithms only. Table 12.4 presents detection and mapping results of moving obstacles with the two setups. The best results were obtained by fusing all sensors in setup 1 with complementary max-pooling fusion. Complementary fusion thus surpassed competitive fusion, possibly due to non-overlapping detections from different sensors caused by calibration and localization errors. In setup 2, however, competitive Bayesian fusion was superior to complementary fusion, since all included algorithms used the same camera and thus were guaranteed to overlap after mapping. Figure 12.8 shows a qualitative example of a single frame from the dynamic evaluation in which two humans in front of the tractor were detected and mapped. As some of the humans are positioned either behind, next to, or far from the vehicle, only detections and ground truth positions inside the sensors' field of view were included in the evaluation.

Table 12.3: Listing of setups and included detection algorithms. Adapted from Korthals et al. (2018).

Class	object	heat	object	objects/human	human	human	anomaly
Algorithm	detection	DynamicHeat	SVM	FCN	LDCF	YOLO	DeepAnomaly
Setup	1			2			

Table 12.4: Sensor fusion of setup 1 and 2 with different fusion strategies. Reprinted from Korthals et al. (2018).

Setup	Fusion	F_1 (%)	Precision (%)	Recall (%)
1	max.	70.81	57.23	92.86
	bay.	42.58	39.76	45.83
2	max.	57.32	51.14	65.22
	bay.	61.22	56.96	66.18

12.4.3 Process Evaluation

The process evaluation method proposed in section 12.3 was evaluated on four process-relevant classes: *traversable*, *non-traversable*, *processable*, and *moving obstacle*. The evaluation was carried out in two range intervals, as not all sensors had the same range capabilities. A near field interval included all obstacles within 12.5 m in front of the tractor, whereas a far field interval included all obstacles from 12.5 m to 25 m in front of the tractor. The evaluation was conducted along the trajectory of the vehicle corresponding to three laps around the field. Figure 12.9 shows the two confusion matrices for near field and far field process evaluation. The evaluation showed that the best results were obtained at near field and that the performance degraded with distance. At near field, the accuracy for all classes was above 90%. When comparing with the static evaluation

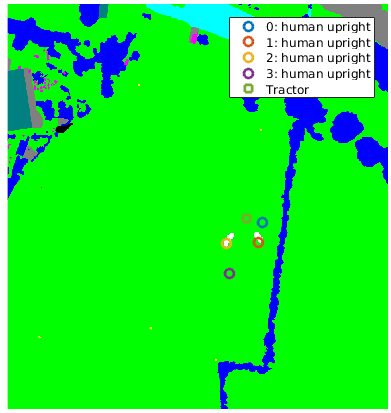
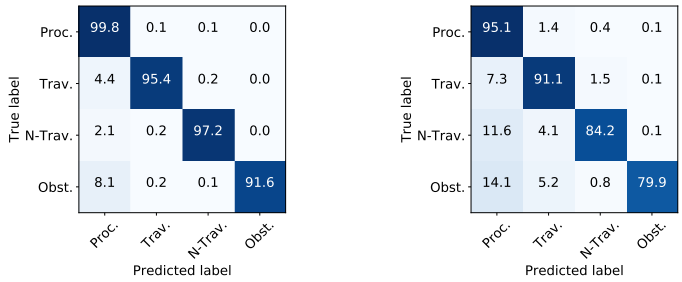


Figure 12.8: Example from dynamic evaluation with human detections overlaid on annotated map. Colored circles indicate ground truth positions of human obstacles and the tractor. Reprinted from Korthals et al. (2018).

A, this indicates that the HMM was able to learn temporal and spatial relationships between classes, thus increasing the overall classification performance.



(a) Near field (b) Far field

Figure 12.9: Confusion matrices for near and far field process evaluation. Reprinted from Korthals et al. (2018).

13 Concluding Remarks

In this part of the study, a method was proposed for high-level fusion and global mapping of object detections from multiple modalities. Semantical occupancy grid mapping in 2D was used to probabilistically fuse detections from a stereo camera, thermal camera, radar, and lidar both spatially and temporally. Inverse sensor models were presented for each modality based on state-of-the-art detection algorithms, and two fusion methods were proposed for fusing different modalities and different object layers. Complementary fusion was used across different sensors where calibration and localization errors could cause misalignments, whereas competitive fusion was used across inverse sensor models from the same sensor. Finally, process-relevant properties were extracted along the vehicle path with an HMM to simulate an actual traversal of a field including obstacles.

Results showed a gradual improvement in classification accuracy of globally mapped detections as more sensors and inverse sensor models were introduced. Recency weighting was successfully introduced in the occupancy grid mapping to handle moving objects, and evaluations were carried out on both static and dynamic obstacle scenarios. Evaluating actual traversals, four process-relevant properties were extracted with accuracies above 90% at near field and at or above 80% at far field.

The evaluation was conducted with cell-wise comparison between mapped predictions and ground truth GPS annotations. This was useful for comparing methods and for illustrating relative improvements with sensor fusion. The map-based evaluation, however, did not describe the actual safety of such a system. The process evaluation method addressed the issue of safety by runtime classification of properties in front of the vehicle. However, actual use cases for agricultural machines including operating speeds and braking distances were not addressed.

Future work on object mapping should apply end-to-end supervised training of 2D-mapped object detections from each sensor. This could help reduce misalignments and potentially learn relationships between different information sources.

Part VI Discussion and Conclusion

14 Discussion

In agriculture, only a few industrial projects investigate fully autonomous vehicles for monitoring and processing broad-acre and orchard environments. In the automotive industry, on the other hand, all major car manufacturers invest enormous amounts on self-driven technologies. However, automating agricultural vehicles and robots has a huge potential for reducing manual labor and optimizing yield. Scientific research on obstacle detection in agriculture has focused primarily on single-modality systems that use simple and traditional methods for assessing traversability. In this thesis, however, obstacles were detected and recognized, and multiple sensing modalities were investigated for increasing classification performance.

In this section, the main contributions and results from each part of the thesis are first summarized and discussed individually. The combined work is then discussed and related to the end-goal of fully autonomous agricultural vehicles.

In **part II**, data material necessary for developing and evaluating obstacle detection methods was presented. Two multi-modal research perception platforms were described along with datasets acquired in a wide range of agricultural environments. One of the datasets, FieldSAFE, was made publicly available. The FieldSAFE dataset can facilitate future research on agricultural obstacle detection with multiple sensing modalities. A semi-automated procedure was proposed and used for annotating both static and dynamic obstacles from GPS-referenced drone footage. With large-scale annotations, the dataset further allows for training state-of-the-art deep neural networks as exemplified in chapter 6 and 10. The annotation procedure further generalizes to other domains in which outdoor semantic annotations are useful such as autonomous driving, scene analysis, and augmented reality. The proposed procedure is semi-automated, as drone-acquired orthophotos and videos must be annotated manually. However, the method could be fully automated by using other ground truth sources such as georeferenced and labeled 3D point clouds from the Danish Agency for Data Supply and Efficiency[1]. Although these do not capture dynamic obstacles such as vehicles and pedestrians, they are useful for obtaining annotated data of e.g. buildings, vegetation, terrain, and bridges. To realize and validate fully autonomous agricultural vehicles, realistic datasets from all possible conditions and environments are necessary. Future work should therefore focus on acquiring additional challenging illumination and weather conditions such as

[1] Kortforsyningen, Styrelsen for Dataforsyning og Effektivisering: https://download.kortforsyningen.dk/content/dhmpunktsky

night-time, rain, dust, and fog. Furthermore, actual footage of animals in their natural habitat is needed in order to train and evaluate methods for animal obstacle avoidance. During data acquisition of the datasets DK1-7 (page 24), this has shown to be a difficult task, as the animals only occasionally reside in the field, and only during the first annual harvests.

In **part III**, two methods were proposed for point-wise classification of 3D point clouds acquired with a rotating multi-beam lidar. The two methods both addressed varying point density in sparse point clouds. The first method applied an adaptive neighborhood radius during feature extraction and thus represented point neighborhoods based on 3D distance measurements. The second method represented individual point clouds as 2D range images and applied a 2D CNN for feature extraction and classification. Here, point neighborhoods were based on the horizontal and vertical laser sampling, instead of actual range measurements. Each method had its advantages and disadvantages. With metric radius-based neighborhoods, the first method always gave consistent predictions that were invariant towards point permutations. However, the classifier was unable to take advantage of laser reflectance values, as the individual laser beams of the lidar were not calibrated correctly. With 2D sampling-based neighborhoods, on the other hand, the second method was able to compensate for the inaccurate calibration of reflectance values. It was thus seen that classification performance was increased significantly, when the reflectance values were added as another range image channel to the network. However, due to the sampling-based neighborhoods, identical (and incorrect) predictions were often given to 3D structures that were far apart in 3D distance, but closely related in their horizontal or vertical sampling.
As deep learning allows for hierarchical feature extraction and represents a flexible and purely data-driven framework, future work on point cloud classification should continue on this path. Other data representations useful for deep learning should therefore be investigated such as voxelization, permutation-invariant point sets, and graph structures. Moreover, different views or representations may be combined in order to exploit advantages from multiple approaches such as shown in Chen et al. (2017).

In **part IV**, three methods were proposed for fusing point clouds with camera images for improving point- and pixel-wise classification. The first method used a lidar for continuously supervising a visual classifier of traversability. The second method combined point clouds and color images probabilistically using a conditional random field for high-level fusion. The third method fused point clouds with color and thermal images on low-level by extending the deep learning approach from part III on range images. The three methods are fundamentally different in their designs and fusion approaches. The first method did not improve classification performance of the lidar, but only the camera. The visual classifier, however, continually adapted to changes in the environment and therefore improved over time. This is especially useful in new and unseen environments where no annotations are available beforehand. The second method explored both spatial, temporal, and multi-modal relationships and showed gradual improvements in classification performance for both modalities, as more relationships were included.

The use of a conditional random field provided great flexibility and is ideal for including prior knowledge and handling multiple data relationships. As such, the second method was the only of the three to exploit inter-frame correlations by utilizing the localization system of the robot. Preliminary results for the third method did not show any improvements with sensor fusion, possibly as a result of data misalignment. However, due to the high capacity and hierarchical fusion approach, a convolutional neural network may still be the most flexible method with the greatest potential. Whereas spatial, temporal, and multi-modal relationships were hand-crafted with cost functions for the conditional random field, deep learning is purely data-driven and can thus exploit possibly undiscovered relationships. Future work should therefore attempt to increase robustness towards calibration inaccuracies and possibly include temporal dependencies between frames by combining a CNN with e.g. long short-term memory (LSTM) (Donahue et al., 2015).

In **part V**, object detection methods from multiple sensors were fused and mapped globally with semantical occupancy grid mapping. A process evaluation method was further used to extract process-relevant properties such as traversability, processability, and obstacle occupancy along the vehicle trajectory to simulate actual runtime operation. These were detected with accuracies above 90% at distances closer than 12.5 m and with accuracies at or above 80% at distances between 12.5 m and 25 m. Compared to the minimum detection distance of 12.3 m specified in the introduction, the results thus seem promising for ensuring safe traversals, although specific use cases must be evaluated under more circumstances to confirm this.
The application of global object mapping as apposed to local mapping with short time frames may in particular suit agricultural use cases. The recurring driving patterns common in both broad-acre and orchards thus allow for multiple and different views of the same physical regions, as a vehicle covers an area with back-and-forth motions. In these cases, global mapping and fusion provides essential prior knowledge from previous tracks and perhaps even previous treatments.

A large part of the proposed methods in this thesis have addressed multi-modal fusion. The ideas of improved detection performance and added redundancy are obvious and appealing. However, implementing efficient and reliable sensor fusion comes at a cost. Exact sensor calibration, registration, and synchronization is needed for initial data alignment. A flexible fusion approach is needed for handling multiple physical quantities with different data representations. And mitigating single points of error with redundancy must be handled explicitly and carefully, as many sensor fusion methods rely on the presence of all modalities simultaneously. The proposed fusion methods in this study are all capable of continuing operation when a single sensor fails. The self-supervised system stops adapting when the lidar fails, and the conditional random field looses multi-modal relationships when either the lidar or camera fails. Similarly, the certainty of occupancy grid map fusion decreases when sensors stop reporting their detections. However, continued inference is still possible for the remaining, working sensors. Other fusion approaches concatenate multi-modal features at low-level before classification (Cadena and Košecká, 2016; Namin et al., 2014). When a sensor fails,

features are undefined, and the classification thus fails. Sensor fusion does therefore not always guarantee improvements, and adding more sensors may sometimes decrease robustness rather than increasing it.

Of all methods and approaches presented in this study, the most promising approach for future research and development may be multi-modal fusion with CNNs. Although preliminary results did not show any improvement with additional modalities, CNNs provide large-capacity models and flexible frameworks for hierarchical feature extraction and hierarchical fusion. A similar method on a related task has thus shown that different modalities can be fused at both low- and high-level by adding interactions between intermediate layers from parallel subnetworks (Chen et al., 2017). Training deep networks, however, requires large annotated datasets. For research on autonomous cars, large-scale public datasets already exist, and more annotated data are continuously released. For agricultural use cases, such datasets are non-existent and require tremendous efforts to collect. Therefore, the release in this work of two multi-modal datasets for obstacle detection in agriculture may facilitate future research in the domain. And furthermore, the proposed semi-automated annotation process utilizing global GPS labels may prove useful for generating large-scale training data.

15 Conclusion

The main objective of this study was to investigate how a rotating multi-beam lidar can be used for obstacle detection and recognition in agricultural environments, either alone or combined with other sensing modalities. The problems have been addressed in two separate parts of the thesis, namely 1) point cloud classification, and 2) multi-modal fusion. The contributions of the study are made up by individual methods and applications within these parts, together with two published multi-modal datasets and a semi-automated procedure for obtaining ground truth object annotations.

Point cloud classification deals with the issue of discriminating 3D point structures based on shapes and neighborhoods. Two methods were proposed for point classification of lidar-acquired 3D point clouds. The methods both addressed sparsity and local point neighborhoods and were used for consistent feature extraction across entire point clouds. As such, they addressed the first research question of how obstacles can be recognized in sparse point clouds from a rotating multi-beam lidar. One method, based on a traditional processing pipeline, outperformed a generic 3D feature descriptor designed for dense point clouds. The other method used a 2D range image representation which was shown to enable state-of-the-art semantic segmentation in 2D with deep learning. Together, the two methods showed that sparsity in lidar-acquired point clouds can be addressed intelligently by utilizing the known sample patterns. They further showed that the different data representations had different advantages and disadvantages. A combination of multiple representations may therefore accumulate the benefits and potentially provide increased accuracy and robustness.

Multi-modal fusion deals with the issue of combining sensor data from different modalities to increase classification robustness and confidence. It addresses the second research question of how lidar technology can cooperate with other sensing modalities in agricultural environments. Four methods were proposed and evaluated for sensor fusion between lidar and other sensing modalities, incorporating both spatial, temporal, and multi-modal relationships. The methods illustrated the potential of fusion by increasing classification performance when more sensors and information were introduced. Although not evaluated, the methods further introduced redundancy that may increase safety by mitigating single points of error. Multi-modal fusion is thus a powerful tool for increasing accuracy and safety. However, the approaches showed that exact calibration, registration, and synchronization is essential for sensor fusion to work. Therefore, the potential performance gains inevitably come at a cost.

FieldSAFE is a multi-modal dataset including both static and dynamic obstacles. It is one of many datasets acquired and used in the study representing a wide range of realistic agricultural environments such as grass fields, row crops, and orchards. FieldSAFE was recorded in an agricultural grass field using a custom-made perception platform including lidar, radar, stereo camera, thermal camera, 360° camera, IMU and GPS. The dataset can help facilitate future research on obstacle detection and recognition in agriculture. A semi-automated procedure has further produced large-scale ground truth object annotations that enable training of large-capacity models such as state-of-the-art deep neural networks.

Future work on agricultural obstacle detection with lidar should investigate and compare multiple variants of deep learning for classifying sparse point clouds and for fusing multiple modalities. Related work in other domains has thus shown great performance increases when applying convolutional neural networks on various data representations for 3D point clouds. In order to reach full autonomy, real-time integration of methods should further be combined with comprehensive validation and testing in a wider range of realistic scenarios. This study, however, has shown that recent advancements for autonomous vehicles in the automotive industry can be transferred to the agricultural domain. High-capacity data-driven approaches can thus be applied efficiently when large-scale datasets such as FieldSAFE are available.

Bibliography

Abidine, A. Z., Heidman, B. C., Upadhyaya, S. K., and Hills, D. J. (2004). Autoguidance system operated at high speed causes almost no tomato damage. *California Agriculture*, 58(1):44–47.

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282.

Ahtiainen, J., Peynot, T., Saarinen, J., Scheding, S., and Visala, A. (2015). Learned ultra-wideband radar sensor model for augmented lidar-based traversability mapping in vegetated environments. In *Information Fusion (Fusion), 2015 18th International Conference on*, pages 953–960. IEEE.

Anguelov, D., Taskarf, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., and Ng, A. (2005). Discriminative learning of markov random fields for segmentation of 3d scan data. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 169–176. IEEE.

ASI (2016). Autonomous Solutions. https://www.asirobots.com/farming/. Accessed: 2016-09-28.

Asvadi, A., Garrote, L., Premebida, C., Peixoto, P., and Nunes, U. J. (2017). Multimodal vehicle detection: fusing 3d-lidar and color camera data. *Pattern Recognition Letters*.

Ball, D., Upcroft, B., Wyeth, G., Corke, P., English, A., Ross, P., Patten, T., Fitch, R., Sukkarieh, S., and Bate, A. (2016). Vision-based obstacle detection and navigation for an agricultural robot. *Journal of Field Robotics*, 33(8):1107–1130.

Benet, B., Rousseau, V., and Lenain, R. (2016). Fusion between a color camera and a tof camera to improve traversability of agricultural vehicles. In *Conférence CIGR-AGENG 2016. The 6th International Workshop Applications of Computer Image Analysis and Spectroscopy in Agriculture*, pages 8–p.

Bergerman, M., Singh, S., and Hamner, B. (2012). Results with autonomous vehicles operating in specialty crops. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1829–1835. IEEE.

Bertozzi, M. and Broggi, A. (1998). GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Trans. Image Process.*, 7(1):62–81.

Bevly, D. and Cobb, S. (2010). *GNSS for Vehicle Control*. GNSS technology and applications series. Artech House.

Bevly, D. M., Gerdes, J. C., and Wilson, C. (2002). The use of gps based velocity measurements for measurement of sideslip and wheel slip. *Vehicle System Dynamics*, 38(2):127–147.

Biber, P. (2005). Dynamic maps for long-term operation of mobile service robots. In *In Proc. of Robotics: Science and Systems (RSS*.

Blackmore, S. et al. (2009). New concepts in agricultural automation. In *HGCA conference*.

Borenstein, J. and Koren, Y. (1991). The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE transactions on robotics and automation*, 7(3):278–288.

Boykov, Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 105–112. IEEE Comput. Soc.

Brooks, C. A. and Iagnemma, K. (2012). Self-supervised terrain classification for planetary surface exploration rovers. *Journal of Field Robotics*, 29(3):445–468.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.

Cadena, C. and Košecká, J. (2016). Recursive Inference for Prediction of Objects in Urban Environments. In *International Symposium on Robotics Research*, pages 539–555.

Cambou, P., Girardin, G., and Tschudi, Y. (2018). Sensors for robotic vehicles 2018. *Yole Développement*.

Case IH (2016). Case IH Autonomous Concept Vehicle. https://www.caseih.com/northamerica/en-us/Pages/campaigns/autonomous-concept-vehicle.aspx. Accessed: 2018-03-09.

Chang, C.-c. and Lin, C.-j. (2011). LIBSVM. *ACM Transactions on Intelligent Systems and Technology*, 2(3):1–27.

Cheein, F. A., Steiner, G., Paina, G. P., and Carelli, R. (2011). Optimized eif-slam algorithm for precision agriculture mapping based on stems detection. *Computers and electronics in agriculture*, 78(2):195–207.

Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR*.

Chen, Y. and Medioni, G. (1992). Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155.

Christiansen, P., Hansen, M. K., Steen, K. A., Karstoft, H., and Jørgensen, R. N. (2015). Advanced sensor platform for human detection and protection in autonomous farming. In *Precision agriculture'15*, pages 1330–1334. Wageningen Academic Publishers.

Christiansen, P., Kragh, M., Steen, K. A., Karstoft, H., and Jørgensen, R. N. (2017). Platform for evaluating sensors and human detection in autonomous mowing operations. *Precision Agriculture*, 18(3):350–365.

Christiansen, P., Nielsen, L. N., Steen, K. A., Jørgensen, R. N., and Karstoft, H. (2016a). Deepanomaly: combining background subtraction and deep learning for detecting obstacles and anomalies in an agricultural field. *Sensors*, 16(11):1904.

Christiansen, P., Sørensen, R., Skovsen, S., Jæger, C. D., Jørgensen, R. N., Karstoft, H., and Arild Steen, K. (2016b). Towards Autonomous Plant Production using Fully Convolutional Neural Networks. In *International Conference on Agricultural Engineering*. International Commission of Agricultural and Biosystems Engineering.

Christiansen, P., Steen, K. A., Jørgensen, R. N., and Karstoft, H. (2014). Automated detection and recognition of wildlife using thermal cameras. *Sensors*, 14(8):13778–13793.

Colleens, T. and Colleens, J. (2007). Occupancy grid mapping: An empirical evaluation. In *Control & Automation, 2007. MED'07. Mediterranean Conference on*, pages 1–6. IEEE.

Dahlkamp, H., Kaehler, A., Stavens, D., Thrun, S., and Bradski, G. (2006). Self-supervised Monocular Road Detection in Desert Terrain. *Proc of Robotics Science and Systems RSS*.

De Deuge, M., Quadros, A., Hung, C., and Douillard, B. (2013). Unsupervised feature learning for classification of outdoor 3d scans. In *Australasian Conference on Robotics and Automation*, volume 2, page 1.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

Dima, C., Vandapel, N., and Hebert, M. (2004). Classifier fusion for outdoor obstacle detection. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 1, pages 665–671 Vol.1. IEEE.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634.

Douillard, B., Brooks, A., and Ramos, F. (2009). A 3d laser and vision based classifier. In *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on*, pages 295–300. IEEE.

Douillard, B., Fox, D., and Ramos, F. (2010a). A spatio-temporal probabilistic model for multi-sensor multi-class object recognition. In *Robotics Research*, pages 123–134. Springer.

Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, a., Morton, P., and Frenkel, a. (2011). On the segmentation of 3D lidar point clouds. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2798–2805. Ieee.

Douillard, B., Underwood, J., Melkumyan, N., Singh, S., Vasudevan, S., Brunner, C., and Quadros, A. (2010b). Hybrid elevation maps: 3d surface models for segmentation. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1532–1538. IEEE.

Eitel, A., Springenberg, J. T., Spinello, L., Riedmiller, M., and Burgard, W. (2015). Multimodal deep learning for robust RGB-D object recognition. *IEEE International Conference on Intelligent Robots and Systems*, 2015-December:681–687.

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57.

Emmi, L., Gonzalez-de Soto, M., Pajares, G., and Gonzalez-de Santos, P. (2014). New trends in robotics for agriculture: integration and assessment of a real fleet of robots. *The Scientific World Journal*, 2014.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395.

Fleischmann, P. and Berns, K. (2016). A stereo vision based obstacle detection system for agricultural applications. In *Field and Service Robotics*, pages 217–231. Springer.

Freitas, G., Hamner, B., Bergerman, M., and Singh, S. (2012). A practical obstacle detection system for autonomous orchard vehicles. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3391–3398. IEEE.

Gaidon, A., Wang, Q., Cabon, Y., and Vig, E. (2016). Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*.

Gebbers, R. and Adamchuk, V. I. (2010). Precision agriculture and food security. *Science*, 327(5967):828–831.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*.

Geiger, A., Moosmann, F., Car, Ö., and Schuster, B. (2012). Automatic camera and range sensor calibration using a single shot. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3936–3943. IEEE.

Golovinskiy, A., Kim, V. G., and Funkhouser, T. (2009). Shape-based recognition of 3d point clouds in urban environments. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2154–2161. IEEE.

Hadsell, R., Bagnell, J. A., Huber, D., and Hebert, M. (2010). Space-carving kernels for accurate rough terrain estimation. *The International Journal of Robotics Research*, 29(8):981–996.

Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Scoffier, M., Kavukcuoglu, K., Muller, U., and LeCun, Y. (2009). Learning long-range vision for autonomous off-road driving. *Journal of Field Robotics*, 26(2):120–144.

Hähnel, D. (2004). *Mapping with Mobile Robots*. PhD thesis, University of Freiburg.

Haralick, R. M., Shanmugam, K., and Dinstein, I. (1973). Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 3(6):610–621.

Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107.

Häselich, M., Arends, M., Wojke, N., Neuhaus, F., and Paulus, D. (2013). Probabilistic terrain classification in unstructured environments. *Robotics and Autonomous Systems*, 61(10):1051–1059.

Hermans, A., Floros, G., and Leibe, B. (2014). Dense 3D semantic mapping of indoor scenes from RGB-D images. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2631–2638. IEEE.

Himmelsbach, M., Luettel, T., and Wuensche, H.-J. (2009). Real-time object classification in 3d point clouds using point feature histograms. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 994–1000. IEEE.

Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L. (2016). Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*.

Huber, D., Kanade, T., et al. (2011). Integrating lidar into stereo for fast and improved disparity computation. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, pages 405–412. IEEE.

Ingibergsson, J. T. M., Suvei, S.-D., Hansen, M. K., Christiansen, P., and Schultz, U. P. (2015). Towards a DSL for Perception-Based Safety Systems. In *6th International Workshop on Domain-Specific Languages and models for Robotic systems (DSLROB 15)*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456.

ISO/FDIS 18497 (2017). Agricultural machinery and tractors – Safety of highly automated agricultural machines. Standard, International Organization for Standardization, Geneva, CH.

Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., and Bengio, Y. (2017). The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1175–1183. IEEE.

Johnson, A. E. and Hebert, M. (1999). Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):433–449.

Jörg, K.-W. (1995). World modeling for an autonomous mobile robot using heterogenous sensor information. *Robotics and Autonomous Systems*, 14(2-3):159–170.

Kim, D., Sun, J., Oh, S. M., Rehg, J. M., and Bobick, A. F. (2006). Traversability classification using unsupervised on-line visual learning for outdoor robot navigation. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 518–525. IEEE.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Konrad, M., Nuss, D., and Dietmayer, K. (2012). Localization in digital maps for road course estimation using grid maps. In *2012 IEEE Intelligent Vehicles Symposium*, pages 87–92.

Korthals, T., Exner, J., Schöpping, T., and Hesse, M. (2017a). Semantical occupancy grid mapping framework. In *Mobile Robots (ECMR), 2017 European Conference on*, pages 1–8. IEEE.

Korthals, T., Kragh, M., Christiansen, P., Karstoft, H., Jørgensen, R. N., and Rückert, U. (2018). Multi-Modal Detection and Mapping of Static and Dynamic Obstacles in Agriculture for Process Evaluation. *Accepted for publication in Frontiers in Robotics and AI, Research Topic: Multi-modal Sensor Fusion*.

Korthals, T., Kragh, M., Christiansen, P., and Rückert, U. (2017b). Towards Inverse Sensor Mapping in Agriculture. In *IROS 2017 Workshop on Agricultural Robotics: learning from Industry 4.0 and moving into the future*, Vancouver.

Kragh, M., Bjerge, K., and Ahrendt, P. (2016a). 3d impurity inspection of cylindrical transparent containers. *Measurement Science and Technology*, 28(1). Technical Note.

Kragh, M., Christiansen, P., Korthals, T., Jungeblut, T., Karstoft, H., and Jørgensen, R. N. (2016b). Multi-Modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. In *International Conference on Agricultural Engineering*. International Commission of Agricultural and Biosystems Engineering.

Kragh, M., Nyholm Jørgensen, R., and Pedersen, H. (2015). *Object Detection and Terrain Classification in Agricultural Fields using 3D Lidar Data*, volume 9163, pages 188–197. Springer. Springer, Lecture Notes in Computer Science.

Kragh, M., Sand, M., and Karstoft, H. (2018). Multi-Modal Semantic Segmentation in 3D with Range Images. *Draft*.

Kragh, M. and Underwood, J. (2017). Multi-modal obstacle detection in unstructured environments with conditional random fields. *arXiv preprint arXiv:1706.02908*.

Kragh, M., Underwood, J., and Karstoft, H. (2016c). Self-supervised Traversability Assessment in Field Environments with Lidar and Camera. In *International Conference on Agricultural Engineering*. International Commission of Agricultural and Biosystems Engineering.

Kragh, M. F., Christiansen, P., Laursen, M. S., Larsen, M., Steen, K. A., Green, O., Karstoft, H., and Jørgensen, R. N. (2017). Fieldsafe: Dataset for obstacle detection in agriculture. *Sensors*, 17(11).

Krähenbühl, P. and Koltun, V. (2012). Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. *Advances in Neural Information Processing Systems 24*, (4):109—-117.

Laible, S., Khan, Y. N., and Zell, A. (2013). Terrain classification with conditional random fields on fused 3d lidar and camera data. In *Mobile Robots (ECMR), 2013 European Conference on*, pages 172–177. IEEE.

Lalonde, J. F., Vandapel, N., Huber, D. F., and Hebert, M. (2006). Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics*, 23(10):839–861.

Lang, T., Plagemann, C., and Burgard, W. (2007). Adaptive non-stationary kernel regression for terrain modeling. In *Robotics: Science and Systems*, volume 6.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Levinson, J. and Thrun, S. (2013). Automatic Online Calibration of Cameras and Lasers. In *Robotics: Science and Systems IX*. Robotics: Science and Systems Foundation.

Levinson, J. and Thrun, S. (2014). Unsupervised calibration for multi-beam lasers. In *Experimental Robotics*, pages 179–193. Springer.

Li, B., Zhang, T., and Xia, T. (2016). Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*.

Lim, E. H. and Suter, D. (2009). 3d terrestrial lidar classifications with super-voxels and multi-scale conditional random fields. *Computer-Aided Design*, 41(10):701–710.

Litman, T. (2017). *Autonomous vehicle implementation predictions*. Victoria Transport Policy Institute.

Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440.

Lookingbill, A., Rogers, J., Lieb, D., Curry, J., and Thrun, S. (2007). Reverse optical flow for self-supervised adaptive autonomous robot navigation. *International Journal of Computer Vision*, 74(3):287.

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Luettel, T., Himmelsbach, M., and Wuensche, H.-J. (2012). Autonomous Ground Vehicles—Concepts and a Path to the Future. *Proceedings of the IEEE*, 100(Special Centennial Issue):1831–1839.

Lütkebohle, I. (2017). Determinism in Robotics Software. Conference presentation, ROSCon, https://roscon.ros.org/2017/presentations/ROSCon%202017%20Determinism%20in%20ROS.pdf. Online; accessed 31 October 2017.

Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15.

Maier, D., Bennewitz, M., and Stachniss, C. (2011). Self-supervised obstacle detection for humanoid navigation using monocular vision and sparse laser data. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1263–1269. IEEE.

Mao, S., Li, L., Guo, J., and Zhao, C. (2015). A novel obstacle detection method based on monocular camera and laser radar. In *Computational Intelligence and Design (ISCID), 2015 8th International Symposium on*, volume 1, pages 511–515. IEEE.

Maturana, D. and Scherer, S. (2015). Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE.

McDaniel, M. W., Nishihata, T., Brooks, C. a., and Iagnemma, K. (2010). Ground plane identification using LIDAR in forested environments. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3831–3836.

Milella, A., Reina, G., and Underwood, J. (2015). A Self-learning Framework for Statistical Ground Classification using Radar and Monocular Vision. *Journal of Field Robotics*, 32(1):20–41.

Milella, A., Reina, G., Underwood, J., and Douillard, B. (2014). Visual ground segmentation by radar supervision. *Robotics and Autonomous Systems*, 62(5):696–706.

Moore, T. and Stouch, D. (2014). A generalized extended kalman filter implementation for the robot operating system. In *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer.

Moorehead, S. J., Wellington, C. K., Gilmore, B. J., and Vallespi, C. (2012). Automating orchards: A system of autonomous tractors for orchard maintenance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Agricultural Robotics, Vilamoura, Portugal*.

Moosmann, F., Pink, O., and Stiller, C. (2009). Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion. *2009 IEEE Intelligent Vehicles Symposium*, pages 215–220.

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems.

Munoz, D., Bagnell, J. A., and Hebert, M. (2012). Co-inference for multi-modal scene analysis. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, pages 668–681, Berlin, Heidelberg. Springer-Verlag.

Nam, W., Dollár, P., and Han, J. H. (2014). Local decorrelation for improved detection. *Adv. Neural Inf. Process. Syst.*, pages 1–9.

Namin, S. T., Najafi, M., and Petersson, L. (2014). Multi-view terrain classification using panoramic imagery and LIDAR. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, number Iros, pages 4936–4943. IEEE.

Namin, S. T., Najafi, M., Salzmann, M., and Petersson, L. (2015). A Multi-modal Graphical Model for Scene Analysis. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 1006–1013. IEEE.

Nebot, E. and Durrant-Whyte, H. (1999). Initial calibration and alignment of low-cost inertial navigation units for land vehicle applications. *Journal of Robotic Systems*, 16(2):81–92.

Noon, R. (1994). *Engineering Analysis of Vehicular Accidents*. Taylor & Francis.

Pandey, G., McBride, J. R., Savarese, S., and Eustice, R. M. (2015). Automatic extrinsic calibration of vision and lidar by maximizing mutual information. *Journal of Field Robotics*, 32(5):696–722.

Papon, J., Abramov, A., Schoeler, M., and Worgotter, F. (2013). Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2027–2034. IEEE.

Pathak, K., Birk, A., Poppinga, J., and Schwertfeger, S. (2007). 3D Forward sensor modeling and application to occupancy grid based sensor fusion. *IEEE International Conference on Intelligent Robots and Systems*, 2:2059–2064.

Pele, O. and Werman, M. (2010). The Quadratic-Chi Histogram Distance Family. In *Lecture Notes in Computer Science*, volume 6312 LNCS, pages 749–762.

Peynot, T., Scheding, S., and Terho, S. (2010). The Marulan Data Sets: Multi-Sensor Perception in Natural Environment with Challenging Conditions. *International Journal of Robotics Research*, 29(13):1602–1607.

Pezzementi, Z., Tabor, T., Hu, P., Chang, J. K., Ramanan, D., Wellington, C., Babu, B. P. W., and Herman, H. (2017). Comparing apples and oranges: Off-road pedestrian detection on the nrec agricultural person-detection dataset. *arXiv preprint arXiv:1707.07169*.

Pix4D (2014). Pix4D. http://pix4d.com/. Accessed: 2017-09-05.

Posner, I., Cummins, M., and Newman, P. (2009). A generative framework for fast urban labeling using spatial and temporal context. *Autonomous Robots*, 26(2-3):153–170.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5105–5114.

Quadros, A., Underwood, J. P., and Douillard, B. (2012). An occlusion-aware feature for range images. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4428–4435. IEEE.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*.

Rao, D., Deuge, M. D., Nourani–Vatani, N., Williams, S. B., and Pizarro, O. (2017). Multimodal learning and inference from visual and remotely sensed data. *The International Journal of Robotics Research*, 36(1):24–43.

Redmon, J. and Farhadi, A. (2016). YOLO9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*.

Reina, G., Milella, A., Rouveure, R., Nielsen, M., Worst, R., and Blas, M. R. (2016a). Ambient awareness for agricultural robotic vehicles. *Biosystems Engineering*, 146:114–132.

Reina, G., Milella, A., and Worst, R. (2016b). Lidar and stereo combination for traversability assessment of off-road robotic vehicles. *Robotica*, 34(12):2823–2841.

Riegler, G., Ulusoy, A. O., and Geiger, A. (2017). Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 3.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3234–3243.

Rovira-Más, F. and Reid, J. (2004). 3d density and density maps for stereo vision-based navigation. In *Automation Technology for Off-Road Equipment Proceedings of the 2004 Conference*, page 24. American Society of Agricultural and Biological Engineers.

Rusu, R. B., Blodow, N., and Beetz, M. (2009). Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. IEEE.

Rusu, R. B., Blodow, N., Marton, Z. C., and Beetz, M. (2008). Aligning point cloud views using persistent feature histograms. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3384–3391. IEEE.

Rusu, R. B., Bradski, G., Thibaux, R., and Hsu, J. (2010). Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162. IEEE.

Salti, S., Tombari, F., and Di Stefano, L. (2014). Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264.

Shen, Y., Feng, C., Yang, Y., and Tian, D. (2017). Neighbors do help: Deeply exploiting local structures of point clouds. *arXiv preprint arXiv:1712.06760*.

Shi, B., Bai, S., Zhou, Z., and Bai, X. (2015). Deeppano: Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343.

Simonovsky, M. and Komodakis, N. (2017). Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proc. CVPR*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Stachniss, C. (2009). *Robotic mapping and exploration*, volume 55. Springer.

Steder, B., Rusu, R. B., Konolige, K., and Burgard, W. (2011). Point feature extraction on 3d range scans taking into account object boundaries. In *Robotics and automation (icra), 2011 ieee international conference on*, pages 2601–2608. IEEE.

Steen, K. A., Christiansen, P., Karstoft, H., and Jørgensen, R. N. (2016). Using Deep Learning to Challenge Safety Standard for Highly Autonomous Machines in Agriculture. *Journal of Imaging*, 2(1):6.

Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953.

Suvei, S.-D., Haarslev, F., Bodenhagen, L., and Krüger, N. (2018). Stereo and lidar fusion based detection of humans and other obstacles in farming scenarios. In *Proceedings of the 13th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 166–173.

Tabor, T., Pezzementi, Z., Vallespi, C., and Wellington, C. (2015). People in the weeds: Pedestrian detection goes off-road. In *Safety, Security, and Rescue Robotics (SSRR), 2015 IEEE International Symposium on*, pages 1–7. IEEE.

Taylor, Z. and Nieto, J. (2016). Motion-based calibration of multimodal sensor extrinsics and timing offset estimation. *IEEE Transactions on Robotics*, 32(5):1215–1229.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. MIT Press, Cambridge, Mass.

Thrun, S. et al. (2002). Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, 1:1–35.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., et al. (2006). Stanley: The robot that won the darpa grand challenge. *Journal of field Robotics*, 23(9):661–692.

Tu, G., Hansen, M., Kryger, P., and Ahrendt, P. (2016). Automatic behaviour analysis system for honeybees using computer vision. *Computers and Electronics in Agriculture*, 122(March):10–18.

Underwood, J., Hill, A., and Scheding, S. (2007). Calibration of range sensor pose on mobile platforms. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3866–3871. IEEE.

Underwood, J. P., Hill, A., Peynot, T., and Scheding, S. J. (2010). Error modeling and calibration of exteroceptive sensors for accurate mapping applications. *Journal of Field Robotics*, 27(1):2–20.

Vandapel, N., Huber, D., Kapuria, A., and Hebert, M. (2004). Natural terrain classification using 3-D ladar data. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 5:5117–5122.

Vondrick, C., Patterson, D., and Ramanan, D. (2013). Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204.

Weiss, U. and Biber, P. (2011). Plant detection and mapping for agricultural robots using a 3d lidar sensor. *Robotics and autonomous systems*, 59(5):265–273.

Wellington, C., Courville, A., and Stentz, A. (2006). A generative model of terrain for autonomous navigation in vegetation. *The International Journal of Robotics Research*, 25(12):1287–1304.

Wellington, C., Courville, A. C., and Stentz, A. (2005). Interacting markov random fields for simultaneous terrain modeling and obstacle detection. In *Robotics: Science and Systems*, volume 6, pages 1–8.

Wellington, C. and Stentz, A. (2004). Online Adaptive Rough-Terrain Navigation in Vegetation. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 1:96–101 Vol.1.

Wu, B., Wan, A., Yue, X., and Keutzer, K. (2017). Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. *arXiv preprint arXiv:1710.07368*.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920.

Wurm, K. M., Kretzschmar, H., Kümmerle, R., Stachniss, C., and Burgard, W. (2014). Identifying vegetation from laser data in structured outdoor environments. *Robotics and Autonomous Systems*, 62(5):675–684.

Xiao, L., Dai, B., Liu, D., Hu, T., and Wu, T. (2015). CRF based road detection with multi-sensor fusion. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, number Iv, pages 192–198. IEEE.

Yi, L., Su, H., Guo, X., and Guibas, L. (2017). Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *Computer Vision and Pattern Recognition (CVPR)*.

Yue, X., Wu, B., Keutzer, K., Sangiovanni-Vincentelli, A., and Seshia, S. A. (2017). A lidar point cloud generator: from a virtual world to autonomous driving. *Submitted to NIPS 2017 MLITS Workshop*.

Zhang, R., Candra, S. A., Vetter, K., and Zakhor, A. (2015). Sensor fusion for semantic segmentation of urban scenes. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1850–1857. IEEE.

Zhou, S., Xi, J., McDaniel, M. W., Nishihata, T., Salesses, P., and Iagnemma, K. (2012). Self-supervised learning to visually detect terrain surfaces for autonomous robots operating in forested terrain. *Journal of Field Robotics*, 29(2):277–297.

Zhu, X., Zhao, H., Liu, Y., Zhao, Y., and Zha, H. (2010). Segmentation and classification of range image from an intelligent vehicle in urban environment. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1457–1462. IEEE.

Glossary

ACFR	Australian Centre for Field Robotics
ATV	all-terrain vehicle
CAD	computer-aided design
CNN	convolutional neural network
CRF	conditional random field
DOF	degrees of freedom
FCN	fully convolutional network
FOV	field of view
GLCM	Gray-Level Co-Occurrence Matrix
GMM	Gaussian mixture model
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HMM	hidden Markov model
ICP	Iterative Closest Point
IMU	inertial measurement unit
IoU	intersection over union
ISM	inverse sensor model
OGM	occupancy grid map
PCA	principal component analysis
PPS	pulse per second
ROS	Robot Operating System
RTK	Real Time Kinematic
SAFE	Safer Autonomous Farming Equipment
SLAM	simultaneous localization and mapping
SOGM	semantical occupancy grid map
SVM	support vector machine
UTM	Universal Transverse Mercator

Publications Part VII

Paper 1

Advanced sensor platform for human detection and protection in autonomous farming

Peter Christiansen, Mikkel Fly Kragh, Kim Arild Steen, Henrik Karstoft, and Rasmus Nyholm Jørgensen

Peer reviewed
Presented at 10th European Conference on Precision Agriculture (ECPA), July 2015, Tel Aviv, Israel

Advanced sensor platform for human detection and protection in autonomous farming

P. Christiansen[1], M. Kragh[1,†], K. A. Steen[1], H. Karstoft[1], R. N. Jørgensen[1]
[1]*Department of Engineering – Signal Processing, Faculty of Science and Technology, Aarhus University, Finlandsgade 22, 8200 Aarhus N, Denmark*
[†]Corresponding author: mkha@eng.au.dk

Abstract

The concept of autonomous farming concerns automatic agricultural machines operating safely and efficiently without human intervention. In order to ensure safe autonomous operation, real-time risk detection and avoidance must be performed. This paper presents a flexible vehicle-mounted sensor platform for recording positional and imaging data with a total of seven sensors. Different imaging modalities are chosen for robust detection performances in a variety of weather and lighting conditions. Different algorithms applied to recordings from a grass-harvesting case study show that it is possible to detect humans, whereas small animals located in front of the vehicle represent a much greater challenge.

Keywords: safe farming, sensor platform, object detection, computer vision

Introduction

Current technology is capable of automatically navigating and operating agricultural machinery, such as tractors and harvesters, efficiently and more precisely compared to manual operation. However, a crucial deficiency in this technology concerns the safety aspects. In order for an autonomous vehicle to operate safely and be certified for unsupervised operation, it must perform automatic real-time risk detection and avoidance in the field with high reliability.

Robust risk detection imposes a number of challenges for the sensor platform. Varying weather and lighting conditions influence the image quality of sensor modalities in different ways, and thus no sensor is single-handedly capable of detecting objects reliably under all conditions. Active sensors such as radar and LiDAR, and passive sensors such as RGB camera, stereo camera and thermal camera have different strengths and weaknesses concerning weather, lighting, range and resolution, and therefore a variety of these sensors are needed to cover all scenarios (Rasshofer & Gresser 2005). In addition, pose estimation sensors such as accelerometers, gyroscopes and GPS are needed for estimating the vehicle position, velocity and orientation and for synchronizing and registering subsequent frames acquired from the imaging sensors.

Today, driver assistance systems are available for a large number of modern passenger cars, and completely autonomous vehicles operating in urban and sub-urban environments are emerging for experimental usage (Luettel et al. 2012). In the agricultural sector, a variety of machines have been operating autonomously for a decade using either precise GPS coordinates and/or cameras detecting structures in the field (CLAAS Steering Systems 2011). Efforts are made to fully automate the process in a driverless solution, but safety aspects currently prevent authorization for this. For instance the QUAD-AV project has investigated microwave radar, stereo vision,

LiDAR and thermography for detecting obstacles in an agricultural context (Rouveure et al. 2012). Within the project, a detailed study of stereo vision has shown promising results on ground/non-ground classification (Reina & Milella 2012).

The paper describes a flexible vehicle-mounted sensor platform. The sensor platform records imaging data and vehicle position for a moving vehicle using three passive imaging sensors, two active sensors and two pose/position estimation sensors. The sensor platform is designed to record simultaneous data from all sensors, thus preparing for subsequent offline processing. Recordings from a grass-harvesting case study are documented. In the study, different objects including humans of different sizes, appearances and postures, as well as different animals are placed in front of the setup and detected automatically. Based on different object detection algorithms carried out on the imaging sensors, an initial evaluation of the different sensors is given.

Sensors

An overview of the strengths and weaknesses of the selected imaging and active sensors are presented in Table 1. The qualities are evaluated individually and under various conditions.

Table 1. Strengths and weaknesses of sensors.

	Name	RGB	Stereo	Thermal	LiDAR	UWB Radar
Specification	Range	medium	medium	medium	long	medium
	Resolution	+	+	-	-	-
	Depth information	-	+	-	+	+
	Heat information	-	-	+	-	-
	Color information	+	+	-	-	-
	Cost	low	medium	medium	high	medium
Robustness	Light changes	-	-	+	+	+
	Weather changes	-	-	-	+	+
	Camouflaged objects	-	+	+	+	+
	Protruding objects	-	+	-	+	+
	Non-protruding objects	+	-	+	-	-

An *RGB camera* captures the modality of visible light. The sensor is useful for identifying the perceived objects as it provides visual characteristics such as texture, color and shape in high resolution at low cost. It is invariant to protrusion, meaning that non-protruding objects such as small animals, a fallen human or humans/animals in high crops are still visible. However, visual characteristics are affected by weather conditions (rain, fog and snow) and illumination such as dim light (night) or direct light (causing shadows). An RGB camera is not able to exploit depth information to emphasize protruded objects and the lack of depth makes the positioning of objects in 3D space difficult.

A *stereo* camera enables 3D imaging data (depth and color information). Depth and color information are registered and the sensor is thus able to exploit the advantages of both modalities. Depth information can be used to see protruding objects and visually camouflaged animals easily while determining the position of an object relative to the

vehicle. In this way, depth-aware algorithms can abstract from the very different visual characteristics of objects (shape, color and texture) creating simple detection algorithms. Like the RGB camera, the stereo camera is sensitive to illumination and weather conditions, although the depth information is in some cases still retrievable.

A *thermal camera* is an imaging sensor that captures heat radiation represented by intensities (temperatures) to form a monochromatic image. A thermal camera perceives objects of distinct temperatures, making it ideal for detecting living objects in temperate and colder climates, and even in foggy weather (Serrano-Cuerda et al. 2014). A key ability is that the sensed data are unaffected by non-protruding or visually camouflaged animals and that the distinctness of living objects becomes more apparent at night. However, these capabilities are much affected by the ambient temperature as living objects become indistinct when the temperature difference between the objects and the background becomes small (Serrano-Cuerda et al. 2014). The cost of a well-performing and high resolution thermal camera is very high, but low cost cameras are emerging. Object recognition capabilities are low due to a limited resolution and limited visual characteristics.

A *LiDAR* measures range data to a set of surrounding points and generates a point cloud where each point is represented by a 3D position and a reflection intensity. The LiDAR is a high cost sensor, but has dropped significantly in price in recent years. Compared to a stereo camera the LiDAR provides very exact depth information at further range and captures up to 360° horizontally. It is invariant to illumination, temperature and camouflage. The lack of visual and thermal information makes recognition of objects difficult and non-protruding objects are almost or fully undetectable.

A *radar* measures range and/or velocity information of objects by transmitting radio waves and measuring object reflections. A variety of radar technologies exist with both low and high costs. Depending on object materials and sizes, different radar frequencies are optimized for different applications. For human detection applications, ultra-wideband (UWB) short range radar operating at a few GHz is common. Radar is invariant towards changing temperature and light conditions.

Physical design

The sensor platform consists of seven sensors and a controller mounted on a common rack. The left side of Figure 1 shows the rack mounted on a tractor and the right side shows the physical placement with antennas and inertial measurement unit (IMU) at the top, sensors in the middle and the controller at the bottom. The horizontal profile in the middle is adjustable in height and angle such that the imaging and active sensors can be oriented at a downward angle depending on the vehicle height. A standard A-frame is mounted at the bottom of the rack to enable easy mounting on tractors. The A-frame is mounted with dampers for absorbing internal engine vibrations from the vehicle to reduce the amount of mechanical noise acting on the sensors. The LiDAR protrudes from the other sensors such that it has an unobstructed 180° forward field of view.

Figure 1. Sensor frame including controller.

Figure 2 presents the specific sensors and the controller used in the setup. A Logitech (Newark, California, USA) C920 webcam providing 1920×1080 pixels at 30 fps is used as the RGB camera. The stereo camera is a high dynamic range camera with logarithmic, global shutter New Imaging Technology (Paris, France) NSC1003 CMOS sensors providing 1280×1024 pixels at 25 fps. The camera uses 12-bit GRBG Bayer pixel format. The thermal camera is a shutterless Tonbo Imaging Inc (East Palo Alto, California, USA) HawkVision analog IR camera providing 640×480 pixels at 25 fps. The LiDAR is a 32-beam Velodyne (Morgan Hill, California, USA) HDL-32E laser scanner providing 70,000 points at 10 Hz with 1-100 m range. The radar is a 76 GHz Delphi ESR radar with 0.5-80 m range. The GPS is a Trimble (Sunnyvale, California, USA) AG GPS361 Real Time Kinematic (RTK) GPS enhancing the precision of GPS up to centrimetre-level accuracy. The IMU is a Vectornav (Dalla, Texas, USA) VN-100 providing synchronized 3-axis accelerometers, gyros, magnetometers and a barometric pressure sensor. The data-collecting controller is a Conpleks Robotech Controller 701. It is an embedded computer with external interfaces for all sensors that uses ROS-middleware (Robot Operating System) to easily integrate all sensors in a common framework.

System architecture

Figure 2 further illustrates the connections and bandwidths between the sensors and the controller. In ROS, each sensor is given its own node (an executable file) that is responsible for publishing one or more topics. For instance, the IMU has its own node including hardware specific drivers, and it publishes different topics related to the readings of the accelerometer, the gyroscopes and the magnetometers. For each topic, the node can send messages containing sensor data whenever a new sensor-reading is available. Each node is connected to the ROS Master which handles interactions between nodes and supplies all messages with exact timestamps. Using the rosbag package (Dirk n.d.), a recording of all desired topics (and all associated messages) to a single rosbag data-file can be obtained.

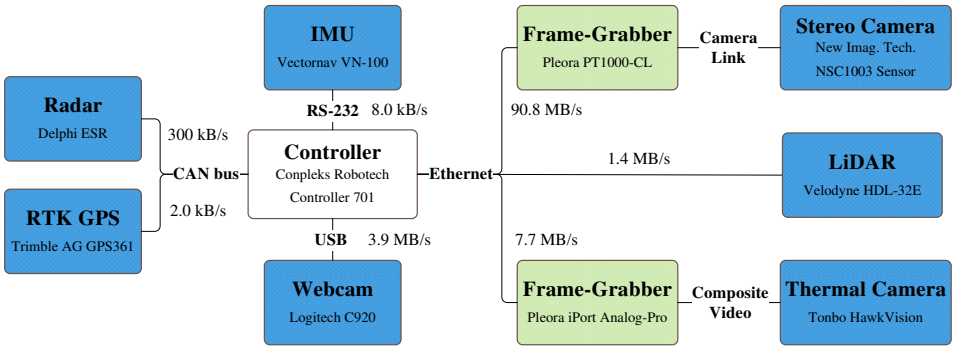


Figure 2. System overview illustrating bandwidths and interfaces between sensors, converters and the controller.

Signal processing

In order to experimentally evaluate detection performances of the different sensors in an agricultural environment, preliminary tests using different object detection algorithms have been carried out on the different imaging and active sensors.

Using only an RGB camera for detecting all possible obstacles in the field is complex and difficult and requires a very large dataset with many representations of each object. Constraining detection to only humans provides a more realistic case in this preliminary study. The RGB camera is therefore processed using a state-of-the-art pedestrian detection algorithm (Dollar et al. 2010). The stereo camera has been calibrated with a stereo calibration algorithm using a checkerboard pattern (Zhang 2000). Subsequently, a ground plane is estimated on the acquired point cloud using the RANSAC algorithm (Fischler & Bolles 1981), and points that lie above this ground plane with a certain threshold are clustered. The LiDAR data is processed using ground plane estimation and clustering of points not belonging to the ground (Moosmann et al. 2009). Clusters with more than 30 points are detected as objects. The thermal camera is processed by thresholding the (temperature-related) intensities by a constant value above the median intensity of the image (Christiansen et al. 2014). Subsequent connected components analysis is used for extracting only components that exceed a certain area. The radar was unfortunately malfunctioning during the data acquisition. Therefore no radar data is available for processing and evaluation.

Results and discussion

Data from six sensors have been recorded in a grass-harvesting case study performed in Denmark in early November. These comprise an RGB camera, a stereo camera, a thermal camera, a LiDAR, a GPS and an IMU. The radar sensor described above unfortunately malfunctioned during the recordings and is therefore omitted in the experimental evaluation.

In the following, two recordings are evaluated including 1) humans of different sizes, appearances and postures and 2) small animals placed in front of the setup.

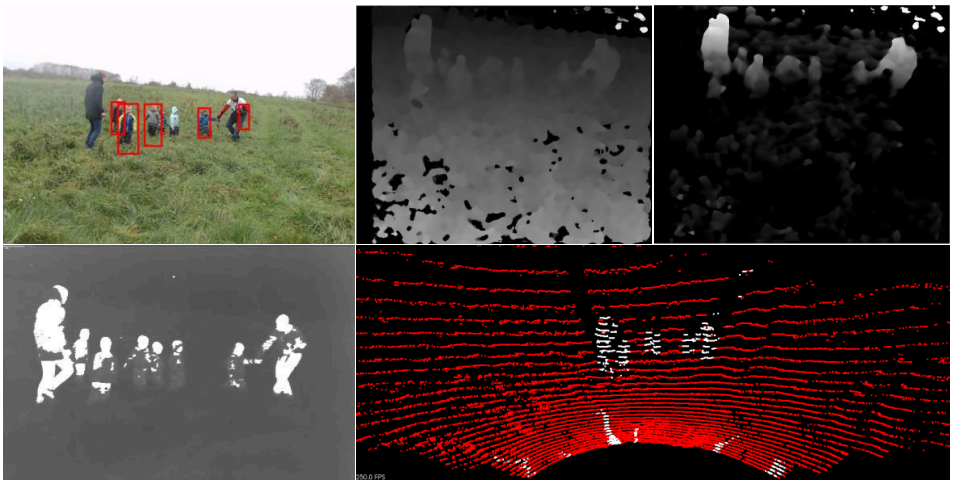


Figure 3. Detection of humans. RGB camera (top left), stereo camera disparity map (top middle), stereo camera protrusion map (top right), thermal camera (bottom left), LiDAR (bottom right).

Figure 3 depicts the human detection performances evaluated at single, synchronized frames for the RGB camera, the stereo camera, the thermal camera and the LiDAR. At the top left, the RGB camera is shown with bounding boxes indicating results of the pedestrian detection algorithm. In the top middle, the disparity map of the stereo camera is shown and, at the top right, a protrusion map indicating objects that protrude from the ground plane is visualized. At the bottom left, the thermal camera is shown with overlaid thresholded components and, at the bottom right, the LiDAR data is visualized with a ground plane and clustered objects (white).
Using only single frames, pedestrian detection applied on the RGB camera fails to detect all humans in the image. Problems concerning occlusion and humans seen from the side or from behind have been observed. However, utilizing a sequence of frames would greatly improve detection performance, as the algorithm most often fails for just a single frame and not for an entire sequence of frames. The stereo camera performs well for detecting humans that protrude from the ground plane. However, the algorithm assumes a certain level of protrusion and a flat surface in order to detect an object. The thermal camera detects all humans when their faces are visible. However, potential problems concern well insulated clothes that cover an entire body and warm weather where temperature differences are much smaller than in the present recording. Using the LiDAR clustering algorithm, most humans are detected robustly when they protrude significantly from the ground. However, problems concerning noise near the sensor due to a higher point density must be solved to avoid false alarms.

Figure 4 depicts animal detection capabilities of a rabbit and a hen. In this scenario, only the thermal camera was capable of detecting the animals. Obviously pedestrian detection applied to the RGB camera is incapable of detecting animals, and since both the algorithms of the stereo camera and LiDAR rely on significantly protruded objects, these modalities both fail to detect small animals. It is therefore clear that more advanced and task-specific algorithms must be investigated for the RGB

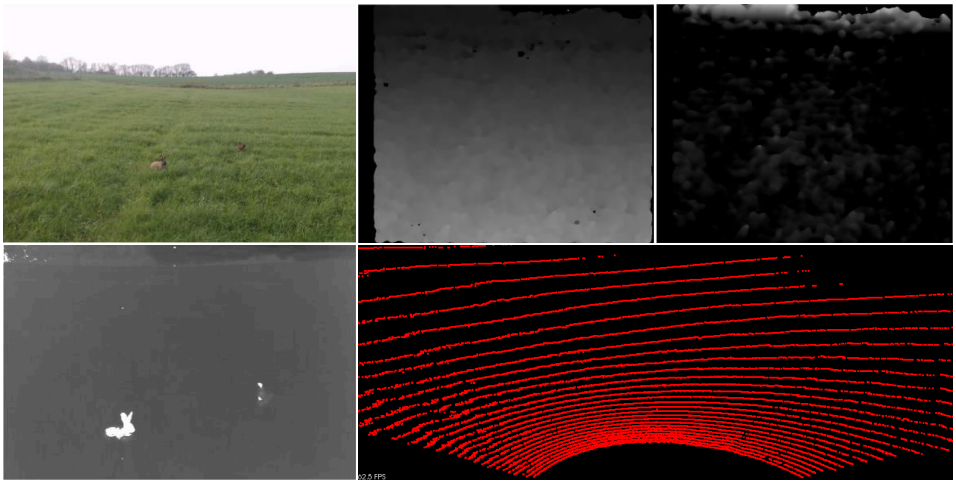


Figure 4. Detection of animals (rabbit and hen). RGB camera (top left), stereo camera disparity map (top middle), stereo camera protrusion map (top right), thermal camera (bottom left), LiDAR (bottom right).

camera, the stereo camera and the LiDAR. Although the thermal camera achieves robust and reliable detection performance for both humans and animals in this study, the results would undoubtedly be significantly worse on a warm and sunny day as reported by (Steen et al. 2012) and (Serrano-Cuerda et al. 2014). A single sensor is therefore insufficient for detecting all objects reliably invariant of temperature and lighting changes.

Conclusion

A flexible vehicle-mounted sensor platform has been developed for capturing time stamped data in the agricultural domain using imaging sensors (RGB, thermal and stereo camera), active sensors (LiDAR and radar) and pose estimations sensors (RTK GPS and IMU). Based on a case study in grass fields, an initial evaluation of the potential of different sensor modalities for detecting humans and animals is given. Using a common pedestrian detection algorithm, an RGB camera is able to detect upright pedestrians, but degrades in performance for more complex poses. The depth-aware sensors (LiDAR and stereo camera) are efficient for detecting objects that protrude significantly above the ground. The LiDAR is invariant towards changing weather and lighting conditions, whereas the stereo camera has the highest resolution making it useful for classifying objects. The thermal camera shows great capabilities in the captured dataset as it is able to detect objects of distinct temperature using a simple procedure that works both for humans and living obstacles. However, the detection would be remarkably more complicated in higher temperature environments, where living objects become indistinct in their heat signatures.

The above arguments and the case study concludes that the use of multiple modalities, more complicated procedures and a fusion of the different modalities is required to achieve a robust detection of obstacles under variable conditions. To provide a thorough evaluation of the algorithms and procedures, the dataset must be expanded to represent

more scenarios including more variable lighting and weather conditions and more representations of more objects.

Acknowledgements

This research is sponsored by the Innovation Fund Denmark as part of the project "SAFE - Safer Autonomous Farming Equipment" (project no. 16-2014-0) and "Multi-sensor system for ensuring ethical and efficient crop production" (project no. 155-2013-6). Additionally, the authors would like to thank "Traktor & Høstspecialisten A/S" for providing a CLAAS tractor for testing.

References

Christiansen, P., Steen, K., Jørgensen, R., and Karstoft, H., 2014. Automated Detection and Recognition of Wildlife Using Thermal Cameras. *Sensors*, 14(8), 13778–13793.

CLAAS Steering Systems, 2011. Tracking control optimisation. Available at: http://claas.via-us.co.uk/booklets/gps-steering-systems/download. (last accessed 12/12/14).

Dirk, T., ROS Wiki: rosbag package. Available at: http://wiki.ros.org/rosbag [Accessed March 20, 2015].

Dollar, P., Belongie, S., and Perona, P., 2010. The Fastest Pedestrian Detector in the West. In *Procedings of the British Machine Vision Conference 2010*. British Machine Vision Association, pp. 68.1–68.11.

Fischler, M.A. and Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), 381–395.

Luettel, T., Himmelsbach, M., and Wuensche, H.-J., 2012. Autonomous Ground Vehicles—Concepts and a Path to the Future. *Proceedings of the IEEE*, 100(Special Centennial Issue), 1831–1839.

Moosmann, F., Pink, O., and Stiller, C., 2009. Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion. *2009 IEEE Intelligent Vehicles Symposium*, 215–220.

Rasshofer, R.H. and Gresser, K., 2005. Automotive Radar and Lidar Systems for Next Generation Driver Assistance Functions. *Advances in Radio Science*, 3, 205–209.

Reina, G. and Milella, A., 2012. Towards Autonomous Agriculture: Automatic Ground Detection Using Trinocular Stereovision. *Sensors*, 12(12), 12405–12423.

Rouveure, R., Nielsen, M., and Petersen, A., 2012. The QUAD-AV Project: multi-sensory approach for obstacle detection in agricultural autonomous robotics. *Proceedings of 2012 International Agricultural Engineering CIGR-AgEng, Valencia, Spain*, 8–12.

Serrano-Cuerda, J., Fernández-Caballero, A., and López, M., 2014. Selection of a Visible-Light vs. Thermal Infrared Sensor in Dynamic Environments Based on Confidence Measures. *Applied Sciences*, 4(3), 331–350.

Steen, K.A., Villa-Henriksen, A., Therkildsen, O.R., and Green, O., 2012. Automatic detection of animals in mowing operations using thermal cameras. *Sensors (Basel, Switzerland)*, 12(6), 7587–97.

Zhang, Z., 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), 1330–1334.

Paper 2

Platform for evaluating sensors and human detection in autonomous mowing operations

Peter Christiansen, Mikkel Fly Kragh, Kim Arild Steen, Henrik Karstoft, and Rasmus Nyholm Jørgensen

Peer reviewed
Accepted for publication in Precision Agriculture, June 2017

Precision Agric (2017) 18:350–365
DOI 10.1007/s11119-017-9497-6

Platform for evaluating sensors and human detection in autonomous mowing operations

P. Christiansen[1] · M. Kragh[1] · K. A. Steen[1] · H. Karstoft[1] · R. N. Jørgensen[1]

Published online: 13 January 2017

Abstract The concept of autonomous farming concerns automatic agricultural machines operating safely and efficiently without human intervention. In order to ensure safe autonomous operation, real-time risk detection and avoidance must be undertaken. This paper presents a flexible vehicle-mounted sensor system for recording positional and imaging data with a total of six sensors, and a full procedure for calibrating and registering all sensors. Authentic data were recorded for a case study on grass-harvesting and human safety. The paper incorporates parts of ISO 18497 (an emerging standard for safety of highly automated machinery in agriculture) related to human detection and safety. The case study investigates four different sensing technologies and is intended as a dataset to validate human safety or a human detection system in grass-harvesting. The study presents common algorithms that are able to detect humans, but struggle to handle lying or occluded humans in high grass.

Keywords Safe farming · Sensor platform · Object detection · Computer vision · ISO 18497 · Autonomous farming

Introduction

Current technology is capable of automatically navigating and operating agricultural machinery, such as tractors and harvesters, efficiently and more precisely compared to manual operation. However, a crucial deficiency in this technology concerns the safety aspects. In order for an autonomous vehicle to operate safely and be certified for

✉ P. Christiansen
pech@eng.au.dk

[1] Department of Engineering-Signal Processing, Faculty of Science and Technology, Aarhus University, Finlandsgade 22, 8200 Aarhus N, Denmark

unsupervised operation, it must perform automatic real-time risk detection and avoidance of humans in the field with high reliability (ISO 18497 2015).

Robust risk detection imposes a number of challenges for the sensor system. Varying weather and lighting conditions influence the image quality of sensing technologies in different ways, and thus no sensor is single-handedly capable of detecting objects reliably under all conditions. Active sensors such as LiDAR, and passive sensors such as RGB camera, stereo camera and thermal camera have different strengths and weaknesses concerning weather, lighting, range and resolution, and therefore a variety of these sensors are needed to cover all scenarios (Rasshofer and Gresser 2005). In addition, attitude estimation sensors such as accelerometers, gyroscopes and GPS are needed for estimating the vehicle position, velocity and orientation and for synchronizing and registering subsequent frames acquired from the imaging sensors.

Today, driver assistance systems are available for a large number of modern passenger cars, and completely autonomous vehicles operating in urban and sub-urban environments are emerging for experimental usage (Paden et al. 2016).

In the agricultural sector, a variety of machines have been operating autonomously for a decade using either precise GPS co-ordinates and/or cameras detecting structures in the field (CLAAS Steering Systems 2011; Pilarski et al. 2002). Efforts have been made to fully automate the process in a driverless solution, but safety aspects currently prevent authorization for this. In Freitas et al. (2012), Yang and Noguchi (2012) and Wei et al. (2005), human detection was performed using only a single sensor (laser scanner or stereo camera). However, multiple sensor modalities should be investigated to evaluate their ability to detect humans. For instance, the QUAD-AV project has investigated microwave radar, stereo vision, LiDAR and thermography for detecting obstacles in an agricultural context (Rouveure et al. 2012). Within the project, a detailed study of stereo vision has shown promising results on ground/non-ground classification (Reina and Milella 2012).

In urban environments, autonomous vehicles can exploit obstacles protruding from the surface. In farming operations, obstacles are commonly placed below or just above an uneven surface of crops introducing specific challenges for autonomous vehicles in agriculture. The likelihood of a human being one of these obstacles is small. However, a child or a fallen, injured or unconscious human provides a risk as these non-protruding objects have reduced mobility. To investigate these challenges, data from agricultural fields and algorithms are needed.

Human safety is addressed in ISO 18497 (an emerging standard for safety of highly automated machinery in agriculture) by defining a minimum obstacle that must be detected with an accuracy of 99.99% (ISO 18497 2015). The minimum obstacle is specified as an olive green barrel shaped object that resembles a small or seated human in green clothing (in this paper defined as an ISO-barrel).

This paper describes a flexible vehicle-mounted sensor platform targeting agricultural fields. The sensor platform records imaging data and vehicle position for a moving vehicle using three passive imaging sensors, one active sensor and two attitude/position estimation sensors. The sensor platform is designed to record simultaneous data from all sensors, thus preparing for subsequent offline processing. Offline processing and visualization of sensor data is presented to investigate the object detection potential for the different sensors. The current paper is an extended version of Christiansen et al. (2015) providing more authentic data in grass-harvesting operations and addressing human safety in more detail. An ISO-barrel was produced under the specification defined in ISO 18497. The ISO-barrel as well as humans and mannequins were placed in standing and lying positions in front of the setup to create recordings that could be used in an actual validation of a human detection system

during grass-harvesting. The extended edition also presents a full procedure for calibrating and registering all sensors using a single calibration thermal checkerboard.

Materials and methods

Sensors

An overview of the strengths and weaknesses of the selected imaging and active sensors are presented in Table 1. The qualities are evaluated individually and under various conditions. A weakness is marked with '−' and a strength is marked with '+'.

Sensor modalities refer to the information a sensor measures. In this paper, a sensor modality is either visual light, depth or heat radiation.

An *RGB camera* captures the modality of visible light. The sensor is useful for identifying the perceived objects as it provides visual characteristics such as texture, color and shape in high resolution at low cost. It is invariant to protrusion, meaning that non-protruding objects such as small animals, a fallen human or humans/animals in high crops are still visible. However, visual characteristics are affected by occlusion from crops, weather conditions (rain, fog and snow) and illumination such as dim light (night) or direct light (causing shadows). An RGB camera is not able to exploit depth information to emphasize protruded objects and the lack of depth makes the positioning of objects in 3D space difficult.

A stereo camera enables 3D imaging data (depth and color information). Depth and color information are registered and the sensor is thus able to exploit the advantages of both modalities. Depth information can be used to see protruded objects and visually camouflaged animals easily while determining the position of an object relative to the vehicle. In this way, depth-aware algorithms can abstract from the very different visual characteristics of objects (shape, color and texture) creating simple detection algorithms. Like the RGB camera, the stereo camera is sensitive to illumination and weather conditions, although the depth information is in some cases still retrievable.

Table 1 Strengths and weaknesses of sensors (Christiansen et al. 2015)

Names	RGB	RGB stereo	Thermal	LiDAR
Specification				
Range	Medium	Medium	Medium	Long
Resolution	+	+	−	−
Depth information	−	+	−	+
Heat information	−	−	+	−
Color information	+	+	−	−
Cost	Low	Medium	Medium	High
Robustness				
Light changes	−	−	+	+
Weather changes	−	−	−	+
Camouflaged objects	−	+	+	+
Protruding objects	−	+	−	+
Non-protruding objects	+	−	+	−

A *thermal camera* is an imaging sensor that captures heat radiation represented by intensities (temperatures) to form a monochromatic image. A thermal camera perceives objects of distinct temperatures, making it ideal for detecting living objects in temperate and colder climates, and even in foggy weather (Serrano-Cuerda et al. 2014). A key ability is that the sensed data are unaffected by non-protruded or visually camouflaged animals and that the distinctness of living objects becomes more apparent at night. However, these capabilities are much affected by the ambient temperature as living objects become indistinct when the temperature difference between the objects and background becomes small (Serrano-Cuerda et al. 2014). The cost of a well-performing and high resolution thermal camera is very high, but low cost cameras are emerging. Object recognition capabilities are low due to a limited resolution and limited visual characteristics.

A *LiDAR* measures range data to a set of surrounding points and generates a point cloud where each point is represented by a 3D position and reflection intensity. The LiDAR is a high cost sensor, but has dropped significantly in price in recent years. Compared to a stereo camera, the LiDAR provides very exact depth information at greater range and some models can capture in 360° horizontally. It is invariant to illumination, temperature and camouflage. The lack of visual and thermal information makes recognition of objects difficult and non-protruding objects are almost or fully undetectable.

Physical design

The sensor platform consisted of seven sensors and a controller mounted on a common rack of 2 m by 0.8 m in size. The left side of Fig. 1 shows the rack mounted on a tractor and the right side shows the physical placement of sensors. A standard A-frame was mounted at the bottom of the rack to enable easy mounting on tractors. The category 1 A-frame was mounted with dampers for absorbing internal engine vibrations from the vehicle to reduce the amount of mechanical noise acting on the sensors. The horizontal profile in the middle was adjustable in height and angle such that the imaging sensors could be oriented in a downward angle depending on the vehicle height. The LiDAR was placed above the sensor frame to minimize view obstructions for the sensor. The rack allowed sensors to be placed roughly 2 m above ground to provide a more downward view into the crop to better detect hidden obstacles. Placing sensors on top of the tractor would provide a similar downward view. However, the tall rack and the A-frame allowed the sensors to be

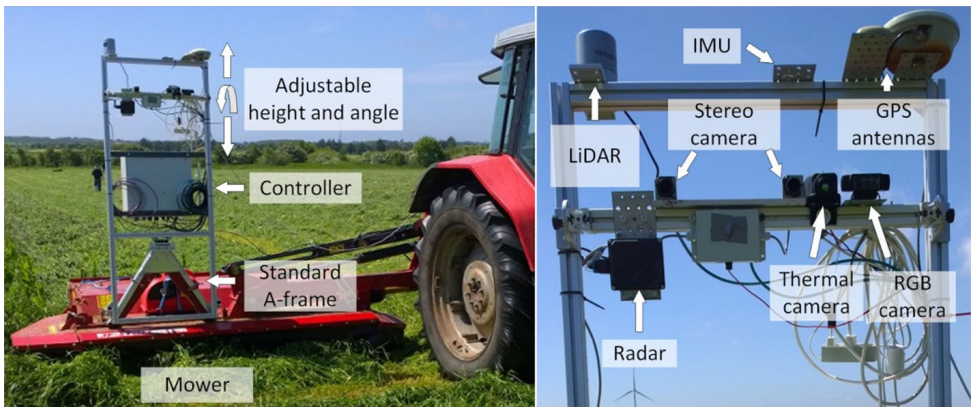


Fig. 1 *Left* sensor frame including controller. *Right* sensors on the sensor platform

easily swapped to another tractor, an all-terrain vehicle or directly on a ground socket while keeping the downward view under data acquisition.

A Logitech HD Pro C920 from Logitech (Silicon Valley, USA) webcam providing 1 920 × 1 080 pixels at 30 fps was used as the RGB camera. The stereo camera was composed of two hardware synchronized Flea3/FL3-GE-28S4C-C cameras from Point Grey (Richmond, Canada) with global shutter and 1 928 × 1 448 pixels at 15 fps. The thermal camera was a shutterless HawkVision analog thermal camera from Tonbo Imaging (Bangalore, India) providing 640 × 480 pixels at 25 fps (interlaced). The HDL-32E LiDAR from Velodyne (Morgan Hill, USA) was a 32-beam laser scanner providing 70 000 points at 10 Hz with 1–100 m range. Figure 1 shows an automotive Delphi ESR 64-target radar from Delphi (Washington, DC, USA) not addressed in the current paper as it was intended for detecting pieces of metal and not humans. The GPS was an AG GPS361 real time kinematic (RTK) GPS from Trimble (Sunnyvale, USA) enhancing the precision of GPS up to centimeter-level accuracy. The IMU was a VN-100 from Vectornav (Dallas, USA) providing synchronized three-axis accelerometers, gyros, magnetometers and a barometric pressure sensor. The data-collecting controller was an Innovation Robotech Controller 701 from Conpleks (Struer, Denmark). It is an embedded computer with external interfaces for all sensors that using ROS-middleware (robot operating system) to easily integrate them into a common framework.

System architecture

Figure 2 further illustrates the connections between the sensors and the controller. In ROS, each sensor was given its own node (an executable file) that was responsible for publishing one or more topics. For instance, the IMU had its own node including hardware-specific drivers, and it published different topics related to the readings of the accelerometer, the gyroscopes and the magnetometers. For each topic, the node could send messages containing sensor data whenever a new sensor-reading was available. Each node was connected to the ROS Master which handled interactions between nodes and supplied all messages with exact timestamps. Using the rosbag package, a recording of all desired topics (and all associated messages) to a single rosbag data-file could be obtained. A JavaScript browser interface was developed to easily monitor and record specific sensors, and enabled the platform to be controlled through Wi-Fi using a mobile phone, tablet or computer.

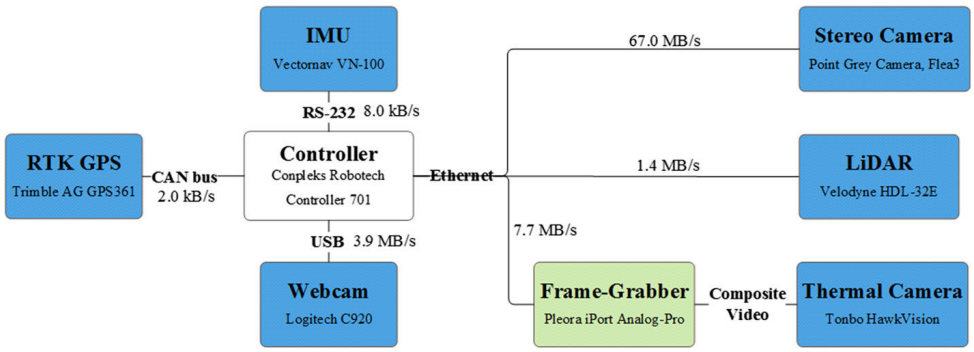


Fig. 2 System overview illustrating bandwidths and interfaces for sensors

Data

Data were collected on a grass field of roughly 7.5 ha near Lem in Denmark (latitude 56.059679° N, longitude 8.368701° E) in the beginning of June 2015. To get authentic data, sensors were mounted to a tractor working in a normal grass-harvesting operation. In operation, obstacles were placed in the trajectory of the tractor to simulate collision hazards. For each obstacle, the tractor approached the object and stopped just before collision. To enable some form of reproducibility and to ensure safety, standing/lying adult and child mannequins were used instead of real humans in the field. To incorporate safety standards, the ISO-barrel was also used. Finally, the mower was turned off and two recordings with real humans were captured. Obstacles from the data are presented in Fig. 3.

In Fig. 4, obstacle positions and the tractor route (divided into laps) are presented, where lap 17 and 18 contained real human obstacles.

Fig. 3 Two real humans, three mannequins and the ISO-barrel

Fig. 4 Tractor route (*lines*), barrel (*circles*), kid mannequin (*diamonds*), adult mannequin (*squares*), well (*stars*) and lap starting point (*small dots*)

Registration of sensors

Registration or sensor fusion is essential for a multi-sensor system to merge and exploit information from all sensors. Registration in multiple modalities is non-trivial and can be handled in different ways (Bahnsen 2013; Zhao and Cheung 2014; Krotosky and Trivedi 2007). In particular, Bahnsen (2013) provided a coherent description of registration methods and the complications for registering different modalities, when objects are not positioned at the same distance.

In this work, common camera and sensor view geometry combined with depth information from the stereo camera were used to project points between sensor frames (Johnson and Bajcsy 2008). Such projections require the *intrinsic* parameters to calibrate cameras individually and *extrinsic* parameters—describing the inter-displacement of sensors—to finalize registration. The inter-displacement between LiDAR and stereo camera was found by matching the two point clouds using the iterative closest point algorithm (Zhang 1994).

The stereo camera and the webcam was calibrated individually using a normal checkerboard and MATLABs computer vision: calibration tool (2015). The calibration tool was able to detect checkerboards, calibrate cameras, map checkerboard to 3D position automatically and, for the stereo camera, find the inter-displacement between the left and right camera. For the webcam, the extrinsic parameters was determined by finding the transformation that matched corresponding 3D checkerboards to, in this setup, the left stereo camera. However, to calibrate and find inter-displacement between thermal and RGB cameras using a traditional and automated calibration tool, the checkerboard must be visible in both modalities. Therefore, a custom-made visual–thermal checkerboard is proposed.

Visual–thermal registration

A normal checkerboard exposed to sunlight can be used to perform thermal–visual registration as black absorbs more energy than white areas. However, the quality of the thermal calibration is dependent on weather conditions, and heat/energy is transferred in the material between black and white areas making square transitions indistinct.

A registration/calibration board was therefore developed using a circuit board with copper squares as shown in Fig. 5 (left).

The circuit board was heated by attaching an aluminum plate mounted with impact resistors on the backside of the board as in Fig. 5 (right). The 60 resistors delivered 216 W of heat using a 12 V car battery. Copper has a low emissivity coefficient, which effectively made the material work as a reflector. Thus, the non-copper squares emitted heat radiation

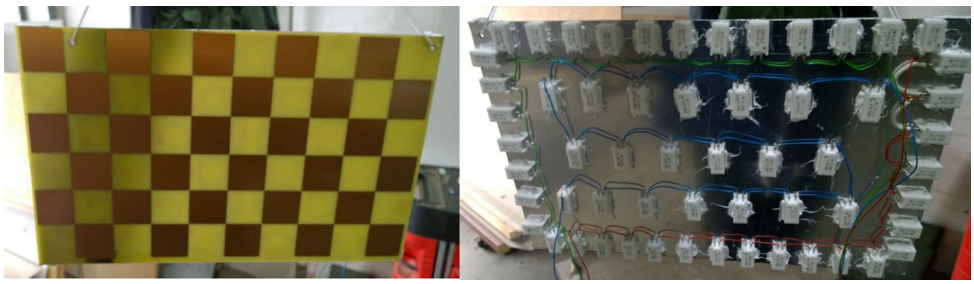


Fig. 5 Front and back side of registration board

from the heated circuit board, and the copper areas reflected heat of the surroundings, giving a distinct transition between copper and non-copper squares.

The thermal checkerboard would, in a normalized thermal image, resemble a traditional black and white checkerboard as presented in Fig. 6 (right). The thermal camera was then calibrated using traditional and automated calibration tools.

The thermal checkerboard did not, for RGB images, resemble a traditional black and white checkerboard as depicted in Fig. 6 (left). Thus, calibrations tools could not be applied directly. To use RGB images, a MATLAB script was developed to enable a user to mark an area inside the checkerboard. This area was then cropped and converted to the LAB color space. Automatically, the A and B channels were modeled into two clusters using a Gaussian mixture model (McLachlan and Basford 1988). Copper and non-copper areas were separated into two individual clusters. The posterior probability of each pixel belonging to a specific cluster generated a gray-scaled image that made the registration board resemble a traditional black and white checkerboard, see Fig. 6 (mid).

Converting RGB images, enabled all camera sensors to be calibrated and registered using only the proposed registration board. However, the procedure required the user to place a rectangular area inside the checkerboard for each image. In Fig. 7, the detected boards and the inter-displacement of sensors are visualized.

In Fig. 8 (middle left), two humans are annotated in the left stereo camera and projected to the stereo point cloud in Fig. 8 (top). The distance to objects inside the annotation was determined using the median distance of pixels inside the bounding box. The bounding box was then defined as four points in the stereo point cloud that could be projected to other

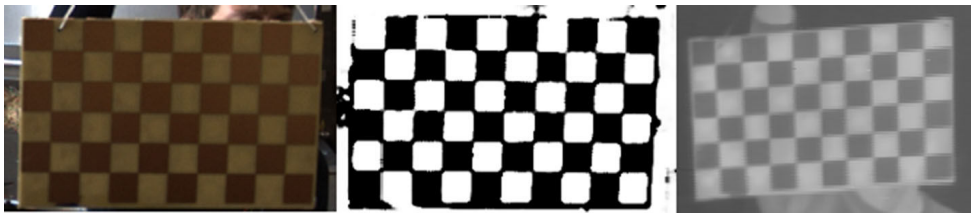


Fig. 6 The registration board (*left*) is transformed into a "classic" checkerboard (*mid*) using a Gaussian mixture model. Thermal image of the registration board (*right*)

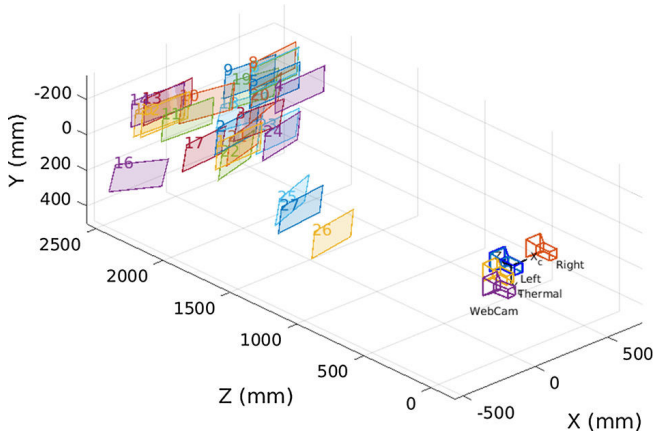


Fig. 7 Registration board placements (numbered *1–25*) and inter-displacement of sensors

Fig. 8 Annotations in the *left image* are projected onto the stereo point cloud (*top*). These annotations are then projected to the right and left stereo camera (*middle left* and *right*), the webcam (*bottom left*) and the thermal camera (*bottom right*)

sensor frames as in Fig. 8. To make a more exact registration of sensors, the registration board should be placed at a broader range of distances from the cameras.

A more quantitative evaluation of the visual–thermal registration is presented in "Appendix: Thermal–visual registration and evaluation" section.

Signal processing

To provide an initial qualitative validation of detection performance of the different sensors in an agricultural environment, preliminary tests using different object detection algorithms have been carried out on the sensors.

Using only an RGB camera for detecting all possible obstacles in the field is complex and difficult and requires a very large dataset with many representations of each object. Constraining detection to only humans provided a more realistic case in this preliminary study. The RGB camera images were therefore processed using a state-of-the-art pedestrian detection algorithm (Dollar et al. 2010).

After stereo camera calibration (Zhang 2000), a point cloud could be generated for each stereo image pair. For both stereo and LiDAR, the same algorithm was used to better compare sensors. A ground plane was estimated on the acquired point cloud using the RANSAC algorithm (Fischler and Bolles 1981). Protruding objects were visualized by determining the height of points relative to the estimated ground plane.

The thermal camera images were processed by thresholding the (temperature-related) intensities by a constant value above the median intensity of the image (Christiansen et al. 2014). Subsequent connected components analysis was used for extracting only components that exceeded a certain area.

Results and discussion

An initial validation of detection algorithms is presented in four scenarios. The first scenario is humans of different sizes, appearances and postures similar to Christiansen et al. (2015) in low grass. Scenarios 2–4 are, respectively, a barrel, a lying child mannequin and a sitting human in high-grass taken from the above described data.

Figure 9 depicts the human detection performance evaluated at single, synchronized frames for the RGB camera, the stereo camera and the LiDAR. At the top left, the RGB camera is shown with bounding boxes indicating results of the pedestrian detection algorithm. In the top middle, the disparity map of the stereo camera is shown and, at the top right, a protrusion map indicating objects that protrude from the ground plane is visualized. At the bottom left, the thermal camera is shown with overlaid thresholded components and, at the bottom right, the LiDAR data are visualized with a ground plane and protruding points.

Using only single frames, pedestrian detection applied to the RGB camera failed to detect all humans in the image. Problems concerning occlusion and humans seen from the side or from behind have been observed. However, utilizing a sequence of frames would greatly improve detection performance, as the algorithm most often failed for just a single frame and not for an entire sequence of frames. The stereo camera performed well for detecting humans that protruded from the ground plane. However, the algorithm assumed a certain level of protrusion in order to detect an object. The thermal camera detected all humans when their faces were visible. However, potential problems concern well-insulated clothes that cover an entire body and warm weather where temperature differences are much smaller than in the present recording. The LiDAR detected most humans robustly when they protruded significantly from the ground.

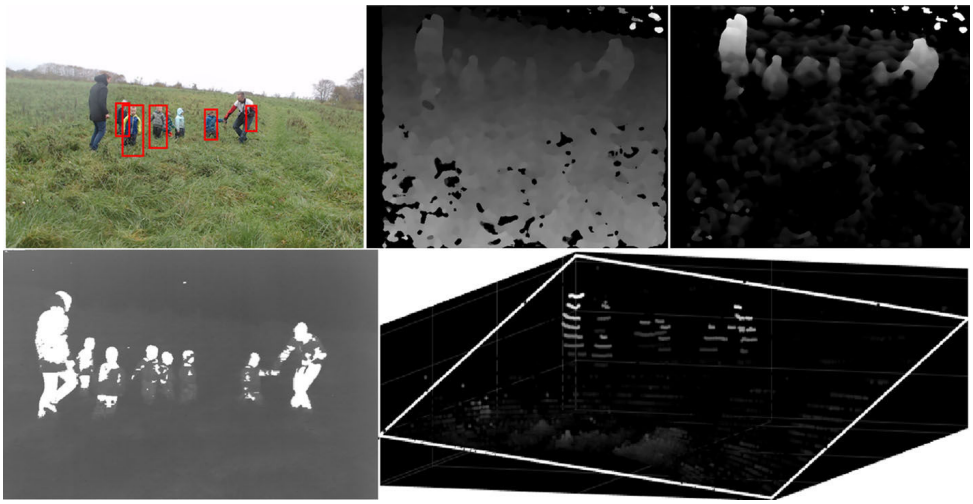


Fig. 9 Human detection. RGB (*top left*), stereo camera disparity map (*top middle*), stereo camera protrusion map (*top right*), thermal camera (*bottom left*), LiDAR (*bottom right*; Christiansen et al. 2015)

Figure 10 depicts three cropped scenarios in high grass with respectively a barrel, a lying child mannequin and a sitting human. The pedestrian detector was able to detect the sitting human as the face and torso were upright and visible. To detect the lying

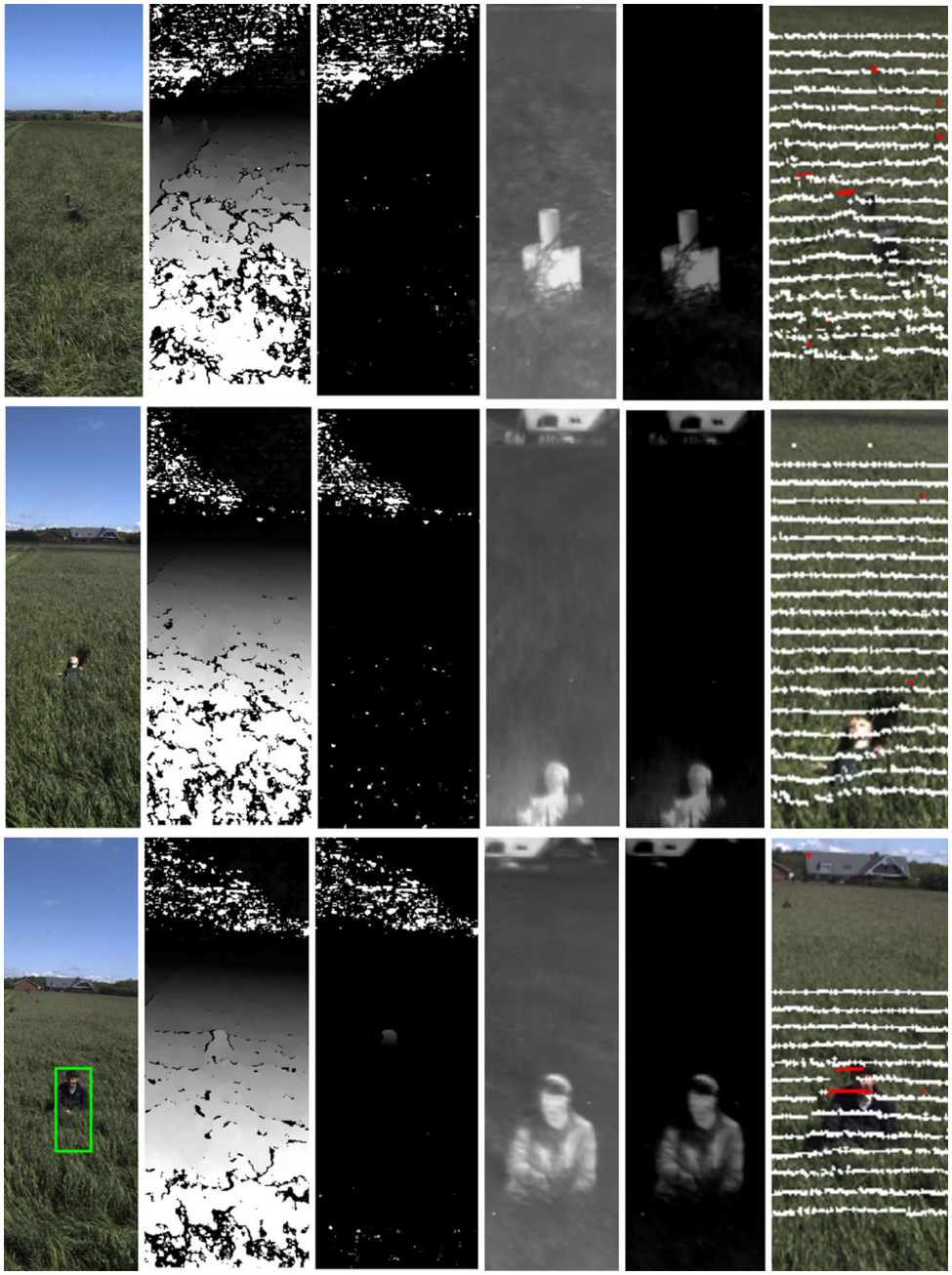


Fig. 10 The three *rows* show respectively a barrel, a lying child mannequin and a sitting human. The *columns* show respectively pedestrian detections, a disparity map from stereo imaging, an object height map based on this, the thermal signature, thermal signature after subtracting the median temperature of the *bottom half* of the image, and the LiDAR projected onto the left stereo camera, where points protruding from the surface by more than 0.25 m are visualized

mannequin, the detector needs to be trained on new data showing humans in similar scenarios. However, the given detector had limited capacity in terms of detecting objects with huge inter-class variation. In the high-grass case, there was a limited reliability of the stereo point cloud which impacted detection performance such that only the sitting human and not the barrel were visible. Exploiting also visual information from the stereo camera should be utilized to improve performance. The LiDAR was more reliable and was able to visualize that both the sitting human and the barrel protruded. The thermal camera achieved robust and reliable detection performance. In scenarios 2–4, all sensors apart from the thermal camera had problems with high grass/crop, presenting a specific challenge that should be addressed in agriculture. The thermal camera will undoubtedly be significantly worse on a warm and sunny day as experienced by Steen et al. (2012) and Serrano-Cuerda et al. (2014). A single sensor is therefore insufficient for detecting all objects reliably, invariant of temperature and lighting changes.

Conclusions

A flexible vehicle-mounted sensor platform was developed for capturing time-stamped data in the agricultural domain using imaging sensors (RGB, thermal and stereo camera), an active sensor (LiDAR) and attitude estimation sensors (RTK GPS and IMU). A registration board was proposed to provide a simple tool for calibrating and registering all sensors in the setup using a single registration board. Authentic data in an actual high grass harvesting operation with a specific focus on human detection were recorded, and an initial evaluation of the potential of different sensor modalities for detecting standing and lying humans including an ISO-barrel was given. Using a common pedestrian detection algorithm, an RGB camera was able to detect upright humans, but degraded rapidly in performance for more complex scenarios. The depth aware sensors (LiDAR and stereo camera) were efficient for detecting objects that protruded significantly above the ground. The LiDAR was invariant towards changing weather and lighting conditions, whereas the stereo camera had the highest resolution making it useful for classifying objects. The thermal camera showed great capabilities in the captured dataset as it was able to detect objects of distinct temperature using a simple procedure that worked well for humans regardless of posture. However, the detection would be much more complicated in environments of higher temperature, where the heat signatures of living objects become indistinct.

The authenticity of the data enabled an initial validation of a human detection system using multiple sensors in a high grass harvesting operation. However, the above arguments and the case study concludes that the use of multiple modalities, more complicated procedures and a fusion of the different modalities is required to achieve robust human detection in high grass harvesting.

Acknowledgements This research is sponsored by the Innovation Fund Denmark as part of the Project "SAFE - Safer Autonomous Farming Equipment" (Project No. 16-2014-0) and "Multi-sensor system for ensuring ethical and efficient crop production" (Project No. 155-2013-6).

Appendix: Thermal–visual registration and evaluation

First a total of 47 thermal and stereo synchronized images were selected from a single calibration recording. For each image, a rectangle area inside the checkerboard was marked manually to specify an image cropping, see Fig. 11. For RGB images, the cropped image was converted to the LAB color space and a Gaussian mixture model separated the pixels into two clusters (copper and non-copper areas). The posterior probability of belonging to one of the Gaussian clusters was determined for all pixels in the original image, see Fig. 12. For thermal images, the cropped image was normalized—transforming pixel

Fig. 11 Image example and a manually marked *rectangle*

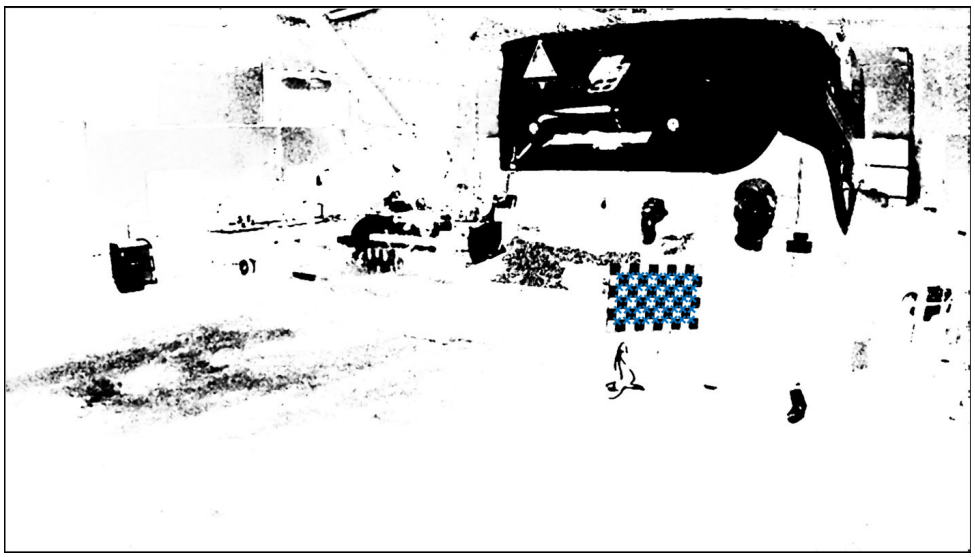


Fig. 12 Posterior probability of belonging to one of the Gaussian clusters for all pixels in the image example. Checkerboard detection is marked with *blue crosses* (Color figure online)

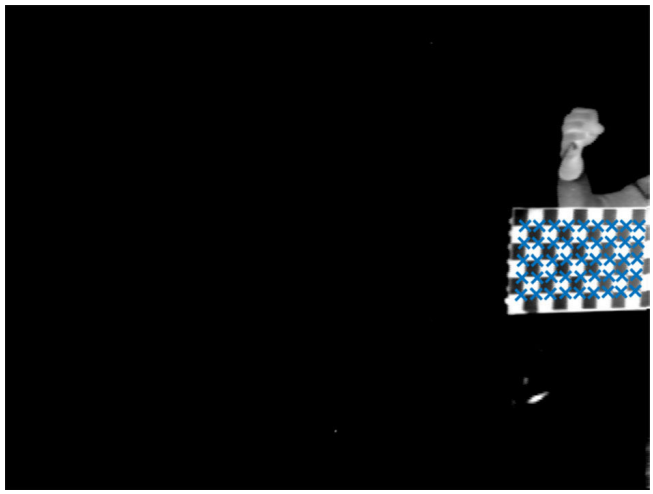


Fig. 13 Thermal image is normalized relative to the checkerboard. Checkerboard detection is marked with *blue crosses* (Color figure online)

values in the range [0 1] by shifting and scaling. The same normalization was applied to the whole thermal image, see Fig. 13. The MATLAB calibration toolbox was able to automatically detect checkerboards of the transformed RGB and thermal images. The calibration toolbox was able to detect the checkerboard in 27 and 43 out of the 45 images for respectively stereo and thermal images. The 27 stereo images were used for calibrating the intrinsic and extrinsic parameters of the stereo camera. The 43 thermal images were used for determining the intrinsic parameters of the thermal camera.

In 25 out of 47 synchronized images, the checkerboard was successfully detected by the MATLAB calibration toolbox for both RGB and thermal images. The toolbox estimated the 3D position of the checkerboard in all 25 images for each camera. The extrinsic parameters of the thermal camera were determined as the least square rigid transformation that mapped the estimated checkerboards from the left RGB camera to the thermal camera (in 3D).

The registration was evaluated on the 25 images to provide a quantitative evaluation of the thermal–visual registration. The camera calibration for the left stereo camera

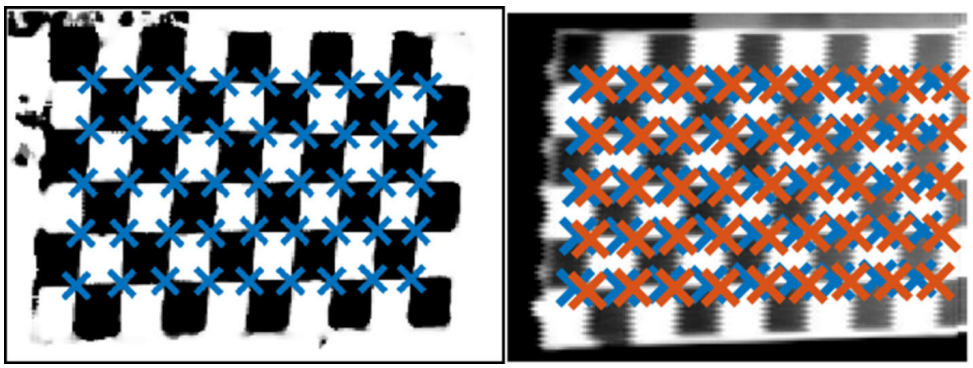


Fig. 14 Zoomed images. *Blue crosses* mark *corners* detected by the MATLAB calibration toolbox for both an RGB image (*left*) and a thermal image (*right*). The *red crosses* (*left*) show how 3D points are projected to the thermal camera (Color figure online)

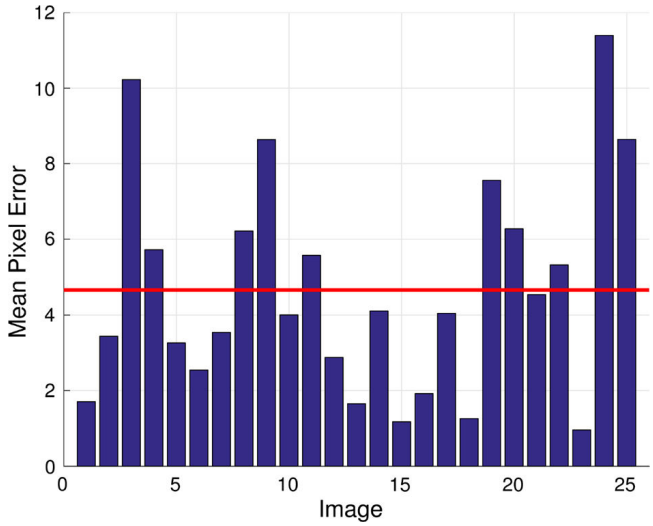


Fig. 15 The mean pixel error for 25 images (*blue bars*) and the mean pixel error across all images (*red line*) (Color figure online)

estimated—as already described—the checkerboard positions in 3D. These positions were then projected to the thermal image using the estimated extrinsic and intrinsic parameters of the thermal camera, see Fig. 4 (right).

The error was determined as the distance between the detected checkerboard and the projected 3D positions. Figure 15 shows the mean pixel error for each of the 25 images and the mean pixel error across all images on 4.66 pixels. The image example used in Figs. 11, 12, 13, and 14 is image 21 with a mean pixel error close to the mean pixel error across all images.

References

Bahnsen, C. (2013). Thermal-visible-depth image registration. Unpublished Master Thesis, Aalborg University, Aalborg, Denmark.

Christiansen, P., Kragh, M., Steen, K. A., Karstoft, H., & Jørgensen, R. N. (2015). Advanced sensor platform for human detection and protection in autonomous farming. *Precision Agriculture, 15,* 291–298.

Christiansen, P., Steen, K. A., Jørgensen, R. N., & Karstoft, H. (2014). Automated detection and recognition of wildlife using thermal cameras. *Sensors, 14*(8), 13778–13793.

CLAAS Steering Systems. (2011). *Tracking control optimisation*. Retrieved 2016, 26 September from http://claas.via-us.co.uk/booklets/gps-steering-systems/download.

Dollar, P., Belongie, S., & Perona, P. (2010). The fastest pedestrian detector in the west. In F. Labrosse, R. Zwiggelaar, Y. Liu & B. Tiddeman (Eds.), *Proceedings of the British machine vision conference* 2010 (pp 68.1–68.11). BMVA Press, Durham University, UK.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM, 24*(6), 381–395.

Freitas, G., Hamner, B., Bergerman, M., & Singh, S. (2012). A practical obstacle detection system for autonomous orchard vehicles. In *2012 IEEE/RSJ international conference on intelligent robots and systems* (pp 3391–3398).

ISO/DIS 18497:2015: *Agricultural and forestry tractors and self-propelled machinery—Safety of highly automated machinery*. Retrieved 2016, 26 September from https://drive.google.com/file/d/0B1ilODNTH9nzRUV2N0JzbklubFU/view.

Johnson, M. J., & Bajcsy, P. (2008). Integration of thermal and visible imagery for robust foreground detection in tele-immersive spaces. In P. Solbrig (Ed.), *Proceedings of the 11th international conference on information fusion* (pp. 1265–1272). Piscataway, USA: IEEE.

Krotosky, S. J., & Trivedi, M. M. (2007). Mutual information based registration of multimodal stereo videos for person tracking. *Computer Vision and Image Understanding, 106*(2–3), 270–287.

McLachlan, G. J., & Basford, K. E. (1988). Mixture models: Inference and applications to clustering. In *Statistics*: *textbooks and monographs*. New York, USA: Dekker.

Paden, B., Cáp, M., Yong, Z. S., Yershov, D., & Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles, 1*(1), 33–55. arXiv:cs.CV/1604.07446v1.

Pilarski, T., Happold, M., Pangels, H., Ollis, M., Fitzpatrick, K., & Stentz, A. (2002). The Demeter System for automated harvesting. *Autonomous Robots, 13,* 9–20.

Rasshofer, R. H., & Gresser, K. (2005). Automotive radar and lidar systems for next generation driver assistance functions. *Advances in Radio Science, 3,* 205–209.

Reina, G., & Milella, A. (2012). Towards autonomous agriculture: Automatic ground detection using trinocular stereovision. *Sensors, 12*(12), 12405–12423.

Rouveure, R., Nielsen, M., & Petersen, A. (2012). The QUAD-AV Project: Multi-sensory approach for obstacle detection in agricultural autonomous robotics. In *International conference of agricultural engineering*. Valencia, Spain: EurAgEng.

Serrano-Cuerda, J., Fernández-Caballero, A., & López, M. (2014). Selection of a visible-light vs. thermal infrared sensor in dynamic environments based on confidence measures. *Applied Sciences, 4*(3), 331–350.

Steen, K. A., Villa-Henriksen, A., Therkildsen, O. R., & Green, O. (2012). Automatic detection of animals in mowing operations using thermal cameras. *Sensors, 12*(6), 7587–7597.

The MathWorks, Inc. (2015). *MATLAB and computer vision system toolbox*. Natick, MA, USA: The MathWorks, Inc.

Wei, J., Rovira-Mas, F., Reid, J. F., & Han, S. (2005). Obstacle detection using stereo vision to enhance safety of autonomous machines. *Transactions of the ASAE, 48*(6), 2389–2397. doi:10.13031/2013.20078.

Yang, L., & Noguchi, N. (2012). Human detection for a robot tractor using omni-directional stereo vision. *Computers and Electronics in Agriculture, 89*, 116–125.

Zhang, Z. (1994). Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision, 13*(2), 119–152.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*(11), 1330–1334.

Zhao, J., & Cheung, S. S. (2014). Human segmentation by geometrically fusing visible-light and thermal imageries. *Multimedia Tools and Applications, 76*(1), 7361–7389.

Paper 3

FieldSAFE: Dataset for Obstacle Detection in Agriculture

Mikkel Fly Kragh, Peter Christiansen, Morten Stigaard Laursen, Morten Larsen, Kim Arild Steen, Ole Green, Henrik Karstoft, and Rasmus Nyholm Jørgensen

Peer reviewed
Accepted for publication in MDPI Sensors, Special Issue: Sensors in Agriculture, November 2017

sensors

MDPI

Article

FieldSAFE: Dataset for Obstacle Detection in Agriculture

Mikkel Fly Kragh [1,*,†], Peter Christiansen [1,†], Morten Stigaard Laursen [1], Morten Larsen [2], Kim Arild Steen [3], Ole Green [3], Henrik Karstoft [1] and Rasmus Nyholm Jørgensen [1]

1 Department of Engineering, Aarhus University, Aarhus N 8200, Denmark; repetepc@gmail.com (P.C.); msl@eng.au.dk (M.S.L.); hka@eng.au.dk (H.K.); rnj@eng.au.dk (R.N.J.)
2 Conpleks Innovation ApS, Struer 7600, Denmark; morten.larsen@conpleks.com
3 AgroIntelli, Aarhus N 8200, Denmark; kas@agrointelli.com (K.A.S.); olg@agrointelli.com (O.G.)
* Correspondence: mkha@eng.au.dk; Tel.: +45-5176-1455
† These authors contributed equally to this work.

Received: 28 September 2017; Accepted: 7 November 2017; Published: 9 November 2017

Abstract: In this paper, we present a multi-modal dataset for obstacle detection in agriculture. The dataset comprises approximately 2 h of raw sensor data from a tractor-mounted sensor system in a grass mowing scenario in Denmark, October 2016. Sensing modalities include stereo camera, thermal camera, web camera, 360° camera, LiDAR and radar, while precise localization is available from fused IMU and GNSS. Both static and moving obstacles are present, including humans, mannequin dolls, rocks, barrels, buildings, vehicles and vegetation. All obstacles have ground truth object labels and geographic coordinates.

Keywords: dataset; agriculture; obstacle detection; computer vision; cameras; stereo imaging; thermal imaging; LiDAR; radar; object tracking

1. Introduction

For the past few decades, precision agriculture has revolutionized agricultural production systems. Part of the development has focused on robotic automation, to optimize workflow and minimize manual labor. Today, technology is available to automatically steer farming vehicles such as tractors and harvesters along predefined paths using accurate global navigation satellite systems (GNSS) [1]. However, a human operator is still needed to monitor the surroundings and intervene when potential obstacles appear in front of the vehicle to ensure safety.

In order to completely eliminate the need for a human operator, autonomous farming vehicles need to operate both efficiently and safely without any human intervention. A safety system must perform robust obstacle detection and avoidance in real time with high reliability. Additionally, multiple sensing modalities must complement each other in order to handle a wide range of changes in illumination and weather conditions.

A technological advancement like this requires extensive research and experiments to investigate combinations of sensors, detection algorithms and fusion strategies. Currently, a few publicly known commercial R&D projects exist within companies that seek to investigate the concept [2–4]. In scientific research, projects investigating autonomous agricultural vehicles and sensor suites have existed since 1997, where a simple vision-based anomaly detector was proposed [5]. Since then, a number of research projects has experimented with obstacle detection and sensor fusion [6–14]. However, to our knowledge, no public platforms or datasets are available that address the important issues of multi-modal obstacle detection in an agricultural environment.

Within urban autonomous driving, a number of datasets has recently been made publicly available. Udacity's Self-Driving Car Engineer Nanodegree program has given rise to multiple challenge datasets

including stereo camera, LiDAR and localization data [15–17]. A few research institutions such as the University of Surrey [18], Linköping University [19], Oxford [20], and Virginia Tech [21] have published similar datasets. Most of the above cases, however, only address behavioral cloning, such that ground truth data are only available for control actions of the vehicles. No information is thus available for potential obstacles and their location in front of the vehicles.

The KITTI dataset [22], however, addresses these issues with object annotations in both 2D and 3D. Today, it is the de facto standard for benchmarking both single- and multi-modality object detection and recognition systems for autonomous driving. The dataset includes high-resolution grayscale and color stereo cameras, a LiDAR and fused GNSS/IMU sensor data.

Focusing specifically on image data, an even larger selection of datasets is available with annotations of typical object categories such as cars, pedestrians and bicycles. Annotations of cars are often represented by bounding boxes [23,24]. However, pixel-level annotation or semantic segmentation has the advantage of being able to capture all objects, regardless of their shape and orientation. Some of these are synthetically-generated images using computer graphic engines that are automatically annotated [25,26], whereas others are natural images that are manually labeled [27,28].

In agriculture, only a few similar datasets are publicly available. The Marulan Datasets [29] provide multi-sensor data from various rural environments and include a large variety of challenging environmental conditions such as dust, smoke and rain. However, the datasets focus on static environments and only contain a few humans occasionally walking around with no ground truth data available. Recently, the National Robotics Engineering Center (NREC) Agricultural Person-Detection Dataset [30] was made publicly available. It contains labeled image sequences of humans in orange and apple orchards acquired with moving sensing platforms. The dataset is ideal for pushing research on pedestrian detection in agricultural environments, but only includes a single modality (stereo vision). Therefore, a need still exists for an object detection dataset that allows for investigation of sensor combinations, multi-modal detection algorithms and fusion strategies.

While some similarities between autonomous urban driving and autonomous farming are present, essential differences exist. An agricultural environment is often unstructured or semi-structured, whereas urban driving involves planar surfaces, often accompanied by lane lines and traffic signs. Further, distinction between traversable, non-traversable and processable terrain is often necessary in an agricultural context such as grass mowing, weed spraying or harvesting. Here, tall grass or high crops protruding from the ground may actually be traversable and processable, whereas ordinary object categories such as humans, animals and vehicles are not. In urban driving, however, a simplified traversable/non-traversable representation is common, as all protruding objects are typically regarded as obstacles. Therefore, sensing modalities and detection algorithms that work well in urban driving do not necessarily work well in an agricultural setting. Ground plane assumptions common for 3D sensors may break down when applied on rough terrain or high grass. Additionally, vision-based detection algorithms may fail when faced with visual ambiguous information from, e.g., animals that are camouflaged to resemble the appearance of vegetation in a natural environment.

In this paper, we present a flexible, multi-modal sensing platform and a dataset called FieldSAFE for obstacle detection in agriculture. The platform is mounted on a tractor and includes stereo camera, thermal camera, web camera, 360° camera, LiDAR and radar. Precise localization is further available from fused IMU and GNSS. The dataset includes approximately 2 h of recordings from a grass mowing scenario in Denmark, October 2016. Both static and moving obstacles are present including humans, mannequin dolls, rocks, barrels, buildings, vehicles and vegetation. Ground truth positions of all obstacles were recorded with a drone during operation and have subsequently been manually labeled and synchronized with all sensor data. Figure 1 illustrates an overview of the dataset including recording platform, available sensors, and ground truth data obtained from drone recordings. Table 1 compares our proposed dataset to existing datasets in robotics and agriculture. The dataset supports research into object detection and classification, object tracking, sensor fusion, localization and mapping. It can be downloaded from https://vision.eng.au.dk/fieldsafe/.

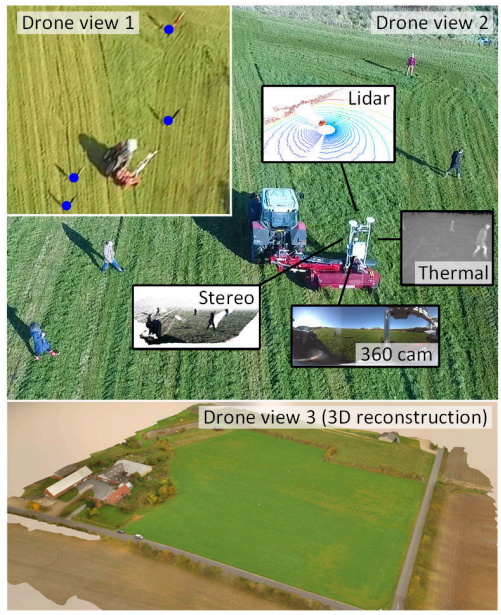


Figure 1. Recording platform surrounded by static and moving obstacles. Multiple drone views record the exact position of obstacles, while the recording platform records local sensor data.

Table 1. Comparison to existing datasets in robotics and agriculture.

Dataset	Environment	Length	Localization	Sensors	Obstacles	Annotations
KITTI [22]	urban	6 h	✓	stereo camera, LiDAR	cars, trucks, trams, pedestrians, cyclists	2D + 3D bounding boxes
Oxford [20]	urban	1000 km	✓	stereo camera, LiDARs, color cameras	cars, trucks, pedestrians, cyclists	none
Marulan [29]	rural	2 h	✓	lasers, radar, color camera, infra-red camera	humans, box, poles, bricks, vegetation	none
NREC [30]	orchards	8 h	✓	stereo camera	humans, vegetation	bounding boxes (only humans)
FieldSAFE (ours)	grass field	2 h	✓	stereo camera, web camera, thermal camera, 360° camera, LiDAR, radar	humans, mannequins, rocks, barrels, buildings, vehicles, vegetation	GPS position and labels

2. Sensor Setup

Figure 2 shows the recording platform mounted on a tractor during grass mowing. The platform was mounted on an A-frame (standard in agriculture) with dampers for absorbing internal engine vibrations from the vehicle. The platform consists of the exteroceptive sensors listed in Table 2, the proprioceptive sensors listed in Table 3 and a Conpleks Robotech Controller 701 used for data collection with the Robot Operating System (ROS) [31]. The stereo camera provides a timestamped left (color) and right (grayscale) raw and rectified image pair along with an on-device calculated depth image. Post-processing methods are further available for generating colored 3D point clouds. The web camera and 360° camera provide timestamped compressed color images. The thermal camera provides a raw grayscale image that allows for conversion to absolute temperatures. The LiDAR provides raw distance measurements and calibrated reflectivities for each of the 32 laser beams. Post-processing methods are available for generating 3D point clouds. The radar provides raw CAN messages with up to 16 processed radar detections per frame from mid- and long-range modes simultaneously. The radar detections consist of range measurements, azimuth angles and amplitudes. ROS topics and data

formats for each sensor are available on the FieldSAFE website. Code examples for data visualization are further available on the corresponding git repository.

Figure 2. Recording platform.

Table 2. Exteroceptive sensors.

Sensor	Model	Resolution	FOV	Range	Acquisition Rate
Stereo camera	Multisense S21 CMV2000	1024 × 544	85° × 50°	1.5–50 m	10 fps
Web camera	Logitech HD Pro C920	1920 × 1080	70° × 43°	-	20 fps
360° camera	Giroptic 360cam	2048 × 833	360° × 292°	-	30 fps
Thermal camera	Flir A65, 13 mm lens	640 × 512	45° × 37°	-	30 fps
LiDAR	Velodyne HDL-32E	2172 × 32	360° × 40°	1–100 m	10 fps
Radar	Delphi ESR	16 targets/frame	90° × 4.2°	0–60 m	20 fps
		16 targets/frame	20° × 4.2°	0–174 m	20 fps

Table 3. Proprioceptive sensors.

Sensor	Model	Description	Acquisition Rate
GPS	Trimble BD982 GNSS	Dual antenna RTK GNSS system. Measures position and horizontal heading of the platform.	20 Hz
IMU	Vectornav VN-100	Measures acceleration, angular velocity, magnetic field and barometric pressure.	50 Hz

The proprioceptive sensors include GPS and IMU. An extended Kalman filter has been setup to provide global localization by fusing GPS and IMU with the robot_localization package [32] available in ROS. The localization code and resulting pose information are available along with the raw localization data.

Figure 3 illustrates a synchronized pair of frames from stereo camera, 360° camera, web camera, thermal camera, LiDAR and radar.

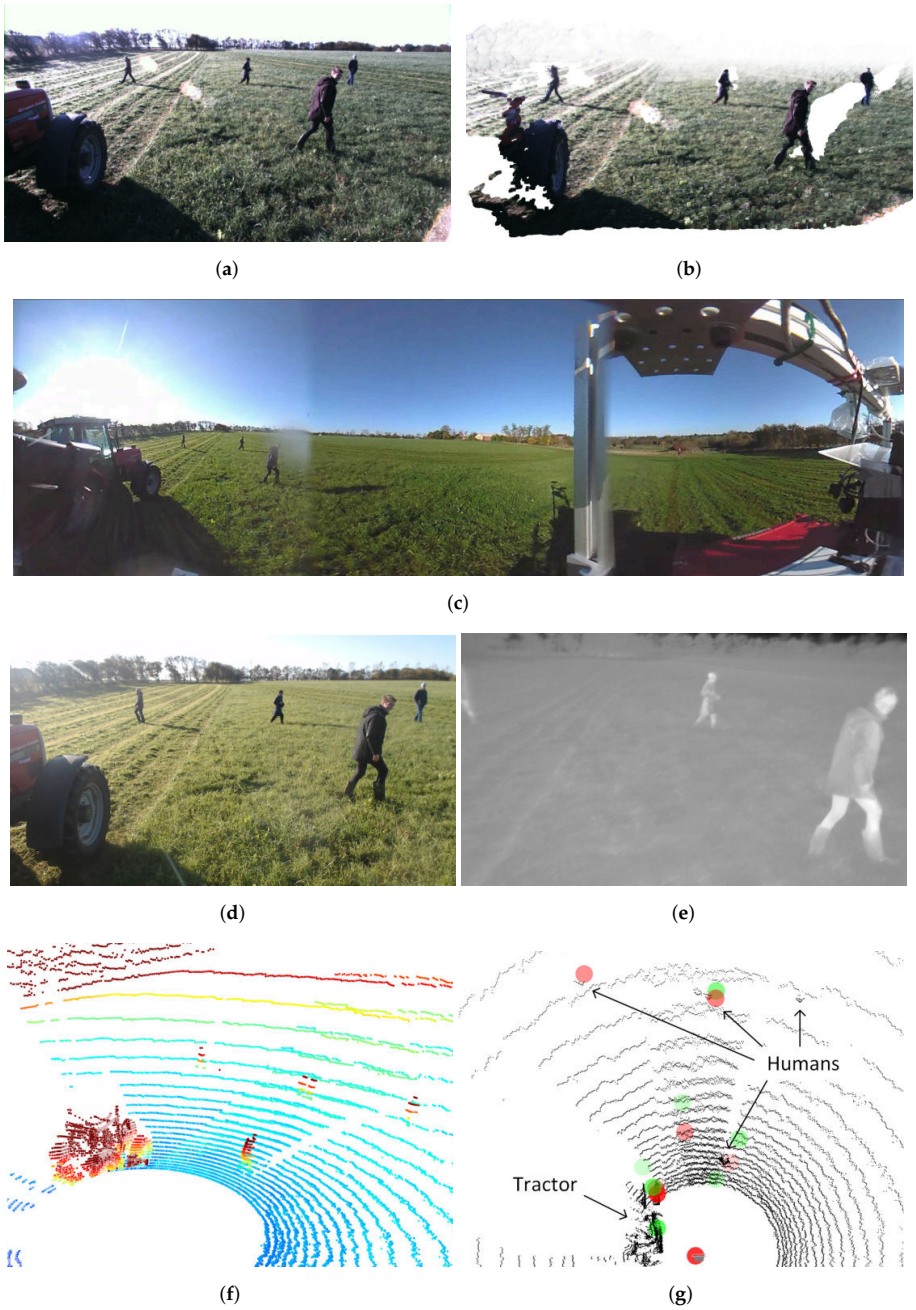


Figure 3. Example frames from the FieldSAFE dataset. (**a**) Left stereo image; (**b**) stereo pointcloud; (**c**) 360° camera image (cropped); (**d**) web camera image; (**e**) thermal camera image (cropped); (**f**) LiDAR point cloud (cropped and colored by height); (**g**) radar detections overlaid on LiDAR point cloud (black). Green and red circles denote detections from mid- and long-range modes, respectively.

Synchronization: Trigger signals for the stereo and thermal cameras were synchronized and generated from a pulse-per-second signal from an internal GNSS in the LiDAR, which allowed exact timestamps for all three sensors. The remaining sensors were synchronized in software using a best-effort approach in ROS, where the ROS system time was used to timestamp each message once it got delivered. However, best-effort message delivery does not provide any guarantees for delivery times, and the specific time delays for the different sensors therefore depend on the internal processing in the sensor, the transmission to the computer, network traffic load, the kernel scheduler and software drivers in ROS [33]. Time delays can therefore vary significantly and are not necessarily constant.

IMU and GNSS both use serial communication and therefore have very small transmission latencies. The same applies for radar that sends its data on the CAN bus. The web camera, however, uses a USB 2.0 interface and thus experiences a short delay in the transmission. A typical delay for the web camera has been measured as 100 ms. The 360° camera uses the TCP protocol and experiences a large amount of packet retransmissions. The delay has therefore been measured up to 4.5 s. The time delays are both specified in relation to the stereo camera, which is synchronized to the LiDAR and thermal camera.

Registration: All sensors were registered by estimating extrinsic parameters (translation and rotation). A common reference frame, base link, was defined at the mount point of the recording frame on the tractor. From here, extrinsic parameters were estimated either by hand measurements or using automated calibration procedures. Figure 4 illustrates the chain of registrations and how they were carried out. The LiDAR and the stereo camera were registered by optimizing the alignment of 3D point clouds from both sensors. For this procedure, the iterative closest point (ICP) was used on multiple static scenes. An average over all scenes was used as the final estimate. The stereo and thermal cameras were registered and calibrated using the camera calibration method available in the Computer Vision System Toolbox in MATLAB. Since the thermal camera did not perceive light in the visual spectrum, a custom-made visual-thermal checkerboard was used. For a more detailed description of this procedure, we refer the reader to [34]. The remaining sensors were registered by hand, by estimating extrinsic parameters of their positions. All extrinsic parameters are contained in the dataset. Instructions for how to extract these are available at the FieldSAFE website. Here, the estimated intrinsic camera parameters are further available for download.

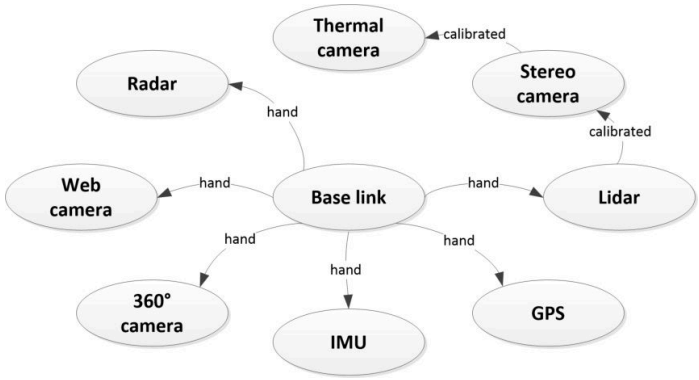
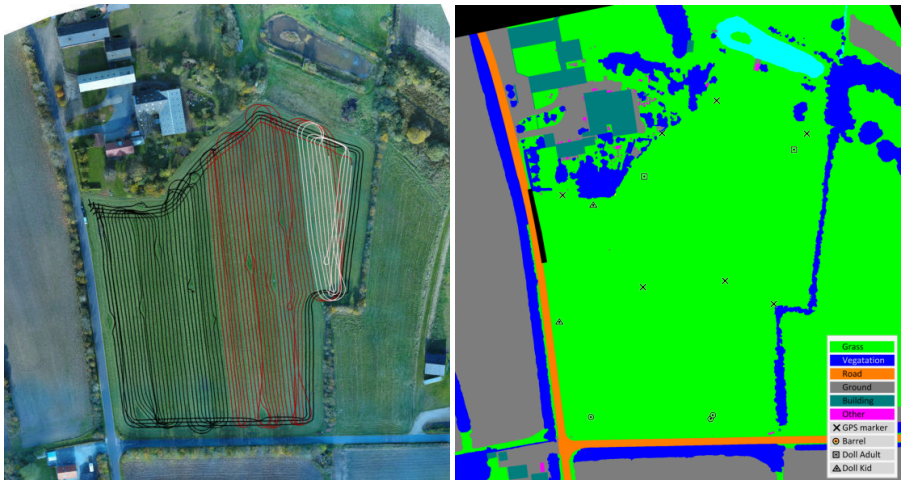


Figure 4. Sensor registration. "Hand" denotes a manual measurement by hand, whereas "calibrated" indicates that an automated calibration procedure was used to estimate the extrinsic parameters.

3. Dataset

The dataset consists of approximately 2 h of recordings during grass mowing in Denmark, 25 October 2016. The exact position of the field was 56.066742, 8.386255 (latitude, longitude). Figure 5a

shows a map of the field with tractor paths overlaid. The field is 3.3 ha and surrounded by roads, shelterbelts and a private property.



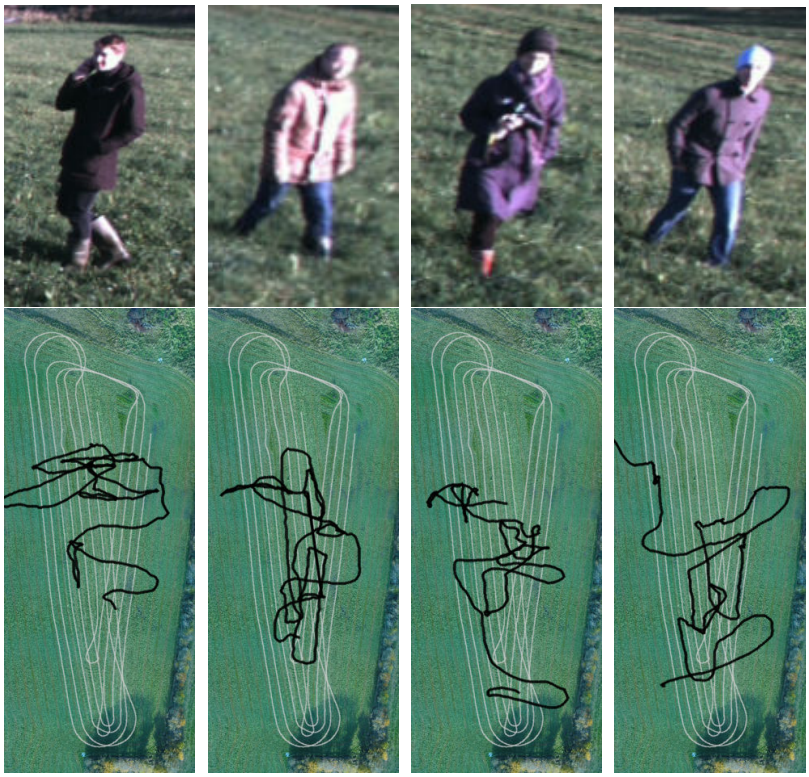
(**a**) (**b**)

Figure 5. Colored and labeled orthophotos. (**a**) Orthophoto with tractor tracks overlaid. Black tracks include only static obstacles, whereas red and white tracks also have moving obstacles. Currently, red tracks have no ground truth for moving obstacles annotated. (**b**) Labeled orthophoto.

A number of static obstacles exemplified in Figure 6 were placed on the field prior to recording. They included mannequin dolls (adults and children), rocks, barrels, buildings, vehicles and vegetation. Figure 5b shows the placement of static obstacles on the field overlaid on a ground truth map colored by object classes.

Figure 6. Examples of static obstacles.

Additionally, a session with moving obstacles was recorded where four humans were told to walk in random patterns. Figure 7 shows the four subjects and their respective paths on a subset of the field. The subset corresponds to the white tractor tracks in Figure 5a. The humans crossed the path of the tractor a number of times, thus emulating dangerous situations that must be detected by a safety system. Along the way, various poses such as standing, sitting and lying were represented.



(**a**) Human 1 (**b**) Human 2 (**c**) Human 3 (**d**) Human 4

Figure 7. Examples of moving obstacles (from the stereo camera) and their paths (black) overlaid on the tractor path (grey).

During the entire traversal and mowing of the field, data from all sensors were recorded. Along with video from a hovering drone, a static orthophoto from another drone and corresponding manually-annotated class labels, these are all available from the FieldSAFE website.

4. Ground Truth

Ground truth information on object location and class labels for both static and moving obstacles is available as timestamped global (geographic) coordinates. By transforming local sensor data from the tractor into global coordinates, a simple look-up of the class label in the annotated ground truth map is possible.

Prior to traversing and mowing the field, a number of custom-made markers were distributed on the ground and measured with exact global coordinates using a handheld Topcon GRS-1 RTK GNSS. A DJI Phantom 4 drone was used to take overlapping bird's-eye view images of an area covering the field and its surroundings. Pix4D [35] was used to stitch the images and generate a high-resolution orthophoto (Figure 5a) with a ground sampling distance (GSD) of 2 cm. The orthophoto was manually labeled pixel-wise as either grass, ground, road, vegetation, building, GPS marker, barrel, human or other (Figure 5b). Using the GPS coordinates of the markers and their corresponding positions in the orthophoto, a mapping between GPS coordinates and pixel coordinates was estimated.

For annotating the location of moving obstacles, a DJI Matrice 100 was used to hover approximately 75 m above the ground while the tractor traversed the field. The drone recorded video at 25 fps with a

resolution of 1920 × 1080. Due to limited battery capacity, the recording was split into two sessions of each 20 min. The videos were manually synchronized with sensor data from the tractor by introducing physical synchronization events in front of the tractor in the beginning and end of each session. Using the seven GPS markers that were visible within the field of view of the drone, the videos were stabilized and warped to a bird's-eye view of a subset of the field. As described above for the static orthophoto, GPS coordinates of the markers and their corresponding positions in the videos were then used to generate a mapping between GPS coordinates and pixel coordinates. Finally, the moving obstacles were manually annotated in each frame of one of the videos using the vatic video annotation tool [36]. Figure 7 shows the path of each object overlaid on a subset of the orthophoto. The second video is yet to be annotated.

5. Summary and Future Work

In this paper, we have presented a calibrated and synchronized multi-modal dataset for obstacle detection in agriculture. The dataset supports research into object detection and classification, object tracking, sensor fusion, localization and mapping. We envision the dataset to facilitate a wide range of future research within autonomous agriculture and obstacle detection for farming vehicles.

In future work, we plan on annotating the remaining session with moving obstacles. Additionally, we would like to extend the dataset with more scenarios from various agricultural environments while widening the range of encountered illumination and weather conditions.

Currently, all annotations reside in a global coordinate system. Projecting these annotations to local sensor frames inevitably causes localization errors. Therefore, we would like to extend annotations with, e.g., object bounding boxes for each sensor.

Acknowledgments: This research is sponsored by the Innovation Fund Denmark as part of the project "SAFE—Safer Autonomous Farming Equipment" (project No. 16-2014-0). The authors thank Anders Krogh Mortensen for his valuable help in processing all drone recordings and generating stitched, georeferenced orthophotos. Further, we thank the participating companies in the project, AgroIntelli, Conpleks Innovation ApS, CLAAS E-Systems, KeyResearch and RoboCluster, for their help in organizing the field experiment, providing sensor and processing equipment and promoting the project in general.

Author Contributions: M.F.K.and P.C. designed the sensor platform including interfacing, calibration, registration and synchronization. M.F.K. and P.C. conceived of and designed the experiments and provided manual ground truth annotations. M.S.L. and M.L. contributed with sensor interfacing, calibration and synchronization. K.A.S., O.G. and R.N.J. contributed with agricultural domain knowledge, provided test facilities and performed the experiments. H.K. contributed with insight into the experimental design and computer vision. M.F.K. wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abidine, A.Z.; Heidman, B.C.; Upadhyaya, S.K.; Hills, D.J. Autoguidance system operated at high speed causes almost no tomato damage. *Calif. Agric.* **2004**, *58*, 44–47.
2. Case IH. Case IH Autonomous Concept Vehicle, 2016. Available online: http://www.caseih.com/apac/en-in/news/pages/2016-case-ih-premieres-concept-vehicle-at-farm-progress-show.aspx (accessed on 9 August 2017).
3. ASI. Autonomous Solutions, 2016. Available online: https://www.asirobots.com/farming/ (accessed on 9 August 2017).
4. Kubota, 2017. Available online: http://www.kubota-global.net/news/2017/20170125.html (accessed on 16 August 2017).
5. Ollis, M.; Stentz, A. Vision-based perception for an automated harvester. In Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems, Innovative Robotics for Real-World Applications (IROS '97), Grenoble, France, 11 September 1997; Volume 3, pp. 1838–1844.
6. Stentz, A.; Dima, C.; Wellington, C.; Herman, H.; Stager, D. A system for semi-autonomous tractor operations. *Auton. Robots* **2002**, *13*, 87–104.

7. Wellington, C.; Courville, A.; Stentz, A.T. Interacting markov random fields for simultaneous terrain modeling and obstacle detection. In Proceedings of the Robotics: Science and Systems, Cambridge, MA, USA, 8–11 June 2005; Volume 17, pp. 251–260.
8. Griepentrog, H.W.; Andersen, N.A.; Andersen, J.C.; Blanke, M.; Heinemann, O.; Madsen, T.E.; Nielsen, J.; Pedersen, S.M.; Ravn, O.; Wulfsohn, D. Safe and reliable: Further development of a field robot. *Precis. Agric.* **2009**, *9*, 857–866.
9. Moorehead, S.S.J.; Wellington, C.K.C.; Gilmore, B.J.; Vallespi, C. Automating orchards: A system of autonomous tractors for orchard maintenance. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Workshop on Agricultural Robotics, Vilamoura, Portugal, 7–12 October 2012; p. 632.
10. Reina, G.; Milella, A. Towards autonomous agriculture: Automatic ground detection using trinocular stereovision. *Sensors* **2012**, *12*, 12405–12423.
11. Emmi, L.; Gonzalez-De-Soto, M.; Pajares, G.; Gonzalez-De-Santos, P. New trends in robotics for agriculture: Integration and assessment of a real fleet of robots. *Sci. World J.* **2014**, *2014*, doi:10.1155/2014/404059.
12. Ross, P.; English, A.; Ball, D.; Upcroft, B.; Corke, P. Online novelty-based visual obstacle detection for field robotics. In Proceedings of the IEEE International Conference on Robotics and Automation, Seattle, WA, USA, 26–30 May 2015; pp. 3935–3940.
13. Ball, D.; Upcroft, B.; Wyeth, G.; Corke, P.; English, A.; Ross, P.; Patten, T.; Fitch, R.; Sukkarieh, S.; Bate, A. Vision-based obstacle detection and navigation for an agricultural robot. *J. Field Robot.* **2016**, *33*, 1107–1130.
14. Reina, G.; Milella, A.; Rouveure, R.; Nielsen, M.; Worst, R.; Blas, M.R. Ambient awareness for agricultural robotic vehicles. *Biosyst. Eng.* **2016**, *146*, 114–132.
15. Didi. Didi Data Release #2—Round 1 Test Sequence and Training. Available online: http://academictorrents.com/details/18d7f6be647eb6d581f5ff61819a11b9c21769c7 (accessed on 8 November 2017).
16. Udacity. Udacity Didi Challenge—Round 2 Dataset. Available online: http://academictorrents.com/details/67528e562da46e93cbabb8a255c9a8989be3448e (accessed on 8 November 2017).
17. Udacity, Didi. Udacity Didi $100k Challenge Dataset 1. Available online: http://academictorrents.com/details/76352487923a31d47a6029ddebf40d9265e770b5 (accessed on 8 November 2017).
18. DIPLECS. DIPLECS Autonomous Driving Datasets, 2015. Available online: http://ercoftac.mech.surrey.ac.uk/data/diplecs/ (accessed on 31 August 2017).
19. Koschorrek, P.; Piccini, T.; Öberg, P.; Felsberg, M.; Nielsen, L.; Mester, R. A multi-sensor traffic scene dataset with omnidirectional video. Ground Truth—What is a good dataset? In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Portland, OR, USA, 23–28 June 2013.
20. Maddern, W.; Pascoe, G.; Linegar, C.; Newman, P. 1 Year, 1000 km: The Oxford RobotCar Dataset. *Int. J. Robot. Res.* **2017**, *36*, 3–15.
21. InSight. InSight SHRP2, 2017. Available online: https://insight.shrp2nds.us/ (accessed on 31 August 2017).
22. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets Robotics: The KITTI Dataset. *Int. J. Robot. Res. (IJRR)* **2013**, *32*, 1231–1237.
23. Matzen, K.; Snavely, N. NYC3DCars: A dataset of 3D vehicles in geographic context. In Proceedings of the International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013.
24. Caraffi, C.; Vojir, T.; Trefny, J.; Sochman, J.; Matas, J. A system for real-time detection and tracking of vehicles from a single Car-Mounted camera. In Proceedings of the 2012 15th International IEEE Conference on Intelligent Transportation Systems (ITSC), Anchorage, AK, USA, 16–19 September 2012; pp. 975–982.
25. Ros, G.; Sellart, L.; Materzynska, J.; Vazquez, D.; Lopez, A. The SYNTHIA Dataset: A Large collection of synthetic images for semantic segmentation of urban scenes. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
26. Gaidon, A.; Wang, Q.; Cabon, Y.; Vig, E. Virtual worlds as proxy for multi-object tracking analysis. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
27. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.

28. Neuhold, G.; Ollmann, T.; Rota Bulò, S.; Kontschieder, P. The mapillary vistas dataset for semantic understanding of street scenes. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
29. Peynot, T.; Scheding, S.; Terho, S. The Marulan Data Sets: Multi-sensor perception in natural environment with challenging conditions. *Int. J. Robot. Res.* **2010**, *29*, 1602–1607.
30. Pezzementi, Z.; Tabor, T.; Hu, P.; Chang, J.K.; Ramanan, D.; Wellington, C.; Babu, B.P.W.; Herman, H. Comparing apples and oranges: Off-road pedestrian detection on the NREC agricultural person-detection dataset. *arXiv* **2017**, arXiv:1707.07169.
31. Quigley, M.; Conley, K.; Gerkey, B.P.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An Open-Source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 17 May 2009.
32. Moore, T.; Stouch, D. A Generalized extended kalman filter implementation for the robot operating system. In *Advances in Intelligent Systems and Computing*; Springer: Cham, Switzerland, 2016.
33. Lütkebohle, I. Determinism in Robotics Software. Conference Presentation, ROSCon, 2017. Available online: https://roscon.ros.org/2017/presentations/ROSCon%202017%20Determinism%20in%20ROS.pdf (accessed on 31 October 2017).
34. Christiansen, P.; Kragh, M.; Steen, K.A.; Karstoft, H.; Jørgensen, R.N. Platform for evaluating sensors and human detection in autonomous mowing operations. *Precis. Agric.* **2017**, *18*, 350–365.
35. Pix4D. 2014. Available online: http://pix4d.com/ (accessed on 5 September 2017).
36. Vondrick, C.; Patterson, D.; Ramanan, D. Efficiently scaling up crowdsourced video annotation. *Int. J. Comput. Vis.* **2013**, *101*, 184–204.

Paper 4

Object Detection and Terrain Classification in Agricultural Fields using 3D Lidar Data
Mikkel Fly Kragh, Rasmus Nyholm Jørgensen, and Henrik Pedersen

Peer reviewed
Presented at the 10th International Conference on Computer Vision Systems (ICVS), July 2015, Copenhagen, Denmark

Object Detection and Terrain Classification in Agricultural Fields Using 3D Lidar Data

Mikkel Kragh( ), Rasmus N. Jørgensen, and Henrik Pedersen

Department of Engineering, Aarhus University, Finlandsgade 22, Aarhus, Denmark
{mkha,rnj,hpe}@eng.au.dk

Abstract. Autonomous navigation and operation of agricultural vehicles is a challenging task due to the rather unstructured environment. An uneven terrain consisting of ground and vegetation combined with the risk of non-traversable obstacles necessitates a strong focus on safety and reliability. This paper presents an object detection and terrain classification approach for classifying individual points from 3D point clouds acquired using single multi-beam lidar scans. Using a support vector machine (SVM) classifier, individual 3D points are categorized as either ground, vegetation, or object based on features extracted from local neighborhoods. Experiments performed at a local working farm show that the proposed method has a combined classification accuracy of 91.6 %, detecting points belonging to objects such as humans, animals, cars, and buildings with 81.1 % accuracy, while classifying vegetation with an accuracy of 97.5 %.

Keywords: Object detection · Terrain classification · Agriculture · Lidar

1 Introduction

Autonomous farming is the concept of automatic agricultural machines operating safely and efficiently without human intervention. In order to ensure safe autonomous operation, robust real-time risk detection is crucial. Humans, animals, trees, other machines, etc. must be detected in due time to perform risk avoidance.

A lidar sensor measures range data to a set of surrounding points and generates a point cloud where each point is represented by a 3D position. It provides very accurate depth information in 360 ° horizontally and is robust towards changing lighting conditions. The lidar sensor has been used extensively in the automotive industry for detecting and localizing objects in urban environments by distinguishing between ground and obstacles [11]. In agriculture, however, a subdivision between objects and vegetation is necessary, since some apparent obstacles actually represent traversable crops. Therefore, a classification of points into ground, vegetation, and objects is needed. The ground class identifies accessible terrain, whereas the object class identifies obstacles/risks. The vegetation class serves as an intermediate category identifying both crops, bushes,

L. Nalpantidis et al. (Eds.): ICVS 2015, LNCS 9163, pp. 188–197, 2015.
DOI: 10.1007/978-3-319-20904-3_18

and trees. Depending on the agricultural context, vegetation can thus be either obstacles or a natural part of the field area.

In the literature, different approaches have been used to detect objects and characterize terrain in agricultural environments. [1,12–14] use single-beam lidar sensors and a mathematical density function for homogeneous grass to discriminate obstacles from grass and foliage. [6,15] use multi-beam lidars to perform ground plane identification in rough terrain. However, vegetation is not discriminated from objects. [8,9,18] use a feature-based approach for classifying individual points into the classes: scatter, linear, and surface. The objective is to identify vegetation (scatter); wires and tree branches (linear); and ground surfaces, rocks, and tree trunks (surface). [19] adds to this the objective of differentiating between vegetation and objects for increasing safety. This is done with a feature-based approach using online adaptation allowing the system to automatically collect and interpret training data. However, the results of this approach are only visually verified, and only a few specific cases are handled.

In this paper, we present an object detection approach for classifying individual points from 3D point clouds acquired with a vehicle-mounted Velodyne HDL-32E lidar. Our method calculates for each point 13 different features based on a local neighborhood. In order to account for the varying point density experienced with a vehicle-mounted lidar, we propose an adaptive neighborhood radius depending on the distance ensuring high resolution at short distance and preventing noisy features at far distance. Using a support vector machine (SVM), each point is categorized into one of three classes: ground, vegetation, or object.

The paper is divided into 5 sections. Section 2 presents the proposed approach including preprocessing, feature extraction, and classification. Section 3 presents the experimental setup and results followed by a discussion in Sect. 4. Ultimately, Sect. 5 presents a conclusion and future work.

2 Approach

The proposed method for object detection and terrain classification builds on individual point classification of single multi-beam lidar scans. A single lidar scan provides a 3D point cloud consisting of N points. For each point, 13 features are calculated using statistics from a local neighborhood. These features describe the distribution of points into surfaces, linear structures, clutter volumes, etc. and serve to distinguish between points representing the three classes: ground, vegetation, and object. Using hand labeled data, an SVM classifier is trained to classify individual points based on their calculated features.

2.1 Preprocessing

An initial step before extracting features performs a rotation and translation of the point cloud according to a globally estimated plane. This ensures that ground points in general lie close to the xy-plane. Due to variations in point density, the point cloud is first resampled using a minimum filter with a fixed sized radius

of 15 cm. A global plane is then estimated using a RANSAC-based plane fitting algorithm [5]. The point cloud is finally translated and rotated according to the normal vector of this plane. The resulting point cloud has an approximately vertically oriented z-axis.

2.2 Feature Extraction

When analyzing 3D data points from a point cloud, the notion of scale is extremely important in order to obtain both robust and accurate information. Point features are calculated using a local neighborhood such that the points located close to an evaluated point contribute with information of the point's context. For instance, one feature might describe how well a point fits with a local planar surface estimated on its neighborhood. The radius of the neighborhood should depend on the desired accuracy but also on the noise levels and the density of the point cloud. Depending on the sensor used for acquiring 3D data, a point cloud can be categorized as either dense or sparse [4]. A dense point cloud has an approximately constant point density, whereas the density of a sparse point cloud (e.g. from a single lidar scan) varies with the distance. Therefore, the process of feature extraction should incorporate information of the local point density and possibly also adjust the radius of the neighborhood accordingly.

Traditionally, the neighborhood radius is kept constant by dividing all points into a global voxel representation [7,8,19]. This approach allows for easy feature calculation and comparison since all voxels are the same size. However, it has the unfortunate property that it does not exploit the high point resolution close to the sensor, and at far distances only few measurements are available resulting in too noisy features. Different approaches have been made to handle this issue of varying point density. An automatic scale selection method estimates the optimal neighborhood radius that minimizes the error of local normal estimation [9]. Another approach is to perform feature extraction on multiple scales and choose the local scale that has the highest saliency [10,17]. However, these approaches both rely on a specific measure that cannot be generalized across all possible features and structures. Also, computing features at multiple scales significantly increases the computational complexity.

Therefore, in this paper we propose a simple heuristic approach that scales the neighborhood radius r linearly with the sensor distance d. This has the benefit of computational simplicity while allowing fine estimation close to the sensor and a more coarse estimate far from the sensor. The specific relationship is given as

$$r = 0.0276d + 0.25 \tag{1}$$

such that a radius of 0.3 m is used at a distance of 2 m, whereas a radius of 3.0 m is used at a distance of 100m.

It is important that all features are made scale-invariant such that the neighborhood radius does not directly influence the features. A common normalization technique is not applicable since the features express different characteristics. Hence, we need to consider normalization for each feature separately.

A total of 13 features related to the height, shape, orientation, distance, and reflectance are calculated. In the following, these are explained in detail, and individual normalization techniques are discussed.

f_1, f_2, f_3, f_4: Height. Four height related features are calculated inspired by the work in [15]. Height features capture structures that protrude from the ground either positively (upwards) or negatively (downwards). f_1 is simply the z-coordinate of the evaluated point i. f_2 is the minimum z-coordinate of the neighborhood. f_3 is the average z-coordinate of all points in the neighborhood. f_4 is the standard deviation of all z-coordinates. Since the standard deviation depends directly on the size of the neighborhood, it is normalized by dividing by the neighborhood radius r. In the following equations, z_i denotes the z-coordinate of the i'th point, and k denotes the number of points within a neighborhood of radius r. k thus varies with r and the specific point density locally around point i.

$$f_1 = z_i \tag{2}$$

$$f_2 = min\left(z_1 \dots z_k\right) \tag{3}$$

$$f_3 = \overline{z} = \frac{1}{k}\sum_{j=1}^{k} z_j \tag{4}$$

$$f_4 = \frac{\sigma_z}{r} = \frac{1}{r}\sqrt{\frac{1}{k}\sum_{j=1}^{k}\left(z_j - \overline{z}\right)^2} \tag{5}$$

f_5, f_6, f_7, f_8: Shape. Principal component analysis (PCA) of the point neighborhood can be used to describe the shape/saliency of the point cloud [8,18,19]. Let $\lambda_1 < \lambda_2 < \lambda_3$ be the eigenvalues of the 3×3 covariance matrix. In case of scattered points (random point distribution), $\lambda_1 \approx \lambda_2 \approx \lambda_3$. For points on planes, $\lambda_2, \lambda_3 \gg \lambda_1$, whereas for linear structures $\lambda_3 \gg \lambda_1, \lambda_2$. Using this intuition, λ_1 captures vegetation, $\lambda_2 - \lambda_1$ captures linear structures, whereas $\lambda_3 - \lambda_2$ captures planar-like data.

Constructing scale-invariant PCA features can be done in different ways. [10] scales λ_2 and λ_3 by the neighborhood radius but leaves λ_1 intact. This results in scale-invariant eigenvalues for planar-like data, whereas scatteredness is left unscaled. [16], on the other hand, uses the ratio of PCA values.

In this paper, we utilize the eigenvalue differences as described above and scale them by the largest eigenvalue. This guarantees scale-invariant features (always adds up to 1) while allowing for the differentiation between scatter, linear, and planar structures.

$$f_5 = \frac{\lambda_1}{\lambda_3} \tag{6}$$

$$f_6 = \frac{\lambda_2 - \lambda_1}{\lambda_3} \tag{7}$$

$$f_7 = \frac{\lambda_3 - \lambda_2}{\lambda_3} \tag{8}$$

In addition to the three PCA shape features, we use a normalized orthogonal residual sum of squares (RSS) proposed by [15].

$$f_8 = \frac{1}{k} \sum_{j=1}^{k} \left((\boldsymbol{p}_j - \overline{\boldsymbol{p}}) \cdot \boldsymbol{v}_1 \right)^2 \tag{9}$$

where $\boldsymbol{v}_1$ is the eigenvector corresponding to the smallest eigenvalue λ_1, $\boldsymbol{p}_i$ is the 3D vector of the i'th point, and $\overline{\boldsymbol{p}}$ is the neighborhood mean (centroid).

f_9, f_10, f_11: Orientation. From the principal component analysis, the eigenvector $\boldsymbol{v}_1$ is equal to the normal vector of a locally estimated plane. $\boldsymbol{v}_1$ thus describes the orientation of the plane. The z-component of the vector has been used to capture ground points assuming that the terrain is fairly flat and not sloped [10,15]. In this paper we include all the components.

$$f_9 = \boldsymbol{v}_1 \cdot (1, 0, 0) \tag{10}$$

$$f_{10} = \boldsymbol{v}_1 \cdot (0, 1, 0) \tag{11}$$

$$f_{11} = \boldsymbol{v}_1 \cdot (0, 0, 1) \tag{12}$$

f_12: Distance. Although the distance-dependent point density to some degree is handled by the varying neighborhood radius, the distance from a point $\boldsymbol{p}_j$ to the sensor $\boldsymbol{s}$ can also be used as a predictor [19].

$$f_{12} = \sqrt{(\boldsymbol{p}_i - \boldsymbol{s}) \cdot (\boldsymbol{p}_i - \boldsymbol{s})} \tag{13}$$

f_13: Reflectance. The lidar sensor utilized in the experiments provides for each point a reflectance intensity. This can help differentiate between different materials, although it depends also on the distance and incident angle [10,19].

$$f_{13} = \text{intensity}_i \tag{14}$$

2.3 Classification

A support vector machine (SVM) classifier is trained on hand-labeled data and used to differentiate between ground, vegetation, and object. In order to balance the training data, a number of ground and vegetation points, corresponding to

the number of object points, are drawn by random. We use the LIBSVM implementation [2] with a radial basis function (RBF) kernel and default SVM parameters $C = 1$ and $\gamma = \frac{1}{\#features} = \frac{1}{13}$. Prior to feeding the classifier, features are normalized by subtracting the mean and dividing by the standard deviation for each dimension across the training data. The normalization parameters are then stored for subsequent use in the test procedure.

3 Experiments and Results

An experimental dataset was acquired on a local working farm in Denmark in November 2014. Figure 1 shows the custom-built vehicle-mounted sensor platform including a Velodyne HDL-32E lidar [3]. In addition to the lidar sensor, a number of visual and pose sensors were mounted for subsequent analysis. The recordings include high and low grass, a large number of trees, 2 buildings, 2 cars, 5 men, 7 children, and 2 dogs, all from different angles and distances. 15 lidar frames from 7 different trials (recordings) were subsequently hand labeled into the three classes: ground, vegetation, and object. Results have been obtained using leave-one-out cross-validation (with 7 folds corresponding to the different trials), thereby training on 6 and testing on a single fold at a time. Separating trials in the cross-correlation should prevent overfitting, which would otherwise occur due to high correlation between frames within the same trial.

Table 1 presents a confusion matrix showing the accumulated counts of points across the 7 folds classified correctly or incorrectly compared to the ground truth. As mentioned above, the uneven distribution of ground, vegetation, and object points is evened out by drawing by random a number of these, corresponding to the number of object points, from individual frames. The results show a combined classification accuracy of 91.6 %. Points belonging to the ground are correctly predicted as ground with 96.4 % accuracy, and points belonging to vegetation are correctly predicted as vegetation with 97.5 % accuracy. Object points, however, are more often mistaken for vegetation, resulting in an object detection accuracy of 81.1 %.

Fig. 1. Sensor platform mounted on tractor.

Table 1. Confusion matrix relating predictions (columns) to ground truth (rows).

	Ground	Vegetation	Object
Ground	44806 (96.4 %)	1234 (2.7 %)	437 (0.9 %)
Vegetation	724 (1.6 %)	43372 (97.5 %)	381 (0.9 %)
Object	728 (1.6 %)	8041 (17.3 %)	37708 (81.1 %)

Figure 2 illustrates examples of two frames with ground truth labels and classifier predictions. The problem of object/vegetation confusion is particularly visible in Fig. 2b on the side of the building. Here, around half of the building is incorrectly predicted as vegetation.

Two feature selection techniques were used to investigate the individual importance of the 13 features. Both techniques use only a subset of all combinations of features, since exhaustive search is impractical with $\sum_{f=1}^{13} \binom{13}{f} = 8191$ combinations. In order to evaluate a feature combination, a common metric is needed. Since the features are ultimately used for classification, a wrapper method detecting possible interactions between features was used. The SVM classifier was thus trained on each feature combination, and the accuracy was used as a score.

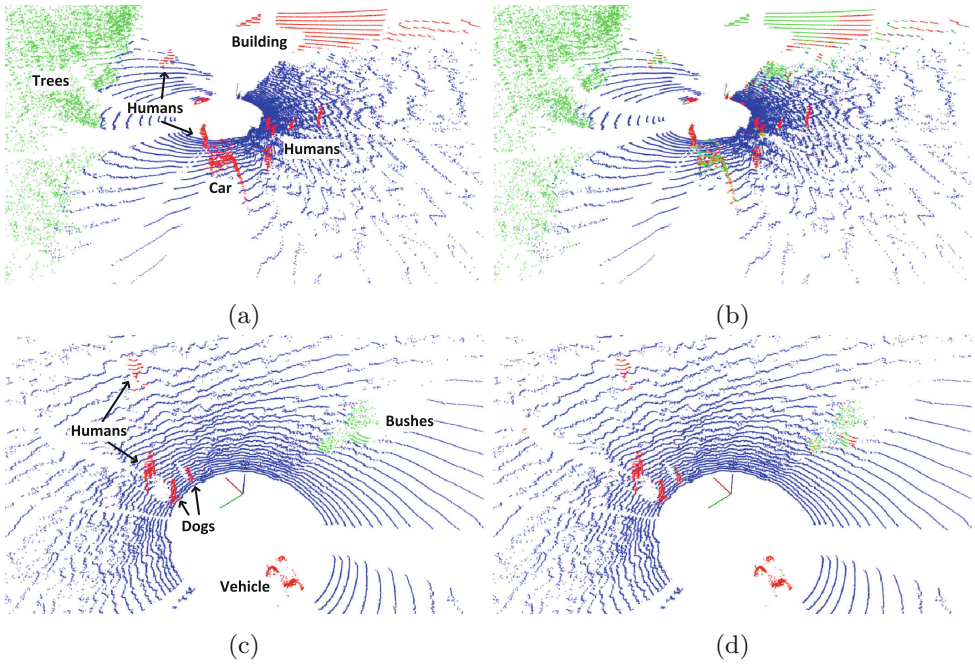
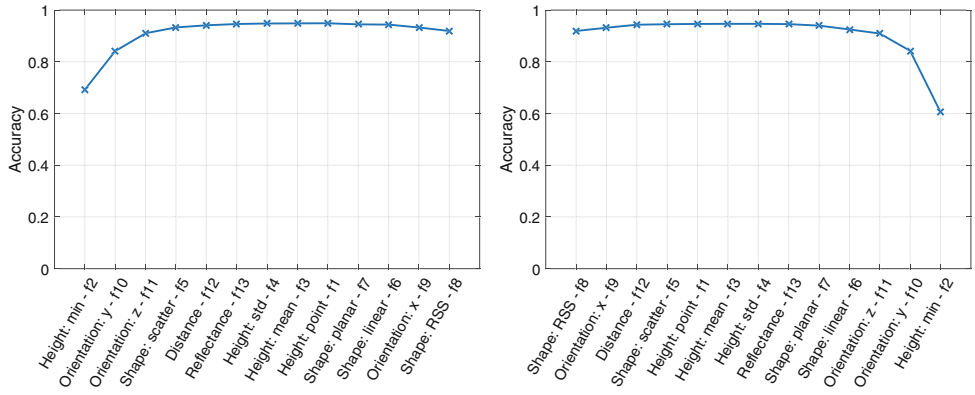


Fig. 2. Examples of classification results. a) and b) respectively show ground truth and classification results of a scene with ground, trees, humans, a car, and a building. c) and d) respectively show ground truth and classification results of a scene with ground, bushes, humans, and dogs. Blue denotes ground, green denotes vegetation, and red denotes objects (Colour figure online).



(a) Forward selection. (b) Backward elimination.

Fig. 3. Feature selection using greedy forward selection and backward elimination.

Greedy forward selection starts by evaluating all features individually and assigns for each a classification score. The feature with the highest score is added to a set of used features, and this set is gradually increased by iteratively adding the highest scoring feature of the remaining unused features. Figure 3a shows the relevance sorting of this approach. The most relevant feature is considered to be f_2 (minimum height), whereas the least relevant is f_8 (RSS).

Greedy backward elimination, on the other hand, starts by evaluating all features in combination leaving out a single feature. The feature that gives the smallest decrease in score is then eliminated, and the process is continued iteratively until a single feature is left. Figure 3b shows the relevance sorting of this approach. As for the forward selection, the most relevant feature is considered to be f_2 (minimum height), and the least relevant is f_8 (RSS).

All computations were performed using C++ on a laptop with an Intel i7 Quad-core CPU at 2.7GHz and 16GB of RAM. The average execution time is 705ms per frame. Preprocessing takes 2.4ms, feature extraction takes 324.9ms, and classification takes 377.9ms.

4 Discussion

Due to the interaction of features, the two feature selection techniques do not fully agree about the sorting of all relevances. However, some observations can be made from the graphs. Using more than 5 features seems to be unnecessary, as it does not significantly increase the accuracy. This is an important observation, since utilizing fewer features results in decreased computational complexity. Another common trend of the two graphs is seen by looking at the three feature categories: height, shape, and orientation. Only one or two features within each category are considered relevant. This implies (but does not prove) that features within each of the categories are correlated and thus redundant. Although the two techniques do not agree about the specific features, they both include a

height, shape, and orientation feature among the most relevant four features. A reasonable choice of feature reduction would therefore be to select the intersection of the 5 most significant features from the two selection techniques.

5 Conclusion

In this paper, we have presented an object detection approach for classifying individual points from 3D point clouds acquired with a vehicle-mounted lidar. Our method calculates for each point 13 different features based on a local neighborhood. In order to account for the varying point density experienced with a vehicle-mounted lidar, the neighborhood radius depends on the distance ensuring high resolution at short distance and preventing noisy features at far distance. Using a support vector machine, each point is categorized into one of three classes: ground, vegetation, or object.

The proposed method shows promising results on an experimental dataset recorded on a working farm including grass, trees, buildings, cars, humans, and animals. It has a combined classification accuracy of 91.6 %. Ground points are correctly classified with an accuracy of 96.4 %, and points belonging to vegetation are correctly predicted as vegetation with 97.5 % accuracy. Object points, however, are more often mistaken for vegetation, resulting in an object detection accuracy of 81.1 %.

In order to increase differentiation performance, further work will focus on temporal accumulation of lidar frames using odometry information from GPS and IMU sensors. Also, further differentiation and characterization of objects will require additional information possibly by fusing lidar and vision sensors.

Acknowledgements. This research is sponsored by the Innovation Fund Denmark as part of the project "SAFE - Safer Autonomous Farming Equipment" (project no. 16-2014-0).

References

1. Castano, A., Matthies, L.: Foliage discrimination using a rotating ladar. In: 2003 IEEE International Conference on Robotics and Automation, vol. 1, pp. 1–6 (2003)
2. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 1–27 (2011). Article No. 27
3. Christiansen, P., Kragh, M., Steen, K.A., Karstoft, H., Jørgensen, R.N.: Advanced sensor platform for human detection and protection in autonomous farming. In: 10th European Conference on Precision Agriculture (ECPA 2015) (2015)
4. Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P., Frenkel, A.: On the segmentation of 3D lidar point clouds. In: Proceedings - IEEE International Conference on Robotics and Automation, pp. 2798–2805. IEEE (2011)
5. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)

6. Hadsell, R., Bagnell, J.A., Huber, D., Hebert, M.: Space-carving kernels for accurate rough terrain estimation. Int. J. Robot. Res. **29**(8), 981–996 (2010)
7. Hebert, M., Vandapel, N.: Terrain classification techniques from ladar data for autonomous navigation. In: Collaborative Technology Alliances Conference (2003)
8. Lalonde, J.F., Vandapel, N., Huber, D.F., Hebert, M.: Natural terrain classification using three-dimensional ladar data for ground robot mobility. J. Field Robot. **23**(10), 839–861 (2006)
9. Lalonde, J.F., Unnikrishnan, R., Vandapel, N., Hebert, M.: Scale selection for classification of point-sampled 3D surfaces. In: Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM, pp. 285–292 (2005)
10. Lim, E.H., Suter, D.: 3D terrestrial LIDAR classifications with super-voxels and multi-scale conditional random fields. CAD Comput. Aided Des. **41**(10), 701–710 (2009)
11. Luettel, T., Himmelsbach, M., Wuensche, H.J.: Autonomous ground vehiclesconcepts and a path to the future. In: Proceedings of the IEEE, vol. 100, (Special Centennial Issue), pp. 1831–1839, May 2012
12. Macedo, J., Manduchi, R., Matthies, L.: Ladar-based discrimination of grass from obstacles for autonomous navigation. In: Proceedings of the International Symposium on Experimental Robotics VII (ISER 2001), pp. 111–120 (2001)
13. Manduchi, R., Castano, A., Talukder, A., Matthies, L.: Obstacle detection and terrain classification for autonomous off-road navigation. Auton. Robots **18**(1), 81–102 (2005)
14. Matthies, L., Bergh, C., Castano, A., Macedo, J., Manduchi, R.: Obstacle detection in foliage with ladar and radar. In: Proceedings of ISRR, pp. 291–300 (2003)
15. McDaniel, M.W., Nishihata, T., Brooks, C.A., Lagnemma, K.: Ground plane identification using LIDAR in forested environments. In: Proceedings - IEEE International Conference on Robotics and Automation, pp. 3831–3836 (2010)
16. Spinello, L., Arras, K.O., Triebel, R., Siegwart, R.: A layered approach to people detection in 3d range data. In: Proceedings of the AAAI Conference on Artificial Intelligence: Physically Grounded AI Track (AAAI) (2010)
17. Unnikrishnan, R., Hebert, M.: Multi-scale interest regions from unorganized point clouds. In: 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops (2008)
18. Vandapel, N., Huber, D., Kapuria, A., Hebert, M.: Natural terrain classification using 3-D ladar data. In: Proceedings of IEEE International Conference on Robotics and Automation, ICRA 2004, vol. 5, pp. 5117–5122 (2004)
19. Wellington, C., Stentz, A.: Online adaptive rough-terrain navigation in vegetation. In: Proceedings of IEEE International Conference on Robotics and Automation, ICRA 2004, vol. 1, pp. 96–101 (2004)

Paper 5

Multi-Modal Obstacle Detection in Unstructured Environments with Conditional Random Fields
Mikkel Fly Kragh and James Underwood

Submitted to International Journal of Robotics Research, February 2017

Awaiting final decision after submitting 1st revision December 13th 2017.

The International Journal of Robotics Research
XX(X):1–27

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Multi-Modal Obstacle Detection in Unstructured Environments with Conditional Random Fields

Mikkel Kragh[1] and James Underwood[2]

Abstract

Reliable obstacle detection and classification in rough and unstructured terrain such as agricultural fields or orchards remains a challenging problem. These environments involve large variations in both geometry and appearance, challenging perception systems that rely on only a single sensor modality. Geometrically, tall grass, fallen leaves, or terrain roughness can mistakenly be perceived as non-traversable or might even obscure actual obstacles. Likewise, traversable grass or dirt roads and obstacles such as trees and bushes might be visually ambiguous.

In this paper, we combine appearance- and geometry-based detection methods by probabilistically fusing lidar and camera sensing with semantic segmentation using a conditional random field. We apply a state-of-the-art multi-modal fusion algorithm from the scene analysis domain and adjust it for obstacle detection in agriculture with moving ground vehicles. This involves explicitly handling sparse point cloud data and exploiting both spatial, temporal, and multi-modal links between corresponding 2D and 3D regions.

The proposed method is evaluated on a diverse dataset, comprising a dairy paddock and a number of different orchards gathered with a perception research robot in Australia. Results show that for a two-class classification problem (ground and non-ground), only the camera leverages from information provided by the other modality. However, as more classes are introduced (*ground*, *sky*, *vegetation*, and *object*), both modalities complement each other and improve the mean classification score. Further improvement is achieved by introducing recursive inference with temporal links between successive frames.

Keywords

Obstacle Detection, Sensor Fusion, Field Robots, Agriculture

1 Introduction

In recent years, automation in the automotive industry has expanded rapidly with products ranging from assisted-driving features to semi-autonomous cars that are fully self-driven in certain restricted circumstances. Currently, the technology is limited to handle only very structured environments in clear conditions. However, frontiers are constantly pushed, and in the near future, fully autonomous cars will emerge that both detect and differentiate between objects and structures in their surroundings at all times.

In agriculture, automated steering systems have existed for around two decades (Abidine et al. 2004).

[1] Department of Engineering, Aarhus University, Denmark
[2] Australian Centre for Field Robotics, The University of Sydney

Corresponding author:
Mikkel Kragh, Department of Engineering, Aarhus University, Denmark
Email: mkha@eng.au.dk

Farmland is an explicitly constructed environment, which permits recurring driving patterns. Therefore, exact route plans can be generated and followed to centimeter precision using accurate global navigation systems. In order to fully eliminate the need for a human driver, however, the vehicles need to perceive the environment and automatically detect and avoid obstacles under all operating conditions. Unlike self-driving cars, farming vehicles further need to handle unknown and unstructured terrain and need to distinguish traversable vegetation such as crops and high grass from actual obstacles, although both protrude from the ground. These strict requirements are often addressed by introducing multiple sensing modalities and sensor fusion, thus increasing detection performance, solving ambiguities, and adding redundancy. Typical sensors are monocular and stereo color cameras, thermal camera, radar, and lidar. Due to the difference in their physical sensing, the detection capabilities of these modalities both complement and overlap each other (Peynot et al. 2010; Brunner et al. 2013).

A number of approaches have been made to combine multiple modalities for obstacle detection in agriculture. Self-supervised systems have been proposed for stereo-radar (Reina et al. 2016a), rgb-radar (Milella et al. 2015, 2014), and rgb-lidar (Zhou et al. 2012). Here, one modality is used to continuously supervise and improve the detection results of the other. In contrast, actual sensor fusion provides reduced uncertainty when combining multiple sensors as opposed to applying each sensor individually. A distinction is often made between low-level (early) fusion, combining raw data from different sensors, and high-level (late) fusion, integrating information at decision level. At low-level, lidar has been fused with other range-based sensors (lidar and radar) using a joint calibration procedure (Underwood et al. 2010). Additionally, lidar has been fused with cameras (monocular, stereo, and thermal) by projecting 3D lidar points onto corresponding images and concatenating either their raw outputs (Dima et al. 2004; Wellington et al. 2005) or pre-calculated features (Häselich et al. 2013). This approach potentially leverages the full potential of all sensors, but suffers from the fact that only regions covered by all modalities are defined. Furthermore, it assumes perfect extrinsic calibration between the sensors involved. At high-level, lidar and camera have been fused for ground/non-ground classification, where the idea is to simply weight the a posteriori outputs of individual classifiers by their prior classification performances (Reina et al. 2016b). Another approach combines lidar and camera in grid-based fusion for terrain classification into four classes, where again a weighting factor is used for calculating a combined probability for each cell (Laible et al. 2013). A similar approach uses occupancy grid mapping to combine lidar, radar, and camera by probabilistically fusing their equally weighted classifier outputs (Kragh et al. 2016). However, weighting classifier outputs by a common weighting factor does not leverage the potentially complex connections between sensor technologies and their detection capabilities across object classes. One sensor may recognize class A but confuse B and C, whereas another sensor may recognize C but confuse A and B. By learning this relationship, the sensors can be fused to effectively distinguish all three classes.

Recent work on object detection for autonomous driving has fused lidar and camera at a low-level to successfully learn these relationships and improve localization and detection of cars, pedestrians, and cyclists (Chen et al. 2017). The method involves a multi-view convolutional neural network performing region-based feature fusion. The idea is to apply a region proposal network in 3D to generate bounding boxes of potential objects. These 3D regions can then be projected to 2D such that features from both modalities can be fused for each region. A similar method evaluated on the same dataset has been proposed for high-level fusion of lidar and camera (Asvadi et al. 2017). The detection performance is lower than the above low-level equivalent. However, the method is considerably faster as it exploits a state-of-the-art real-time 2D network for all modalities.

Research within autonomous underwater vehicles (AUV) has fused camera images from an AUV with *a priori* remote sensing data of ocean depth (Rao et al. 2017). Here, high-level features from a deep neural network are fused across the two modalities

to provide improved classification performance, even when one of the modalities is unavailable during inference. Similarly, Eitel et al. (2015) have used a convolutional neural network to fuse color and depth images for robotic object recognition on high-level to handle imperfect or missing sensor data.

Within the domain of scene analysis, lidar and camera have recently been combined to improve classification accuracy of semantic segmentation. In these approaches, a common setup is to acquire synchronized camera and lidar data from a side-looking ground vehicle passing by a scene. A camera takes images at a fixed frequency, and a single-beam vertically-scanning laser is used in a push-broom setting, allowing subsequent accumulation of points into a combined point cloud. By looking at an area covered by both modalities, a scene consisting of a high number of 3D points and corresponding images is then post-processed, either by directly concatenating features of both modalities at low-level (Namin et al. 2014; Posner et al. 2009; Douillard et al. 2010; Cadena and Košecká 2016), or by fusing intermediate classification results provided by both modalities individually at high-level (Namin et al. 2015; Xiao et al. 2015; Zhang et al. 2015; Munoz et al. 2012). For this purpose, conditional random fields (CRFs) are often used, as they provide an efficient and flexible framework for including both spatial, temporal, and multi-modal relationships.

In this paper, we apply semantic segmentation on multiple modalities (lidar and camera) for obstacle detection in agriculture. Unlike object detection (such as detecting cars, pedestrians, and cyclists), semantic segmentation can capture objects that are not easily delimited by bounding boxes (e.g. ground, vegetation, sky). We adapt the offline fusion algorithm of Namin et al. (2015) and adjust it for online applicable obstacle detection in agriculture with a moving ground vehicle. This involves explicitly handling sparse point cloud data and exploiting both spatial, temporal, and multi-modal links between corresponding 2D and 3D regions. We combine appearance- and geometry-based detection methods by probabilistically fusing lidar and camera sensing using a CRF. Visual information from a color camera serves to classify visually distinctive regions, whereas geometric information from a lidar serves to distinguish flat, traversable ground areas from protruding elements. We further investigate a traditional computer vision pipeline and deep learning, comparing the influence on sensor fusion performance. The proposed method is evaluated on a diverse dataset of agricultural orchards (mangoes, lychees, custard apples, and almonds) and a dairy paddock gathered with a perception research robot. The dataset is made publicly available and can be downloaded from `http://data.acfr.usyd.edu.au/ag/obstacles/`.

The technical novelty of the paper lies with the introduction of temporal links in the CRF, making the inference recursive. Additionally, because the application of the framework is new within agriculture, the paper also presents a thorough evaluation in a range of different agricultural domains. The main contributions of the paper are therefore fourfold:

- Adaptation of an offline sensor fusion method used for scene analysis to an online applicable method used for obstacle detection. This involves extending the framework with temporal links between successive frames, utilizing the localization system of the robot.
- Comparison of sensor fusion performance when using traditional computer vision and deep learning.
- Comprehensive evaluation of multi-modal obstacle detection in various agricultural environments. This involves detailed comparisons of single- vs. multi-modality performance, binary vs. multiclass classification, and domain adaptation vs. two domain training strategies.
- Publicly available datasets including calibrated and annotated images, point clouds, and navigation data. The datasets target multi-modal object detection in robotics and allow for testing domain adaptation across a range of different agricultural domains.

The paper is divided into 5 sections. Section 2 presents the proposed approach including initial classifiers for the camera and the lidar, individually, and a CRF for fusing the two modalities. Section

3 presents the experimental platform and datasets, followed by experimental results in section 4. Ultimately, section 5 presents a conclusion and future work.

2 Approach

Our method works by jointly inferring optimal class labels of 2D segments in images and 3D segments in corresponding point clouds. By first training individual, initial classifiers for the two modalities, we use a CRF for combining the information using the perspective projection of 3D points onto 2D images. This provides pairwise edges between 2D and 3D segments, thus allowing one modality to correct the initial classification result of the other. Clustering of 2D pixels into 2D segments and 3D points into 3D segments is necessary in order to reduce the number of nodes in the CRF graph structure.

A schematic overview of the algorithm is shown in Figure 1. A synchronized image and point cloud are fed into a pipeline, where feature extraction, segmentation and an initial classification are performed for each modality. 3D segments from the point cloud are then projected onto the 2D image, and a CRF is trained to fuse the two modalities. Finally, recursive inference is introduced to the CRF by adding temporal links to the previous frame, utilizing the localization system of the robot.

In the following subsections, the 2D and 3D classifiers are first described individually. The CRF fusion algorithm is then explained in detail.

2.1 2D Classifier

Most approaches combining lidar and camera use traditional computer vision with hand-crafted image features for the initial 2D classification (Douillard et al. 2010; Cadena and Košecká 2016; Namin et al. 2015; Xiao et al. 2015; Zhang et al. 2015; Munoz et al. 2012). However, recent advances with self-learned features using deep learning have outperformed the traditional approach for many applications. In this paper, we therefore compare the two approaches and evaluate their influence when fusing image and lidar data. Results are presented in section 4.3.

The **traditional computer vision** pipeline consists of three steps: the image is first segmented, features are then extracted for each segment, and a classifier is finally trained to distinguish a number of classes based on the features. In our case, we segment the image into superpixels using SLIC (Achanta et al. 2012). Figure 2a shows an example of this segmentation. For each superpixel, average RGB values, GLCM features (energy, homogeneity and contrast) (Haralick et al. 1973) and a histogram of SIFT features (Lowe 2004) are extracted. The histogram of SIFT features uses a bag-of-words (BoW) representation built using all images in the training set. Dense SIFT features are calculated over the image, and a histogram of word occurrences is generated for each superpixel. All features are then normalized by subtracting the mean and dividing by the standard deviation across the training set. Finally, they are used to train a support vector machine (SVM) (Wu et al. 2004) classifier with probability estimates using a one-against-one approach with the libsvm library (Chang and Lin 2011). This provides probability estimates $P_{\text{initial}}\left(x_i^{\text{2D}} \mid \mathbf{z}_i^{\text{2D}}\right)$ of class label x_i^{2D}, given the features $\mathbf{z}_i^{\text{2D}}$ of superpixel i. An example heatmap of an *object* class is visualized in Figure 2b.

In recent years, **deep learning** has been used extensively for various machine learning problems. Especially for image classification and semantic segmentation, convolutional neural networks (CNNs) have outperformed traditional image recognition methods and are today considered state-of-the-art (Krizhevsky et al. 2012; He et al. 2015; Long et al. 2015). In this paper, we use a CNN for semantic segmentation (per-pixel classification) proposed by Long et al. (2015). As we have a very limited amount of training data available, we use a model pre-trained on the PASCAL-Context dataset (Mottaghi et al. 2014). This includes 59 general classes, of which only a few map directly to the 9 classes present in our dataset (*ground*, *sky*, *vegetation*, *building*, *vehicle*, *human*, *animal*, *pole*, and *other*). For the remaining classes, we remap such that all objects (bottle, table, chair, computer, etc.) map to a common *other* class, and all traversable surfaces (grass, ground, floor, road, etc.) map to a common

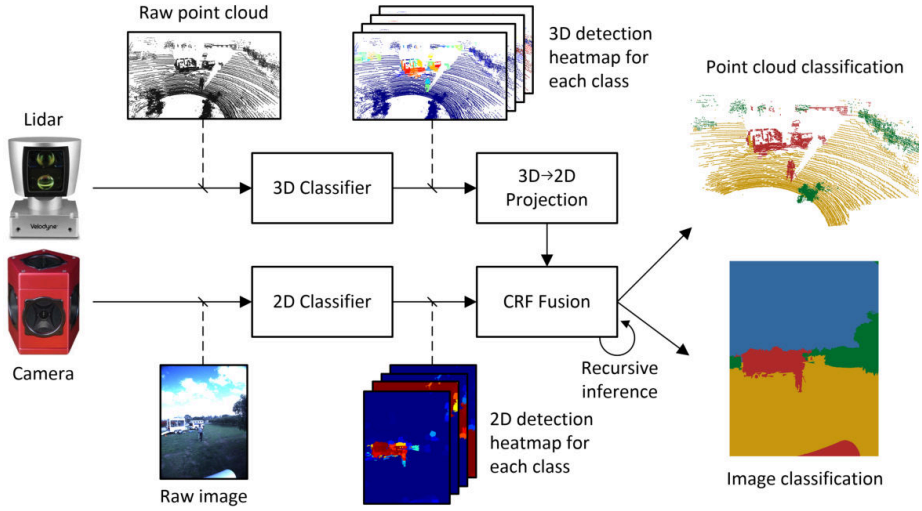
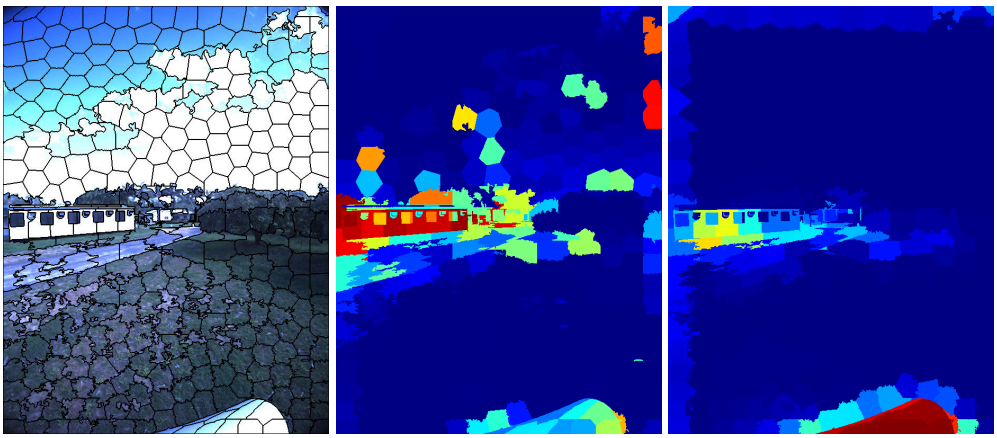


Figure 1. Schematic overview of fusion algorithm.



(a) 2D superpixels **(b)** Traditional vision *object* heatmap **(c)** Deep learning *object* heatmap

Figure 2. Example of 2D segmentation, and probability estimates for traditional vision and deep learning.

ground class. We then maintain the 59 classes of the pre-trained model, and finetune on the overlapping class labels from our annotated dataset. In this way, we preserve the ability of the pre-trained network to

recognize general object classes (humans, buildings, vehicles, etc.), but use our own data for optimizing the weights towards the specific camera, illumination conditions, and agricultural environment used in our setup. From our experiments, this procedure has shown to perform better than simply retraining the last layer of the network from scratch with the agriculturally specific classes present in our dataset.

The softmax layer of the CNN provides per-pixel probability estimates for each object class. However, in this paper, class probability estimates are needed for each superpixel. We therefore use the same superpixel segmentation as for the traditional vision pipeline, and average and normalize per-pixel estimates within each superpixel. An example heatmap of an *object* class is visualized in Figure 2c.

2.2 3D Classifier

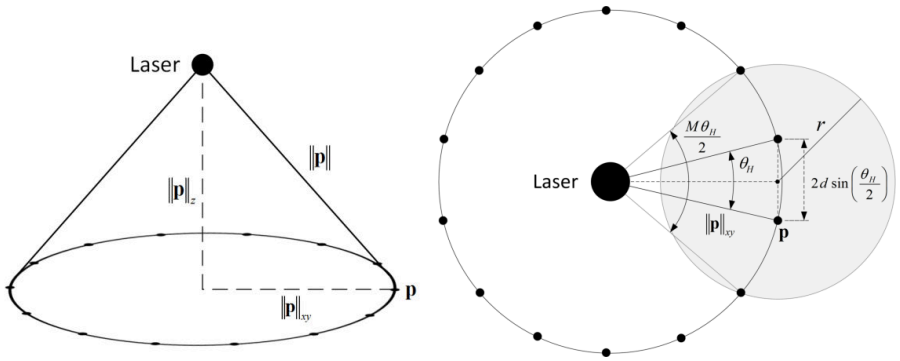
When classifying individual points in a point cloud, the point density and distribution influence the attainable classification accuracy, but also the method of choice for feature extraction. Point features are calculated using a local neighborhood around each point. Traditionally, this is accomplished with a constant neighborhood size (Wellington et al. 2005; Hebert and V 2003; Lalonde et al. 2006; Quadros et al. 2012). For a single-beam laser accumulating points in a push-broom setting, this procedure works fine, as the point distribution is roughly constant, resulting in a dense point cloud. For a rotating, multi-beam lidar generating a single scan, however, the point density varies with distance, resulting in a sparse point cloud. Using a constant neighborhood size in this case, results in either a low resolution close to the sensor or noisy features at far distance. Therefore, in this paper, we use an adaptive neighborhood size depending on the distance between each point and the sensor. This ensures high resolution at short distance and prevents noisy features at far distance. We use the method from Kragh et al. (2015) where the neighborhood size scales linearly with the sensor distance. The intuition behind this relationship assumes a flat ground surface beneath the sensor, such that points from a single, rotating beam pointing towards the ground are distributed equally along a circle. Figure 3a illustrates this circle along with a top-down view in Figure 3b. The radius $\|\mathbf{p}\|_{xy}$ corresponds to the distance in the ground plane between the sensor and a point $\mathbf{p}$. The distance between any two neighboring points on the circle is thus $2\|\mathbf{p}\|_{xy} \sin \frac{\theta_H}{2}$ where θ_H is the horizontal angle difference (angular resolution). In order to achieve a neighborhood (gray area) with M points on a single beam, the neighbourhood radius must be:

$$r = 2\|\mathbf{p}\|_{xy} \sin \frac{M\theta_H}{4} \tag{1}$$

which scales linearly with $\|\mathbf{p}\|_{xy}$. This relationship holds only for single laser beams. However, since the angular resolution for a multi-beam lidar is normally much higher horizontally than vertically, the relationship still serves as a good approximation.

The point cloud is first preprocessed by aligning the xy-plane with a globally estimated plane using the RANSAC algorithm (Fischler and Bolles 1981). This transformation makes the resulting point cloud have an approximately vertically oriented z-axis. Using the adaptive neighborhood, 9 features related to height, shape, and orientation are then calculated for each point (Kragh et al. 2015). f_1-f_4 are height features. f_1 is simply the z-coordinate of the evaluated point, whereas f_2, f_3, and f_4 denote the minimum, mean and variance of all z-coordinates within the neighborhood, respectively. f_5-f_7 are shape features calculated with principal component analysis. As eigenvalues of the 3×3 covariance matrix, they describe the distribution of the neighborhood points (Lalonde et al. 2006). f_8 is the orientation of the eigenvector corresponding to the largest eigenvalue. It serves to distinguish horizontal and vertical structures (e.g. a ground plane and building). Finally, f_9 denotes the reflectance intensity of the evaluated point, provided directly by the lidar sensor utilized in the experiments. Since the size of the neighborhood varies with distance, all features are made scale-invariant.

As for the 2D features, an SVM classifier with probability estimates is trained to provide per-point class probabilities. A segmentation procedure then clusters points into supervoxels by minimizing both spatial distance and class probability difference



(a) Circular scan pattern on flat ground surface. **(b)** Overlaid adaptive neighborhood radius.

Figure 3. Example of adaptive neighborhood radius for single-beam lidar with $M = 4$.

(a) Probability output of *object* class **(b)** Segmented point cloud **(c)** Supervoxel edges

Figure 4. Example of 3D classification, segmentation, and edge construction.

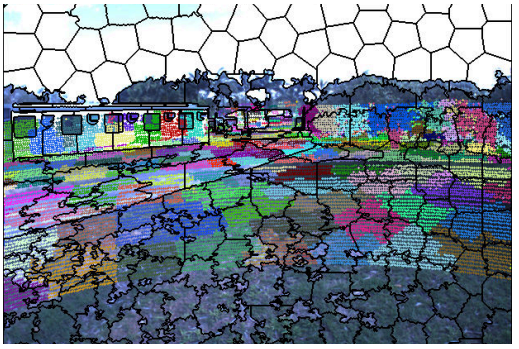


Figure 5. Projection of 3D segments onto 2D superpixels.

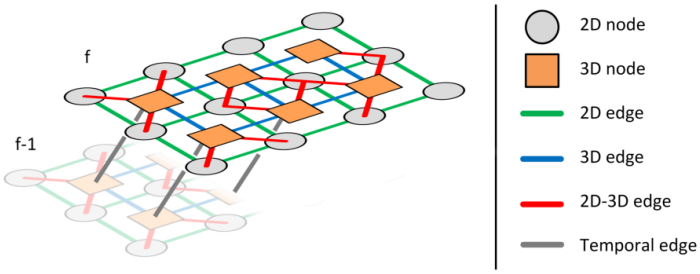


Figure 6. CRF graph with 2D nodes (superpixels), 3D nodes (supervoxels), and edges between them both spatially and temporally.

between segments. Our method uses the approach by Papon et al. (2013) where voxels are clustered iteratively. However, we modify the feature distance measure D between neighboring segments:

$$D = \lambda D_s + \chi^2 \tag{2}$$

where D_s is the spatial Euclidean distance between two segments, χ^2 is the Chi-Squared histogram distance (Pele and Werman 2010) between their mean histograms of probability estimates, and $\lambda > 0$ is a weighting factor. By minimizing this measure during the clustering procedure, points are grouped together based on their spatial distance and initial probability estimates. Each segment i is then given a probability estimate $P_{\text{initial}}\left(x_i^{\text{3D}} \mid \mathbf{z}_i^{\text{3D}}\right)$ by averaging the class probabilities of all points within the segment. Finally, edges between adjacent segments are stored. Figure 4 shows a probability output example of a single class (*object*), the segmented point cloud and its supervoxel edges connecting the segment centers.

Using the extrinsic parameters defining the pose of the lidar and the camera, the point cloud can be projected onto the image using a perspective projection. The extrinsic parameters are given by the solid CAD model of the platform including sensors and refined using an unsupervised calibration method for cameras and lasers (Levinson and Thrun 2013). For computational purposes, the projected points are distorted according to the intrinsics of the camera instead of undistorting the image. Figure 5 illustrates the projected point cloud, pseudo-coloring points by their associated 3D segments. Edges between 2D and 3D segments are then defined by their overlap, such that a large overlap between two segments results in a strong connection, whereas a small overlap results in a weak connection. Single 2D segments can map to multiple 3D segments and vice versa. See section 2.3.2 for further details.

2.3 Conditional Random Field

Once initial probability estimates of all 2D and 3D segments have been found and their edges defined, an undirected graphical model similar to the one visualized in Figure 6 can be constructed. Each 2D and 3D segment (superpixel and supervoxel) is assigned a node in the graph, and edges between the nodes are defined as described in the sections above. In Figure 6, additional recursive edges are shown between frame f and $f-1$. These serve as temporal links between 3D nodes in subsequent frames.

A CRF directly models the conditional probability distribution $p\left(\mathbf{x} \mid \mathbf{z}\right)$, where the hidden variables $\mathbf{x}$ represent the class labels of nodes and $\mathbf{z}$ represent the observations/features. The conditional distribution can be written as:

$$p\left(\mathbf{x} \mid \mathbf{z}\right) = \frac{1}{Z\left(\mathbf{z}\right)} exp\left(-E\left(\mathbf{x} \mid \mathbf{z}\right)\right) \tag{3}$$

where $Z\left(\mathbf{z}\right)$ is the partition (normalization) function and $E\left(\mathbf{x} \mid \mathbf{z}\right)$ is the Gibbs energy. Considering a pairwise CRF for the above graph structure, this

energy can be written as:

$$E(\mathbf{x} \mid \mathbf{z}) = \sum_{i=1}^{N^{2D}} \phi_i^{2D} + \sum_{i=1}^{N^{3D}} \phi_i^{3D} + \sum_{i,j \in E^{2D}} \psi_{ij}^{2D} + \sum_{i,j \in E^{3D}} \psi_{ij}^{3D} + \sum_{i,j \in E^{2D\text{-}3D}} \psi_{ij}^{2D\text{-}3D} + \sum_{i,j \in E^{Time}} \psi_{ij}^{Time} \quad (4)$$

where ϕ_i^{2D} and ϕ_i^{3D} are unary potentials, N^{2D} and N^{3D} are the number of 2D and 3D nodes, ψ_{ij}^{2D}, ψ_{ij}^{3D}, $\psi_{ij}^{2D\text{-}3D}$ and ψ_{ij}^{Time} are pairwise potentials, and E^{2D}, E^{3D}, $E^{2D\text{-}3D}$ and E^{Time} are edges. For simplicity, function variables and weights for the unary and pairwise potentials are left out but explained in more detail in the following sections.

2.3.1 Unary Potentials The unary potentials for 2D and 3D segments are defined by the negative logarithm of their initial class probabilities. This ensures that the conditional probability distribution in equation 3 will correspond exactly to the probability distribution of the initial classifiers if no pairwise potentials are present:

$$\phi_i^{2D}\left(x_i^{2D}, \mathbf{z}_i^{2D}\right) = -log\left(P_{\text{initial}}\left(x_i^{2D} \mid \mathbf{z}_i^{2D}\right)\right) \quad (5)$$

$$\phi_i^{3D}\left(x_i^{3D}, \mathbf{z}_i^{3D}\right) = -log\left(P_{\text{initial}}\left(x_i^{3D} \mid \mathbf{z}_i^{3D}\right)\right) \quad (6)$$

where $\mathbf{z}_i^{2D}$ and $\mathbf{z}_i^{3D}$ are the 2D and 3D features described above, and x_i^{2D} and x_i^{3D} are the class labels. The potentials describe the cost of assigning label x to the i'th 2D or 3D segment. If the probability estimate of the initial classifier is close to 1, the cost is low, whereas if the probability is close to 0, the cost is high.

For unary potentials, no CRF weights are included, since we assume class imbalance to be handled by the initial classifiers.

2.3.2 Pairwise Potentials In equation 4, three different types of pairwise potentials and edges appear. These are 2D edges between neighboring 2D superpixel nodes, 3D edges between neighboring 3D supervoxel nodes, 2D-3D edges connecting 2D and 3D nodes through the perspective projection, and recursive edges connecting subsequent frames.

2D and 3D edges The pairwise potentials for neighboring 2D or 3D segments act as smoothing terms by introducing costs for assigning different labels. As is common for 2D segmentation and classification, the cost depends on the exponentiated distance between the two neighbors, such that a small distance will incur a high cost and vice versa (Boykov and Jolly 2001; Krähenbühl and Koltun 2012). In 2D, the distance is in RGB-space:

$$\psi_{ij}^{2D}\left(x_i^{2D}, x_j^{2D}, \mathbf{z}_i^{2D}, \mathbf{z}_j^{2D}\right) = w_p^{2D}\left(x_i^{2D}, x_j^{2D}\right) \cdot \delta\left(x_i^{2D} \neq x_j^{2D}\right) \exp\left(-\frac{|I_i - I_j|^2}{2\sigma_{2D}^2}\right) \quad (7)$$

where I_i is the RGB-vector for superpixel i and σ_{2D} is a weighting factor trained with cross-validation. $\mathbf{w}_\mathbf{p}^{2D}$ is a weight matrix. It is learned during training and represents the importance of the pairwise potentials. The matrix is symmetric and class-dependent, such that interactions between classes are taken into account. As is common for pairwise potentials, an indicator function (delta function) ensures that the potential is zero for neighboring segments that are assigned the same label.

In 3D, the cost depends on the difference between plane normals (Hermans et al. 2014; Namin et al. 2015):

$$\psi_{ij}^{3D}\left(x_i^{3D}, x_j^{3D}, \mathbf{z}_i^{3D}, \mathbf{z}_j^{3D}\right) = w_p^{3D}\left(x_i^{3D}, x_j^{3D}\right) \cdot \delta\left(x_i^{3D} \neq x_j^{3D}\right) \exp\left(-\frac{|\theta_i - \theta_j|^2}{2\sigma_{3D}^2}\right) \quad (8)$$

where θ_i is the angle between the vertical z-axis and the locally estimated plane normal for supervoxel i and σ_{3D} is a weighting factor trained with cross-validation. The angle is calculated as $\theta = cos^{-1}(f_8)$ (see section 2.2). Similar to 2D, the weight matrix $\mathbf{w}_\mathbf{p}^{3D}$ is symmetric and class-dependent.

2D-3D edges The pairwise potential for 2D and 3D segments connected through the perspective projection is defined by their area of overlap as in Namin et al. (2015). Let S_i^{2D} denote the set of pixels in 2D segment i, and let $S_j^{3D\to 2D}$ denote the set of pixels intersected by the projection of 3D segment

j onto the image. Then, we first define a weight $\omega\left(S_i^{\text{2D}}, S_j^{\text{3D}}\right)$ as the cardinality (number of elements) of the intersection of the two sets:

$$\omega\left(S_i^{\text{2D}}, S_j^{\text{3D}}\right) = \left|S_i^{\text{2D}} \cap S_j^{\text{3D}\to\text{2D}}\right| \tag{9}$$

Effectively, this describes the area of overlap between a 2D segment i and a projected 3D segment j. The pairwise potential is then calculated by normalizing this weight by the maximum weight across all 2D segments that are overlapped by the projected 3D segment j:

$$\psi_{ij}^{\text{2D-3D}}\left(x_i^{\text{2D}}, x_j^{\text{3D}}, \mathbf{z}_i^{\text{2D}}, \mathbf{z}_j^{\text{3D}}\right) = w_p^{\text{2D-3D}}\left(x_i^{\text{2D}}, x_j^{\text{3D}}\right) \cdot \delta\left(x_i^{\text{2D}} \neq x_j^{\text{3D}}\right) \frac{\omega\left(S_i^{\text{2D}}, S_j^{\text{3D}}\right)}{\max\limits_{k \in E_j^{\text{2D-3D}}} \omega\left(S_k^{\text{2D}}, S_j^{\text{3D}}\right)} \tag{10}$$

where k denotes a 2D segment in the set of all edges $E_j^{\text{2D-3D}}$ generated during the projection of 3D segment j onto the image. Using this definition of the pairwise potential between 2D and 3D segments, we introduce a cost of assigning corresponding 2D and 3D nodes with different class labels. The cost depends on the overlap between the segments, such that a large overlap will result in a high cost, and vice versa. The normalization in equation 10 ensures that the weights for associating a 3D node to multiple 2D nodes sums to 1. However, it does not guarantee the opposite. The sum of weights for associating a 2D node to multiple 3D nodes can thus in theory take any positive value.

Similar to 2D and 3D edges, the weight matrix for 2D-3D edges $\mathbf{w}_{\mathbf{p}}^{\text{2D-3D}}$ is class-dependent. However, since the potential concerns different domains (2D and 3D), the weights are made asymmetric as in Winn and Shotton (2006). That is, the cost of assigning x_i^{2D} to class A and x_i^{3D} to class B might not be the same as the other way around. This allows for interactions that depend on both class label and sensor technology.

Recursive edges Recursive inference adds a temporal link from the current frame to a previous frame. By utilizing the localization system of the robot, the location of 3D nodes in a previous frame f_p are transformed from the sensor frame into the world frame. From here, they are then transformed into the current frame f_c where they will likely overlap with the same observed structures. Effectively, this adds another view point to the sensors and can thus help solve potential ambiguities. The extrinsic parameters defining the transformation from the navigation frame (localization system) to the sensor frame (lidar) are given by the CAD model of the platform and refined using an extrinsic calibration method for range-based sensors (Underwood et al. 2010). In the CRF, recursive inference introduces another pairwise potential:

$$\psi_{ij}^{\text{Time}}\left(x_{i,f_c}^{\text{3D}}, x_{j,f_p}^{\text{3D}}, \mathbf{p}_{i,f_c}^{\text{3D}}, \mathbf{p}_{j,f_p}^{\text{3D}}\right) = w_p^{\text{Time}}\left(x_{i,f_c}^{\text{3D}}, x_{j,f_p}^{\text{3D}}\right) \delta\left(x_{i,f_c}^{\text{3D}} \neq x_{j,f_p}^{\text{3D}}\right) \cdot \exp\left(-\frac{\overline{\text{diag}\left(\Sigma_{\text{Nav}}\right)}}{2\sigma_{\text{Nav}}^2}\right) \cdot \exp\left(-\frac{\left|\mathbf{p}_{i,f_c}^{\text{3D}} - T_{f_p}^{f_c}\left(\mathbf{p}_{j,f_p}^{\text{3D}}\right)\right|^2}{2\sigma_{\text{Time}}^2}\right) \tag{11}$$

Here, x_{i,f_c}^{3D} is the label of 3D node i in the current frame f_c and x_{j,f_p}^{3D} is the label of 3D node j in a previous frame f_p. $\overline{\text{diag}\left(\Sigma_{\text{Nav}}\right)}$ is the mean localization variance, calculated as the mean along the diagonal of the localization covariance matrix averaged from frame f_p to f_c. It incorporates the position and orientation variances and is therefore a measure of the localization accuracy. σ_{Nav} is a corresponding weighting factor. $T_{f_p}^{f_c}$ is the transformation from frame f_p to f_c, and σ_{Time} is an associated weighting factor. Both weighting factors are trained with cross-validation.

The transformation is provided by the localization system of the robot. The potential thus depends on the Euclidean distance between a 3D node in the current frame and a transformed 3D node in a previous frame, such that a cost is introduced for assigning different labels at the same 3D location. By also incorporating localization accuracy, a cost is only introduced when localization can be trusted. Only 3D nodes can be transformed, as 2D nodes do not have a 3D position. However, since 3D nodes in a previous

frame are connected with corresponding 2D nodes, 2D information is indirectly carried on to subsequent frames as well. Similar to 2D and 3D edges, the weight matrix for temporal edges $\mathbf{w}_{\mathbf{p}}^{\text{Time}}$ is symmetric and class-dependent.

The obtainable improvement with recursive inference depends on a number of factors. First, the navigation system must be accurate enough to allow reliable transforms of 3D nodes from one frame to another. Second, the time span between frame f_p and f_c must be large enough to actually add another view point to the sensors. If f_p and f_c are too close, the robot will not have moved, and no new information is introduced. However, localization errors can accumulate with distance and time, and therefore f_p and f_c should not be too far apart. Even further, recursive inference assumes that the world is static between frame f_p and f_c. If an object (e.g. human) is moving, errors will accumulate over time.

For training the weight matrix $\mathbf{w}_{\mathbf{p}}^{\text{Time}}$, annotations in 2D and 3D should ideally be available for both frame f_c and f_p. However, this would effectively double the required size of the training set, compared to the other pairwise potentials. As we are only interested in decoding nodes from f_c (and not f_p) during inference, a training procedure utilizing only annotations from the current frame f_c is proposed. All nodes (2D and 3D) from the previous frame f_p are thus unobserved and have unknown labels. In order to allow the likelihood of annotated nodes to be maximized, we marginalize out all unobserved nodes. That is, we sum over all possible classes for each unobserved node, such that the accumulated log likelihood over the entire graph is independent of class labels for unobserved nodes. In practice, this procedure therefore only optimizes nodes in frame f_c, using any information from frame f_p that can increase performance.

2.3.3 Training and Inference During training, the CRF weights $\mathbf{w} = \left[\mathbf{w}_p^{\text{2D}}, \mathbf{w}_p^{\text{3D}}, \mathbf{w}_p^{\text{2D-3D}}, \mathbf{w}_p^{\text{Time}}\right]$ are estimated with maximum likelihood estimation. Additionally, bias weights are introduced for all pairwise terms to account for tendencies independent of the features. To avoid overfitting, we use L_2-regularization for all non-bias weights. Since the graph is cyclic, exact inference is intractable and loopy belief propagation is therefore used for approximate inference. The same applies at test time for decoding. The decoding procedure seeks to determine the most likely configuration of class labels by minimizing the energy $E(\mathbf{x} \mid \mathbf{z})$. The energy can thus be seen as a cost for choosing the label sequence $\mathbf{x}$ given all measurements $\mathbf{z}$.

3 Experimental Platform and Datasets

3.1 Platform

The experimental research platform in Figure 7 has been used to collect data from various locations in Australia. The robotic platform is based on a Segway RMP 400 module and has a localization system consisting of a Novatel SPAN OEM3 RTK-GPS/INS with a Honeywell HG1700 IMU, providing accurate 6-DOF position and orientation estimates. A Point Grey Ladybug 3 panospheric camera system with 6 cameras and a Velodyne HDL-64E lidar both cover a 360° horizontal view around the vehicle recording synchronized images and point clouds.

Since this paper focuses on obstacle detection, only the forward-facing camera and the corresponding overlapping part of the point clouds are used for the evaluation.

3.2 Datasets

From May to December 2013, data were collected across different locations in Australia. The diverse datasets include recordings from both a dairy paddock and orchards with mangoes, lychees, custard apples, and almonds. Figure 8 illustrates a few examples from the forward-facing Ladybug camera during the recordings. Various objects/obstacles such as humans, cows, buildings, vehicles, trees, and hills are present in the datasets. A total of 120 frames have been manually annotated per-pixel in 2D images and per-point in 3D point clouds. By annotating both modalities separately, we can evaluate non-overlapping regions and get reliable ground truth data even if there is a slight calibration error between the two modalities. 9 categories are defined (*ground*, *sky*, *vegetation*, *building*, *vehicle*, *human*, *animal*, *pole*,

Figure 7. Robotic platform "Shrimp" with lidar, panospheric camera, and navigation system.

(a) Mangoes **(b)** Lychees **(c)** Custard apples **(d)** Almonds **(e)** Dairy

Figure 8. Example images from datasets.

Table 1. Dataset overview.

Dataset	Environment	Season	Length	Annotated frames	Annotated 2D/3D segments	Obstacles*
Mangoes	Orchard	Summer	408 m (359 s)	36	12096 / 28001	Buildings, trailer, cars, tractor, boxes, humans
Lychees	Orchard	Summer	122 m (121 s)	15	5040 / 7400	Buildings, trailers, cars, humans, iron bars
Apples	Orchard	Summer	159 m (128 s)	23	7728 / 9708	Trailer, car, humans, poles
Almonds	Orchard	Spring	258 m (212 s)	31	10416 / 33260	Buildings, cars, humans, dirt pile, plate
Dairy	Field	Winter	91 m (106 s)	15	5040 / 18511	Humans, hills, poles, cows

* All frames contain ground and vegetation (trees).

and *other*). Due to the physics of the lidar, *sky* is only present in the images. Table 1 presents an overview of the datasets. The dataset along with all annotations is made publicly available and can be downloaded from http://data.acfr.usyd.edu.au/ag/obstacles/.

4 Experimental Results

To evaluate the proposed algorithm, a number of experiments were carried out on the datasets presented in Table 1. First, the overall results are presented by evaluating the improvement in classification when introducing the fusion algorithm. Then, we specifically address binary and multiclass scenarios, compare traditional vision with deep learning, and evaluate the transferability of features and classifiers across domains (*mangoes*, *lychees*, *apples*, *almonds*, and *dairy*) with domain adaptation. Finally, we compare the performance of domain adaptation and domain training.

To obtain sufficient training examples for each class, the categories *building*, *vehicle*, *human*, *animal*, *pole* and *other* were all mapped to a common *object* class. A total of four classes were thus used for the following experiments, $x_i = \{ground, sky, vegetation, object\}$. For all experiments, 5-fold cross-validation was used corresponding to the 5 different datasets in Table 1. That is, for each dataset, data from the remaining four datasets were used for training initial classifiers and CRF weights. This was done to test the system in the more challenging but realistic scenario, where training data is not available for the identical conditions as where the system would be deployed.

For image classification and CRF training and decoding, we used MATLAB along with the computer vision library VLFeat (Vedaldi and Fulkerson 2008), and the undirected graphical models toolbox UGM (Schmidt 2007). For point cloud classification, we used C++ and Point Cloud Library (PCL) (Rusu and Cousins 2011). A list of parameter settings for all algorithms is available in Appendix A.

4.1 Results Overview

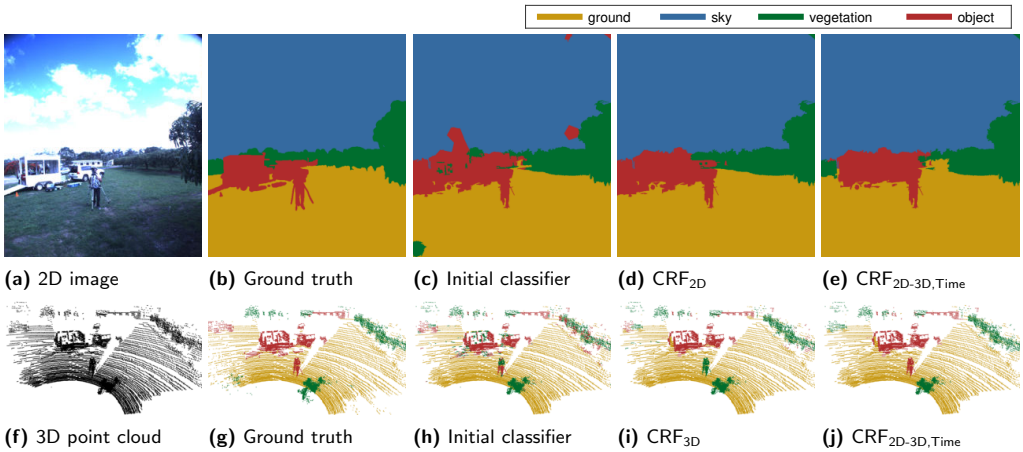
Table 2 presents the results for applying the CRF with the three different types of pairwise potentials enabled. *Initial*, CRF_{2D}, and CRF_{3D} thus refer to single-modality results obtained with the direct output of the initial 2D or 3D classifier and the "smoothed" version of the CRF, respectively. $CRF_{2D\text{-}3D}$ additionally introduces sensor fusion by adding edges across the two modalities, while $CRF_{2D\text{-}3D,Time}$ further adds temporal links across subsequent frames. The results are presented in terms of intersection over union (IoU) and accuracy. Both measures were evaluated per-pixel in 2D and per-point in 3D, thus disregarding the superpixel and supervoxel clusters. Results were obtained with the traditional vision classifier (instead of the deep learning variant) for 2D as it provided the better fusion results. A detailed comparison of traditional vision and deep learning is described in section 4.3.

From Table 2, we see a gradual improvement in classification performance when introducing more terms in the CRF. First, the initial classifiers for 2D and 3D were improved separately by adding spatial links between neighboring segments. This caused an increase in mean IoU of 5.7% in 2D and 7.0% in 3D. Then, by introducing multi-modal links between 2D and 3D, the performance was further increased. In 2D, the increase in mean IoU was only 1.4%, whereas in 3D it amounted to 7.9%. The most prominent increases belonged to the *object* class, where appearance or geometric clues from one modality significantly helped recognize the class in the other modality. Ultimately, adding recursive inference provided the best overall performance. In 2D, a subtle increase in mean IoU of 0.2% was achieved, whereas in 3D, recursive inference caused an increase of 1.5%. The most significant increase was for the *object* class in 3D with an increase in IoU of 3.0%. As recursive inference links 3D nodes between frames, it makes intuitive sense that 3D performance was improved more than 2D.

Figure 9 illustrates an example of a corresponding image and point cloud classified with the initial classifiers and with the CRF. From (c), it is clear that the initial classification of the image was noisy

Table 2. Classification results for 2D and 3D.

	IoU					accuracy
	ground	sky	vegetation	object	mean	
2D, Initial	0.847	0.933	0.729	0.233	0.685	0.900
2D, CRF_{2D}	0.893	**0.971**	0.763	0.342	0.742	0.937
2D, $CRF_{2D\text{-}3D}$	**0.907**	**0.971**	0.774	0.372	0.756	**0.943**
2D, $CRF_{2D\text{-}3D,Time}$	**0.907**	**0.971**	**0.775**	**0.379**	**0.758**	**0.943**
3D, Initial	**0.936**	-	0.735	0.365	0.678	0.881
3D, CRF_{3D}	0.933	-	0.846	0.466	0.748	0.923
3D, $CRF_{2D\text{-}3D}$	0.929	-	0.886	0.667	0.827	0.943
3D, $CRF_{2D\text{-}3D,Time}$	0.933	-	**0.897**	**0.697**	**0.842**	**0.948**



(a) 2D image **(b)** Ground truth **(c)** Initial classifier **(d)** CRF_{2D} **(e)** $CRF_{2D\text{-}3D,Time}$

(f) 3D point cloud **(g)** Ground truth **(h)** Initial classifier **(i)** CRF_{3D} **(j)** $CRF_{2D\text{-}3D,Time}$

Figure 9. Example results. The two rows show 2D and 3D results, respectively.

and affected by saturation problems in the raw image. When introducing 2D edges in the CRF (d), most of these mistakes were corrected. Finally, when combined with information from 3D, the CRF was able to correct *vegetation* and *ground* pixels around the trailer (e). For 3D, some confusion between *vegetation* and *object* occurred in the initial 3D estimate (h), but was mostly solved by introducing 3D edges in the CRF (i). The person in the front of the scene was mistakenly classified as vegetation when using 3D edges, but this was corrected after fusing with information from 2D (j). In some cases, misclassifications in one domain also affected the other. In 2D, sensor fusion introduced a misclassification of the trailer ramp (e), which was seen as *ground* by the initial 3D classifier. Most likely, this happened because the ramp was flat and essentially served the purpose of connecting the ground and the trailer.

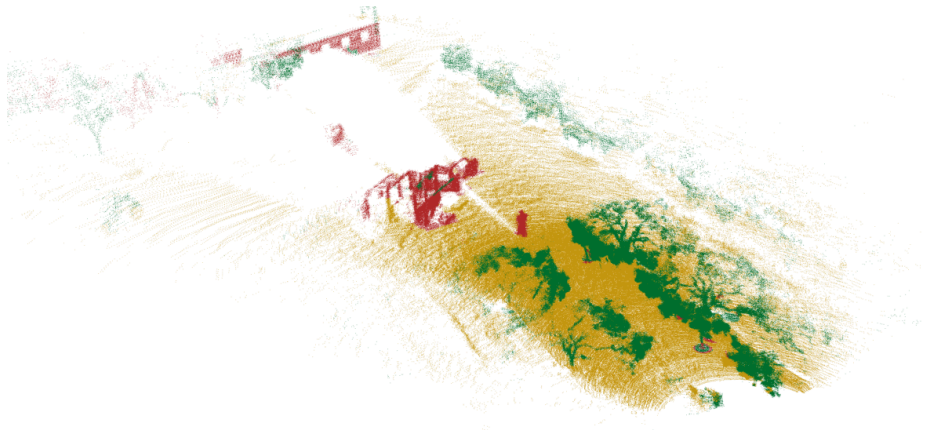


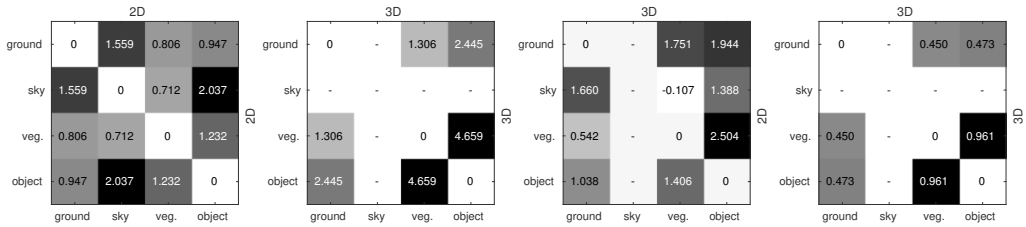
Figure 10. Example of accumulated classification results in 3D of a trajectory.

For the same example section of the dataset presented in Figure 9, Figure 10 illustrates the accumulated classification results in 3D, of a trajectory along the end of a row. This section was chosen as a compact area with many examples of the different classes. The accumulated point cloud was generated by applying the $CRF_{2D\text{-}3D,Time}$ fusion method to each frame and then transforming all 3D points from the sensor frame into the world frame. To generate the figure, the most recent class prediction within any $0.5m$ radius is chosen to represent the region. That is, if a point $\mathbf{p}_1$ was given class label c_1 at time t_1, then this inherited class label c_2 of point $\mathbf{p}_2$ at time t_2 if $|\mathbf{p}_2 - \mathbf{p}_1| \leq 0.5m$ and $t_2 > t_1$. Effectively, this corresponds to always trusting the most recent prediction of the algorithm.

Figure 11 visualizes the learned CRF weights averaged over the 5 cross-validation folds. As explained in section 2.3.2, (a), (b), and (d) are symmetric, whereas (c) is asymmetric. For visualization purposes, we trained the CRF without bias weights, as these would introduce another matrix for each potential and thus make the interpretation of the weights more difficult. Figure 11a shows the weight matrix for neighboring 2D segments. The weights depend on the certainty of the initial classifier and how often adjacent superpixels with different labels appeared in the training set. *ground-object* and *vegetation-sky* appeared often and thus had low weights, whereas *ground-sky* and *object-sky* were rare and therefore were penalized with high weights. Intuitively, this makes sense, as vegetation often separates the ground from the sky in agricultural fields. Figure 9 illustrates how *object* superpixels in the middle of the sky in (c) were corrected by the CRF to *sky* in (d). This was directly caused by a high value of w_p^{2D} (*ground*, *sky*) and multiple adjacent *sky* neighbors.

Figure 11b shows the weight matrix for neighboring 3D segments. Here, the highest weight was for *object-vegetation*. Structurally, these classes were difficult to distinguish with the initial classifier as seen in Figure 9 (h). However, when introducing spatial links in the CRF, most ambiguities were solved as seen in (i).

Figure 11c shows the weight matrix for the 2D-3D fusion. As mentioned in section 2.3.2, the matrix is asymmetric, as we allow different interactions between the 2D and 3D domain. The interpretation of these weights is considerably more complex



(a) $\mathbf{w}_p^{\text{2D}}$ **(b)** $\mathbf{w}_p^{\text{3D}}$ **(c)** $\mathbf{w}_p^{\text{2D-3D}}$ **(d)** $\mathbf{w}_p^{\text{Time}}$

Figure 11. Learned CRF weight matrices averaged over cross-validation folds. High weights correspond to rare occurences and vice versa.

than $\mathbf{w}_p^{\text{2D}}$ and $\mathbf{w}_p^{\text{3D}}$, since the weights incorporate calibration and synchronization errors between the lidar and the camera, and since overlapping 2D and 3D segments intuitively cannot have different class labels. However, a notable outlier was the weight for *sky-vegetation* which was negative. The only apparent explanation for this is a calibration error between the two modalities. Physically, a 2D segment cannot be *sky* if an overlapping 3D segment has observed it. Therefore, label inconsistencies near border regions of *vegetation* and *sky* will cause the CRF weight to decrease.

Figure 11d shows the weight matrix for recursive inference. The weights were all rather small and thus matched the small increase in classification performance when introducing recursive inference. As the weights describe the cost of assigning different labels at the approximate same 3D location, we see the same trend as for neighboring 3D segments in Figure 11b.

4.2 Binary and Multiclass Classification

Due to the physics of the camera and the lidar, the two modalities perceive significantly different characteristics of the environment. The lidar is ideal for distinguishing elements that are geometrically unique, whereas the camera is ideal for distinguishing visual uniqueness. The choice of classes therefore highly affects the resulting improvement with the CRF fusion stage.

In this section, we compare binary and multiclass classification scenarios. The first scenario maps all annotated labels except *ground* to a common *non-ground* class, such that $x_i = \{ground, non\text{-}ground\}$. The second scenario is the same 4-class scenario as presented above. For convenience, the results from Table 9 are replicated in this section.

Table 3 presents the results for the 2D and 3D domains separately. For 2-class classification, the CRF fusion only improved 2D performance, whereas 3D performance actually declined. This is because the geometric classifier (lidar) is good at detecting ground points, and thus can single-handedly distinguish *ground* and *non-ground*. For 4-class classification, however, the CRF fusion introduced improvements in both 2D and 3D. This was caused by the geometric classifier being less discriminative for *vegetation* and *object*, since both classes were represented by obstacles protruding from the ground. Therefore, color and texture cues from the visual classifier could help separate the classes.

4.3 2D Classifiers

As described in section 2.1, a traditional vision pipeline with hand-crafted features was compared to a deep learning approach with self-learned features. Figure 12 compares the two approaches before and after applying the CRF fusion. (a) and (b) show 2D and 3D results for each class, respectively. Filled bars denote initial classification results, whereas hatched bars show classification results after sensor

Table 3. Classification results for binary and multiclass scenarios.

	2-class scenario		4-class scenario	
	mean IoU	accuracy	mean IoU	accuracy
2D, initial	0.914	0.956	0.685	0.900
2D, CRF_{2D}	0.933	0.966	0.742	0.937
2D, $CRF_{2D\text{-}3D}$	**0.938**	**0.969**	0.756	**0.943**
2D, $CRF_{2D\text{-}3D,Time}$	**0.938**	**0.969**	**0.758**	**0.943**
3D, initial	0.927	**0.963**	0.678	0.881
3D, CRF_{3D}	**0.928**	**0.963**	0.748	0.923
3D, $CRF_{2D\text{-}3D}$	0.901	0.949	0.827	0.943
3D, $CRF_{2D\text{-}3D,Time}$	0.900	0.949	**0.842**	**0.948**

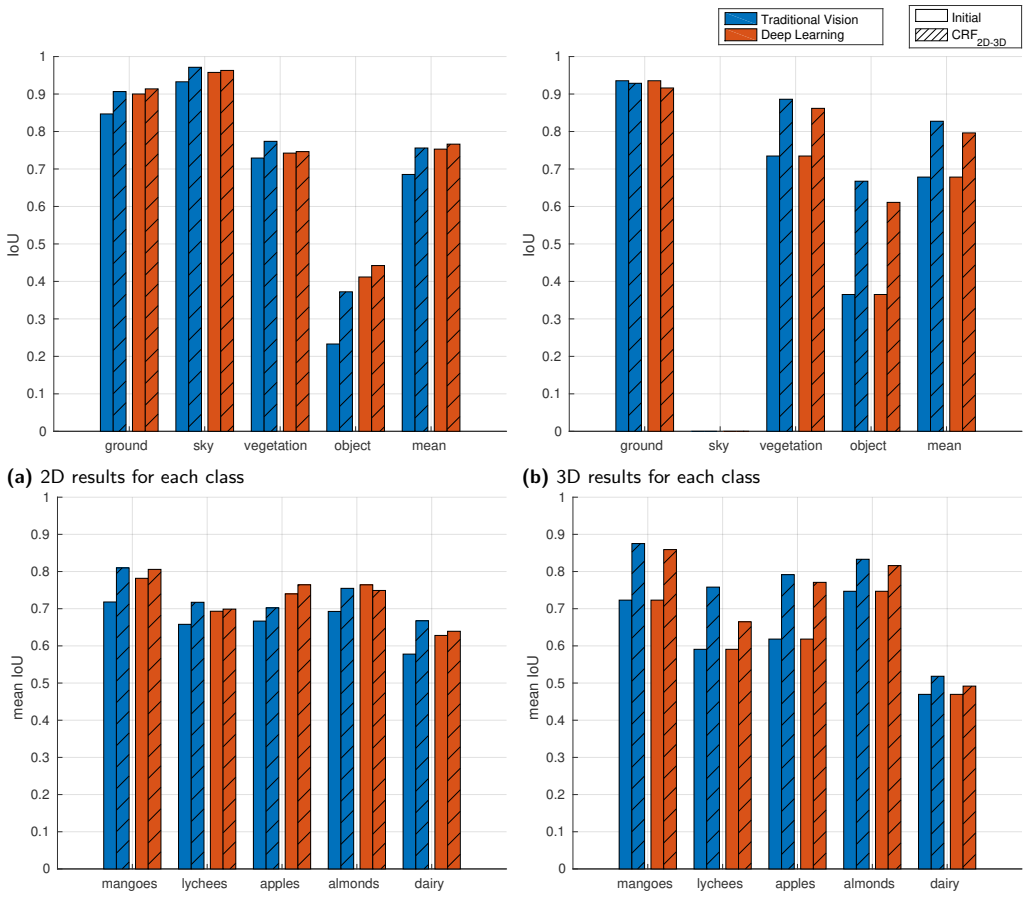
fusion ($CRF_{2D\text{-}3D}$). In Figure 12a, we see that the initial classification results for deep learning were significantly better than for traditional vision with a mean IoU of 75.3% vs. 68.5%. The most significant difference was for the *object* class. Here, deep learning had a clear advantage, since the CNN was pre-trained on an extensive dataset with a wide collection of object categories. When fused with 3D data, however, traditional vision and deep learning reached more similar mean IoUs of 75.6% and 73.6%, respectively. The improvement in classification performance was thus much higher for the traditional vision pipeline than for deep learning. If we look at 3D classification in Figure 12b, the best mean IoU was obtained when fusing with the traditional vision pipeline. Here, a mean IoU of 82.7% was achieved, compared to 79.6% for deep learning. A possible explanation for this is that deep learning is extremely good at recognition, since it uses a hierarchical feature representation and thus incorporates contextual information around each pixel. However, in doing this, a large receptive field (spatial neighborhood) is utilized, which along with multiple max-pooling layers reduces the classification accuracy near object boundaries (Chen et al. 2014). And since the fusion stage of the CRF assumes exact localization in both 2D and 3D, we actually experience a smaller improvement when fusing with deep learning.

Figure 12 (c) and (d) show 2D and 3D results for each dataset, respectively. Here, we see the same tendency that deep learning was superior in 2D in its initial classification for all datasets. However, when fused with 3D data, the two methods basically performed equally well. Traditional vision was better for *lychees* and *dairy*, deep learning was better for *apples*, and they were almost equal for *mangoes* and *almonds*.

To summarize, when evaluating individual performance, deep learning was better than traditional vision. However, when applying a CRF and fusing with lidar, the two methods gave similar results. The CRF was thus able to compensate for the shortcomings in the traditional vision approach.

4.4 Domain Adaptation

In section 4.1, we evaluated the combined classification results over all datasets. In this section, we revisit and break apart these results into separate datasets. In this way, we can evaluate the transferability of features and classifiers across datasets and across classes. Within machine learning and transfer learning, this is generally referred to as domain adaptation. This will allow us to answer a question like: how well do the features and classifiers trained on the combined imagery from *mangoes*, *lychees*, *apples* and *almonds* generalize to recognize a new scenario, such



(a) 2D results for each class

(b) 3D results for each class

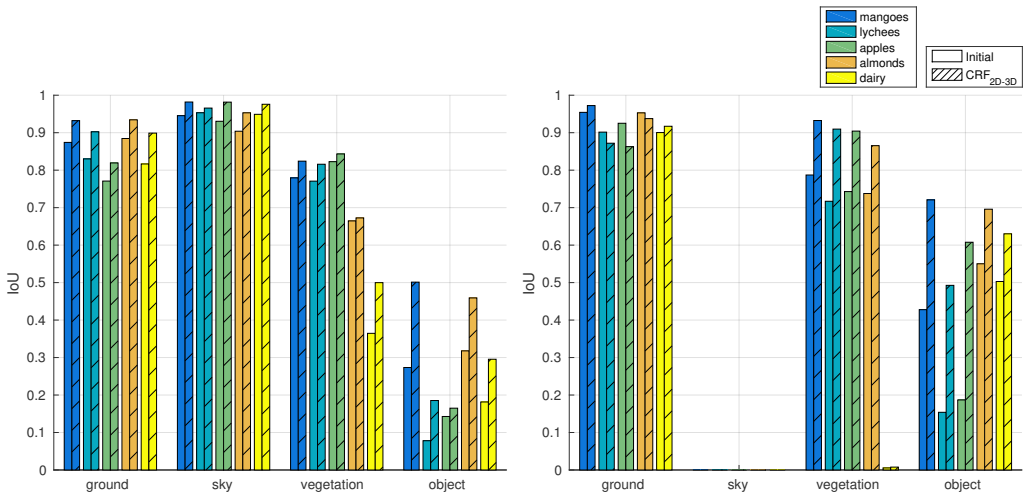
(c) 2D results for each dataset

(d) 3D results for each dataset

Figure 12. Evaluation of traditional vision vs. deep learning before and after sensor fusion.

as *vegetation* in the *dairy* dataset? Figure 13 compares the classification performances in 2D and 3D separately across object classes and datasets. Filled bars denote initial classification results, whereas hatched bars show classification results after sensor fusion ($CRF_{2D\text{-}3D}$).

Figure 13a shows that for 2D, features and classifiers transferred quite well for *ground* and *sky*, possibly due to a combination of limited variation in visual appearance and an extensive amount of training data. However, a larger variation was observed across datasets for *vegetation* and *object*. For the *vegetation* class, the *dairy* dataset had the



(a) 2D **(b)** 3D

Figure 13. Classification results across object classes and datasets before and after sensor fusion.

lowest 2D classification performance. This might be because the mean distance to the tree line was much higher for the dairy paddock than for the orchards, as seen in Figure 8. The visual appearance varies with distance, and especially features describing texture are affected by associated changes in scale and resolution. For the *object* class, a large variation in 2D performance was seen across all datasets. This is most likely due to the large variation in *object* appearances, as the class covered humans, vehicles, buildings, and animals. Also, as listed in Table 1, not all datasets included examples of buildings and animals. Figure 13b shows that for 3D, the features and classifiers transferred well for *ground*, but experienced the same tendencies in variation for *vegetation* and *object* as seen in 2D. For the *vegetation* class, the *dairy* dataset had an IoU close to 0%. This is likely due to the mean distance to the tree line which was outside the range of the lidar. Only a few 3D points within range were labeled *vegetation*, and since the classification performance decreases with distance, most of these were misclassified. For the *object* class, a large variation in 3D performance was seen across all datasets, similar to 2D. However, the initial 3D classifier performed better than 2D, suggesting slightly better transferability for 3D features and classifiers.

Evaluating the transferability of CRF weights, we compared the increase in classification performance across the different datasets (the difference between filled and hatched bars of the same colour in Figure 13). Generally, the CRF weights transferred well across all datasets in both 2D and 3D. However, in 3D, the *ground* class experienced both increases and decreases. Difference in terrain roughness could possibly explain this phenomenon.

To summarize, with minor exceptions, features and classifiers transferred well across the *ground*, *sky*, and *vegetation* classes for all datasets in both 2D and 3D. For these classes, the CRF framework is able to deliver performance increases even when training data is supplied from different environments, which is reasonable given that the appearance of these classes to some degree is independent of the

specific site. For the *object* class, however, features and classifiers transferred poorly in both 2D and 3D, resulting in considerable performance variations across datasets. This was likely caused by limited training data covering the large variation in geometry and appearance within the *object* class, as cows were only present in the *dairy* dataset, tractors in *mangoes*, iron bars in *lychees*, etc.

4.5 Domain Training

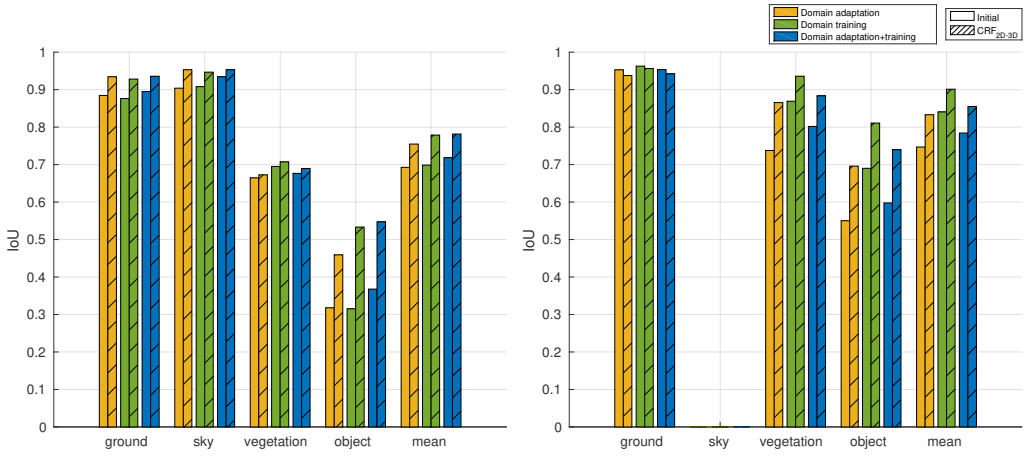
For all the above evaluations, 5-fold cross-validation was used corresponding to the 5 different datasets (domains). That is, when testing on e.g. *apples*, no data from *apples* were used to train the algorithms. In this section, we compare this approach with two less challenging scenarios, where training data are available from the same domain.

As the *almonds* dataset consisted of recordings from two separate days, we split it into *almonds-day1* and *almonds-day2* with 16 and 15 annotated frames, respectively. In the first scenario, we limited the dataset to include *almonds* only. That is, when testing on *almonds-day1*, we trained on the *almonds-day2* dataset, and vice versa. This meant that the training data represented the exact same environment, although captured on a different day. In the second scenario, we combined domain training with domain adaptation. That is, when testing on *almonds-day1*, we trained on the *almonds-day2* dataset plus all the remaining datasets. In this way, a small portion of the training data represented the same environment as the test setup.

Figure 14 shows a comparison of 2D and 3D performance between domain adaptation, domain training, and domain adaptation+training on the *almonds* dataset. Filled bars denote initial classification results, whereas hatched bars show classification results after sensor fusion ($\text{CRF}_{\text{2D-3D}}$). For all methods, we calculated the average performance over the entire *almonds* dataset. Only the training data varied between the three methods. Note that the brown bars for domain adaptation were simply copied from *almonds* in Figure 13 to ease the comparison.

Figure 14a shows that for 2D, the three methods only resulted in minor performance variations for *ground*, *sky*, and *vegetation*. This is a surprising result, as the appearance of both *ground* and *vegetation* in the *almonds* dataset differed quite significantly from the remaining datasets as shown in Figure 8d. Despite this, the 2D classifiers successfully discriminated the classes even when no training data from the specific environment were available (domain adaptation). For the *object* class, however, significant improvements were introduced with the two domain training strategies. For domain training, initial IoU was similar to domain adaptation, while fusion with 3D resulted in an increase of 7.4%. For domain adaptation+training, initial IoU was increased by 5.0%, while fusion with 3D resulted in an increase of 8.8%. Again, this underlines that the large variation in *object* appearances required more training data for the initial classifier. However, fusion with 3D seemed to circumvent this requirement. Therefore, although initial 2D mean IoU was better for domain adaptation+training, 3D fusion compensated for the differences and made both domain training approaches perform equally well.

Figure 14b shows that for 3D, the *ground* class was relatively unaffected by domain training. This is most likely due to the ground geometry of *almonds* being very similar to that of *mangoes*, *lychees*, and *apples*. The *vegetation* and *object* classes, on the other hand, both experienced large improvements, especially for domain training. For the *object* class, domain training increased initial IoU by 14.0%, while fusion with 3D resulted in an increase of 11.5%. Domain adaptation+training, however, gave smaller increases of 4.8% and 4.4%, respectively. The same trend was seen for *vegetation*, where domain training gave increases of 13.1% and 7.1%, whereas domain adaptation+training gave increases of 6.4% and 1.8%. This could be caused by the particular *vegetation* geometry of the *almonds* dataset. From Figure 8d, it is clear that the *almonds* dataset was the only dataset captured during flowering, whereas *mangoes*, *lychees*, and *apples* were all captured during fruit-set. The 3D lidar data therefore varied significantly for *vegetation* due to differences in geometry and 3D point densities. Domain adaptation, without knowledge of the specific geometry of *vegetation*, therefore



(a) 2D **(b)** 3D

Figure 14. Domain adaptation vs. domain training on the *almonds* dataset. Domain training includes training data from *almonds* only, whereas domain adaptation+training additionally includes training data from other domains.

gave the lowest 3D performance. Domain training, on the other hand, gave the best performance, as the 3D classifier was trained specifically on *vegetation* geometry of *almonds* during flowering. Finally, domain adaptation+training was in between. Possibly, adding training data from other domains may have made the features of *vegetation* and *object* less seperable. That is, if the two classes were easily distinguished from a small amount of *almonds* training data, the addition of more (possibly overlapping) feature examples from other domains may have partially contaminated the training set. This could suggest that including training data from the same season (flowering or fruit-set) may be more important for 3D classification than including it from the same environment (*mangoes*, *lychees*, *apples*, or *almonds*).

To summarize, domain training generally showed better performance than domain adaptation. Including training data from the same environment thus gave slightly better 2D performance and considerably better 3D performance. The performance increases were class-dependant, such that classes with large inter-domain variation in appearance and geometry benefited significantly from domain training. Additionally, combining domain adaptation with domain training introduced more training data and could thus potentially improve performance, as was seen in 2D. However, as seen in 3D, the performance could also decrease. This indicates that domain adaptation should only be considered when the feature distributions of the source and target domains are similar. In this context, the specific season of the dataset may be as important as the specific environment.

4.6 Timing

As stated in the introduction, the proposed method is online applicable and thus uses only current and previous information gathered with the perception system of the robot. This contrasts the fusion algorithm of Namin et al. (2015) from which it was adapted, since their method uses information acquired over the entire traversal of the scene. Their method, therefore, does not distinguish between past, present, and future view points.

Using a combination of libraries from MATLAB and C++, our method has been optimized for research flexibility and not processing speed. In order to run the proposed method in real-time, further optimisation effort would be required, which is outside the scope of this paper.

Table 4 lists the average computation times for the processing pipeline. Combining 2D and 3D computations makes the average processing time per frame 8.5 seconds. This is dominated by segmentation and feature extraction in 2D. For 2D segmentation, a GPU implementation of SLIC could be used to reduce the processing time down to ∼20 ms (Ren et al. 2015). Similarly, 2D feature extraction and classification could be sped up by applying an inference-optimized semantic segmentation deep neural network such as Enet (Paszke et al. 2016). For 3D, the order of feature extraction, classification, and segmentation could be changed to perform feature extraction and classification on supervoxels instead of each point. This would significantly speed up feature extraction and classification, although potentially also reduce the accuracy. Finally, CRF inference, which is currently done in MATLAB, could be sped up by using a C++ toolkit. 8.5 seconds in total is thus plausible to be sped up to realtime, by a combination of replacing MATLAB with C++, plus the use of GPU and parallelization.

Table 4. Average computation times per frame for the processing pipeline.

	2D	3D
Segmentation	1.4 s	0.4 s
Feature extraction	4.5 s	0.9 s
Initial classification	0.3 s	0.6 s
$\text{CRF}_{\text{2D-3D,Time}}$	0.4 s	

5 Conclusion

This paper has presented a method for multi-modal obstacle detection by fusing camera and lidar sensing with a conditional random field. Initial 2D (camera) and 3D (lidar) classifiers have been combined probabilistically, exploiting both spatial, temporal, and multi-modal links between corresponding 2D and 3D regions. The method has been evaluated on data gathered in various agricultural environments with a moving ground vehicle.

Results have shown that for a two-class classification problem (ground and non-ground), only the camera leveraged from information provided by the lidar. In this case, the geometric classifier (lidar) could single-handedly distinguish ground and non-ground structures. However, as more classes were introduced (*ground*, *sky*, *vegetation*, and *object*), both modalities complemented each other and improved the mean classification score.

The introduction of spatial, multi-modal, and temporal links in the CRF fusion algorithm showed gradual improvements in the mean intersection over union classification score. Adding spatial links between neighboring segments in 2D and 3D separately, first improved the initial and individual classification results with 5.7% in 2D and 7.0% in 3D. Then, adding multi-modal links between 2D and 3D caused a further improvement of 1.4% in 2D and 7.9% in 3D. And finally, adding temporal links between successive frames caused an increase of 0.2% in 2D and 1.5% in 3D. The method proves that it is possible to reduce uncertainty when probabilistically fusing lidar and camera as opposed to applying each sensor individually. Whether the performance gains justify the complexity of the method will depend on the specific agricultural application, including whether binary ground/non-ground classification is sufficient, or whether multiclass classification is required.

The introduction of temporal links in the CRF caused a smaller improvement than the introduction of spatial and multi-modal links. We believe, however, that the increase is significant and worth reporting, as it extends and improves an offline method from scene analysis to an online applicable method for robotics.

A traditional computer vision pipeline was compared to a deep learning approach for the 2D classifier. It was shown that deep learning outperformed traditional vision when evaluating their individual performances. However, when applying a CRF and fusing with lidar, the two methods gave similar results.

Finally, transferability was evaluated across agricultural domains (*mangoes*, *lychees*, *apples*, *almonds*, and *dairy*) and classes (*ground*, *sky*, *vegetation*, and *object*). Results showed that features and classifiers transferred well across domains for the *ground* and *sky* classes, whereas *vegetation* and *object* were less transferable due to a larger inter-domain variation in appearance and geometry. Adding domain-specific training data confirmed this observation, as classification results of particularly *vegetation* and *object* were further increased.

In situations where scene parsing can benefit from input from different sensor modalities, the paper provides a flexible, probabilistically consistent framework for fusing multi-modal spatio-temporal data. The approach is flexible and may be extended to include additional heterogeneous data sources in future work, including radar, stereo or thermal vision, all of which are directly applicable within the framework.

Funding

This work is sponsored by the Innovation Fund Denmark as part of the project SAFE - Safer Autonomous Farming Equipment (project no. 16-2014-0) and supported by the Australian Centre for Field Robotics at The University of Sydney and Horticulture Innovation Australia Limited through project AH11009 Autonomous Perception Systems for Horticulture Tree Crops. Further information and videos available at: http://sydney.edu.au/acfr/agriculture.

References

Abidine AZ, Heidman BC, Upadhyaya SK and Hills DJ (2004) Autoguidance system operated at high speed causes almost no tomato damage. *California Agriculture* 58(1): 44–47. DOI:10.3733/ca.v058n01p44. URL http://californiaagriculture.ucop.eduhttp://californiaagriculture.ucanr.org/landingpage.cfm?articleid=ca.v058n01p44.

Achanta R, Shaji A, Smith K, Lucchi A, Fua P and Susstrunk S (2012) SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(11): 2274–2282. DOI:10.1109/TPAMI.2012.120. URL http://ieeexplore.ieee.org/document/6205760/.

Asvadi A, Garrote L, Premebida C, Peixoto P and Nunes UJ (2017) Multimodal vehicle detection: fusing 3d-lidar and color camera data. *Pattern Recognition Letters* .

Boykov Y and Jolly MP (2001) Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1. IEEE Comput. Soc. ISBN 0-7695-1143-0, pp. 105–112. DOI:10.1109/ICCV.2001.937505. URL http://ieeexplore.ieee.org/document/937505/.

Brunner C, Peynot T, Vidal-Calleja T and Underwood J (2013) Selective Combination of Visual and Thermal Imaging for Resilient Localization in Adverse Conditions: Day and Night, Smoke and Fire. *Journal of Field Robotics* 30(4): 641–666. DOI:10.1002/rob.21464. URL http://doi.wiley.com/10.1002/rob.21464.

Cadena C and Košecká J (2016) Recursive Inference for Prediction of Objects in Urban Environments. In: *International Symposium on Robotics Research*. pp. 539–555. DOI:10.1007/978-3-319-28872-7_31. URL http://link.springer.com/10.1007/978-3-319-28872-7{_}31.

Chang Cc and Lin Cj (2011) LIBSVM. *ACM Transactions on Intelligent Systems and Technology* 2(3): 1–27. DOI:10.1145/1961189.1961199. URL http://dl.acm.org/citation.cfm?doid=1961189.1961199.

Chen LC, Papandreou G, Kokkinos I, Murphy K and Yuille AL (2014) Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In: *International Conference on Learning Representations*. ISBN 9783901608353, pp. 1–14. URL http://arxiv.org/abs/1412.7062.

Chen X, Ma H, Wan J, Li B and Xia T (2017) Multi-view 3d object detection network for autonomous driving. In: *IEEE CVPR*.

Dima C, Vandapel N and Hebert M (2004) Classifier fusion for outdoor obstacle detection. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 1. IEEE. ISBN 0-7803-8232-3, pp. 665–671 Vol.1. DOI:10.1109/ROBOT.2004.1307225. URL http://ieeexplore.ieee.org/document/1307225/.

Douillard B, Fox D and Ramos F (2010) A Spatio-Temporal Probabilistic Model for Multi-Sensor Multi-Class Object Recognition. In: *Springer Tracts in Advanced Robotics*, volume 66. ISBN 9783642147425, pp. 123–134. DOI:10.1007/978-3-642-14743-2_11. URL http://link.springer.com/10.1007/978-3-642-14743-2{_}11.

Eitel A, Springenberg JT, Spinello L, Riedmiller M and Burgard W (2015) Multimodal deep learning for robust RGB-D object recognition. *IEEE International Conference on Intelligent Robots and Systems* 2015-December: 681–687. DOI:10.1109/IROS.2015.7353446.

Fischler MA and Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24(6): 381–395. DOI:10.1145/358669.358692. URL http://portal.acm.org/citation.cfm?doid=358669.358692.

Haralick RM, Shanmugam K and Dinstein I (1973) Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics* 3(6): 610–621. DOI:10.1109/TSMC.1973.4309314. URL http://www.scopus.com/inward/record.url?eid=2-s2.0-0015680481{&}partnerID=tZ0tx3y1http://ieeexplore.ieee.org/document/4309314/.

Häselich M, Arends M, Wojke N, Neuhaus F and Paulus D (2013) Probabilistic terrain classification in unstructured environments. *Robotics and Autonomous Systems* 61(10): 1051–1059. DOI:10.1016/j.robot.2012.08.002. URL http://dx.doi.org/10.1016/j.robot.2012.08.002http://linkinghub.elsevier.com/retrieve/pii/S0921889012001285.

He K, Zhang X, Ren S and Sun J (2015) Deep Residual Learning for Image Recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition*, volume 7. ISBN 978-1-4673-6964-0, pp. 171–180. DOI:10.3389/fpsyg.2013.00124. URL http://arxiv.org/pdf/1512.03385v1.pdf.

Hebert M and V N (2003) Terrain Classification Techniques From Ladar Data For Autonomous Navigation. In: *In Collaborative Technology Alliances Conference*.

Hermans A, Floros G and Leibe B (2014) Dense 3D semantic mapping of indoor scenes from RGB-D images. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. ISBN 978-1-4799-3685-4, pp. 2631–2638. DOI:10.1109/ICRA.2014.6907236. URL http://ieeexplore.ieee.org/document/6907236/.

Kragh M, Christiansen P, Korthals T, Jungeblut T, Karstoft H and Nyholm Jørgensen R (2016) Multi-Modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. In: *Proceedings of the International Conference on Agricultural Engineering, Aarhus, Denmark*. pp. 1–8.

Kragh M, Jørgensen RN and Pedersen H (2015) Object Detection and Terrain Classification in Agricultural Fields Using 3D Lidar Data. In: *Computer Vision Systems : 10th International Conference, ICVS 2015, Proceedings*, volume 9163. ISBN 9783319209036, pp. 188–197. DOI:10.1007/978-3-319-20904-3_18. URL http://link.springer.com/10.1007/978-3-319-20904-3{_}18.

Krähenbühl P and Koltun V (2012) Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. *Advances in Neural Information Processing Systems 24* (4): 109—-117. URL http://arxiv.org/abs/1210.5644.

Krizhevsky A, Sutskever I and Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L and Weinberger KQ (eds.) *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., pp. 1097–1105. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Laible S, Khan YN and Zell A (2013) Terrain classification with conditional random fields on fused 3D LIDAR and camera data. In: *2013 European Conference on Mobile Robots*. IEEE. ISBN 978-1-4799-0263-7, pp. 172–177. DOI: 10.1109/ECMR.2013.6698838. URL http://ieeexplore.ieee.org/document/6698838/.

Lalonde JF, Vandapel N, Huber DF and Hebert M (2006) Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics* 23(10): 839–861. DOI:10.1002/rob.20134. URL http://doi.wiley.com/10.1002/rob.20134.

Levinson J and Thrun S (2013) Automatic Online Calibration of Cameras and Lasers. In: *Robotics: Science and Systems IX*. Robotics: Science and Systems Foundation. ISBN 9789810739379. DOI:10.15607/RSS.2013.IX.029. URL http://www.roboticsproceedings.org/rss09/p29.pdf.

Long J, Shelhamer E and Darrell T (2015) Fully convolutional networks for semantic segmentation. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. ISBN 978-1-4673-6964-0, pp. 3431–3440. DOI:10.1109/CVPR.2015.7298965. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7298965$\delimiter"026E30F$nhttp://arxiv.org/abs/1411.4038http://ieeexplore.ieee.org/document/7298965/.

Lowe DG (2004) Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60(2): 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94. URL http://portal.acm.org/citation.cfm?id=996342http://link.springer.com/10.1023/B:VISI.0000029664.99615.94.

Milella A, Reina G and Underwood J (2015) A Self-learning Framework for Statistical Ground Classification using Radar and Monocular Vision. *Journal of Field Robotics* 32(1): 20–41. DOI:10.1002/rob.21512. URL http://doi.wiley.com/10.1002/rob.21512.

Milella A, Reina G, Underwood J and Douillard B (2014) Visual ground segmentation by radar supervision. *Robotics and Autonomous Systems* 62(5): 696–706. DOI: 10.1016/j.robot.2012.10.001. URL http://dx.doi.org/10.1016/j.robot.2012.10.001http://linkinghub.elsevier.com/retrieve/pii/S0921889012001789.

Mottaghi R, Chen X, Liu X, Cho NG, Lee SW, Fidler S, Urtasun R and Yuille A (2014) The Role of Context for Object Detection and Semantic Segmentation in the Wild. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. ISBN 978-1-4799-5118-5, pp. 891–898. DOI:10.1109/CVPR.2014.119. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909514.

Munoz D, Bagnell JA and Hebert M (2012) Co-inference for multi-modal scene analysis. In: *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12. Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-642-33782-6, pp. 668–681. DOI:10.1007/978-3-642-33783-3_48. URL http://dx.doi.org/10.1007/978-3-642-33783-3_48.

Namin ST, Najafi M and Petersson L (2014) Multi-view terrain classification using panoramic

imagery and LIDAR. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Iros. IEEE. ISBN 978-1-4799-6934-0, pp. 4936–4943. DOI: 10.1109/IROS.2014.6943264. URL http://www.scopus.com/inward/record.url?eid=2-s2.0-84911498534{&}partnerID=tZOtx3y1http://ieeexplore.ieee.org/document/6943264/.

Namin ST, Najafi M, Salzmann M and Petersson L (2015) A Multi-modal Graphical Model for Scene Analysis. In: *2015 IEEE Winter Conference on Applications of Computer Vision*. IEEE. ISBN 978-1-4799-6683-7, pp. 1006–1013. DOI:10.1109/WACV.2015.139. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7045993http://ieeexplore.ieee.org/document/7045993/.

Papon J, Abramov A, Schoeler M and Worgotter F (2013) Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds. In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. ISBN 978-0-7695-4989-7, pp. 2027–2034. DOI:10.1109/CVPR.2013.264. URL http://ieeexplore.ieee.org/document/6619108/.

Paszke A, Chaurasia A, Kim S and Culurciello E (2016) Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147* .

Pele O and Werman M (2010) The Quadratic-Chi Histogram Distance Family. In: *Lecture Notes in Computer Science*, volume 6312 LNCS. ISBN 3642155510, pp. 749–762. DOI:10.1007/978-3-642-15552-9_54. URL http://link.springer.com/10.1007/978-3-642-15552-9{_}54.

Peynot T, Underwood J and Kassir A (2010) Sensor Data Consistency Monitoring for the Prevention of Perceptual Failures in Outdoor Robotics. In: *Seventh IARP Workshop on Technical Challenges for Dependable Robots in Human Environments Proceedings*. Toulouse, France, pp. 145–152.

Posner I, Cummins M and Newman P (2009) A generative framework for fast urban labeling using spatial and temporal context. *Autonomous Robots* 26(2-3): 153–170. DOI:10.1007/s10514-009-9110-6. URL http://link.springer.com/10.1007/s10514-009-9110-6.

Quadros A, Underwood J and Douillard B (2012) An occlusion-aware feature for range images. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. ISBN 978-1-4673-1405-3, pp. 4428–4435. DOI: 10.1109/ICRA.2012.6225239. URL http://ieeexplore.ieee.org/document/6225239/.

Rao D, Deuge MD, NouraniVatani N, Williams SB and Pizarro O (2017) Multimodal learning and inference from visual and remotely sensed data. *The International Journal of Robotics Research* 36(1): 24–43. DOI:10.1177/0278364916679892. URL https://doi.org/10.1177/0278364916679892.

Reina G, Milella A, Rouveure R, Nielsen M, Worst R and Blas MR (2016a) Ambient awareness for agricultural robotic vehicles. *Biosystems Engineering* 146: 114–132. DOI: 10.1016/j.biosystemseng.2015.12.010. URL http://linkinghub.elsevier.com/retrieve/pii/S1537511015001889.

Reina G, Milella A and Worst R (2016b) LIDAR and stereo combination for traversability assessment of off-road robotic vehicles. *Robotica* 34(12): 2823–2841. DOI:10.1017/S0263574715000442. URL http://www.journals.cambridge.org/abstract{_}S0263574715000442.

Ren CY, Prisacariu VA and Reid ID (2015) gSLICr: SLIC superpixels at over 250Hz. *ArXiv e-prints* .

Rusu RB and Cousins S (2011) 3D is here: Point Cloud Library (PCL). In: *2011 IEEE International Conference on Robotics and*

Automation. IEEE. ISBN 978-1-61284-386-5, pp. 1–4. DOI:10.1109/ICRA.2011.5980567. URL http://pointclouds.org/http://ieeexplore.ieee.org/document/5980567/.

Schmidt M (2007) UGM: A Matlab toolbox for probabilistic undirected graphical models. http://www.cs.ubc.ca/~schmidtm/Software/UGM.html.

Underwood JP, Hill A, Peynot T and Scheding SJ (2010) Error modeling and calibration of exteroceptive sensors for accurate mapping applications. *Journal of Field Robotics* 27(1): 2–20. DOI:10.1002/rob.20315. URL http://doi.wiley.com/10.1002/rob.20315.

Vedaldi A and Fulkerson B (2008) VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/.

Wellington C, Courville A and Stentz AT (2005) Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection. In: *Proceedings of Robotics: Science and Systems*.

Winn J and Shotton J (2006) The Layout Consistent Random Field for Recognizing and Segmenting Partially Occluded Objects. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06)*, volume 1. IEEE. ISBN 0-7695-2597-0, pp. 37–44. DOI:10.1109/CVPR.2006.305. URL http://ieeexplore.ieee.org/document/1640739/.

Wu TF, Lin CJ and Weng RC (2004) Probability Estimates for Multi-class Classication by Pairwise Coupling. *Journal of Machine Learning* 5: 975–1005. DOI:10.1016/j.visres.2004.04.006. URL http://www.csie.ntu.edu.tw/{~}cjlin/papers/svmprob/svmprob.pdf.

Xiao L, Dai B, Liu D, Hu T and Wu T (2015) CRF based road detection with multi-sensor fusion. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*, Iv. IEEE. ISBN 978-1-4673-7266-4, pp. 192–198. DOI:10.1109/IVS.2015.7225685. URL http://ieeexplore.ieee.org/document/7225685/.

Zhang R, Candra SA, Vetter K and Zakhor A (2015) Sensor fusion for semantic segmentation of urban scenes. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. ISBN 978-1-4799-6923-4, pp. 1850–1857. DOI:10.1109/ICRA.2015.7139439. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7139439http://ieeexplore.ieee.org/document/7139439/.

Zhou S, Xi J, McDaniel MW, Nishihata T, Salesses P and Iagnemma K (2012) Self-supervised learning to visually detect terrain surfaces for autonomous robots operating in forested terrain. *Journal of Field Robotics* 29(2): 277–297. DOI:10.1002/rob.21417. URL http://doi.wiley.com/10.1002/rob.21417.

Appendix A: Parameter List

A list of all parameter settings for 2D and 3D classifiers and the CRF fusion framework is available in Table 5.

Table 5. Algorithm parameters used for initial classifiers (2D and 3D) and CRF fusion.

2D classifiers		3D classifier		CRF fusion	
Image		**Point cloud**		**Pairwise potentials**	
width	616	beams	64	σ_{2D}	0.5
height	808	θ_H	0.08°	σ_{3D}	0.5
SLIC		**Feature extraction**		σ_{Nav}	1
region size	40	M	60	σ_{Time}	$1/\sqrt{8}$
regularization factor	3000	**Supervoxels**		time between f_p and f_c	2.0 s
SIFT		seed resolution	0.1		
bin size	3	voxel resolution	0.2		
magnification factor	4.8	λ	1		
BoW		iterations	10		
vocabulary size	50				
fraction of strongest features	0.5				
SVM		**SVM**			
examples	100000	examples	40000		
kernel	RBF	kernel	RBF		
γ	1/57	γ	1/9		
C	1	C	1		
CNN					
learning rate	10^{-12}				
momentum	0.99				
epochs	10				
data augmentation	horizontal flip				

Paper 6

Multi-modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture

Mikkel Fly Kragh, Peter Christiansen, Timo Korthals, Thorsten Jungeblut, Henrik Karstoft, and Rasmus Nyholm Jørgensen

Peer reviewed
Presented at the International Conference on Agricultural Engineering, June 2016, Aarhus, Danmark

Multi-modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture

Mikkel Kragh[a,*], Peter Christiansen[a,*], Timo Korthals[b,*], Thorsten Jungeblut[b], Henrik Karstoft[a], Rasmus N. Jørgensen[a]

[a] Department of Engineering, Aarhus University, Finlandsgade 22, DK-8200 Aarhus N, Denmark

[b] Cognitronics & Sensor Systems, Bielefeld University, Inspiration 1, D-33619 Bielefeld, Germany

* Corresponding author. Email: {mkha, pech}@eng.au.dk, tkorthals@cit-ec.uni-bielefeld.de

Abstract

In recent years, mapping and automation has been increasingly investigated and applied in precision agriculture. The ultimate goal of this development is to apply autonomous vehicles operating efficiently without any human intervention. Such autonomous operation imposes severe safety hazards, demanding accurate and robust risk detection, and avoidance systems. It is unlikely that one sensor can single-handedly guarantee this, and therefore multiple sensing modalities are often combined in order to increase detection performance and introduce redundancy. In this paper, we present a global mapping approach utilizing diverse sensor technologies to achieve a uniform obstacle interpretation of the environment. Using occupancy grid maps, we fuse information from a monocular color camera, a RADAR, and a LIDAR in combination with IMU-assisted GPS-positioning. For each sensor, we present detection algorithms, mapping from raw sensor data to a 2D grid-based obstacle interpretation of the environment. These are then fused temporally with the occupancy grid algorithm, and afterwards spatially in a competitive and complementary way to produce a combined global obstacle map. The method is evaluated on an extensive dataset recorded at Research Centre Foulum, Denmark, in June 2015. The dataset comprises sensor data from a tractor-mounted recording system in a grass mowing scenario with various obstacles. A ground truth map has been obtained with a mapping drone. Results show promising obstacle detection capabilities and an increase in performance when fusing information across sensor modalities and layers. The proposed mapping framework is able to fuse a vast amount of information across a diverse sensor set, using an efficient and novel approach for obstacle detection in agriculture.

Keywords: Multi-modal Sensor Fusion, Obstacle Detection, Occupancy Grid Mapping, Precision Farming, Agriculture

1. Introduction

The application of robots or vehicles operating autonomously in agricultural fields demands extreme perception capabilities of the safety system. It is unlikely that a single perception sensor is capable of ensuring this safety alone, and thus multiple sensor technologies must be combined to provide accurate and robust risk detection and avoidance. These sensors might operate in different coordinate systems with different representations. For instance, a LIDAR operates in 3D cartesian coordinates, an automotive RADAR operates in 2D polar coordinates, and cameras operate in projective spaces of 2D pixel coordinates. Sensor fusion can be handled on various abstraction levels such as data-, feature- or decision-level, but all methods require a mapping to a common representation. One such fusion algorithm on feature-level is occupancy grid maps (Elfes 1990). In 2D, they represent a global map of the environment and are generated from inverse sensor models (ISMs). An ISM is associated with a specific sensor and includes a detection algorithm of a certain feature (e.g. “vehicle”, “human”, “field”, “ground”) and a mapping from sensor data to a local 2D grid in the vehicle frame.

In research on automotive vehicles, 2D grid mapping is widely applied for fusing information across sensing modalities, providing a simple yet efficient framework (Winner 2015). In agricultural environments, a few applications with grid mapping have been proposed as well (Reina and Milella 2012; Ahtiainen et al. 2015). However, these only use a single or two sensing modalities, and thus do not provide a full evaluation of the potential of occupancy grid mapping.

In this paper, we present a global mapping approach utilizing simultaneous information from a monocular color camera, a thermal camera, a RADAR, and a LIDAR in combination with IMU-assisted GPS-positioning. For each of the sensors, we present detection algorithms, mapping from raw sensor data to a 2D grid-based obstacle interpretation of the environment. These grids represent multiple obstacle layers (“human”, “object”, “vegetation”, etc.) and are updated temporally using the occupancy grid algorithm. Finally, they are fused spatially across layers and sensor modalities using competitive and complementary fusion.

2. Materials and Methods

2.1. Setup

A variety of sensor modalities and corresponding detection algorithms are used to ensure detection and provide redundancy for all relevant obstacle types. A Velodyne HDL-32E LIDAR (laser range scanner) is used for long range depth estimation and is robust towards changes in illumination and weather. A Delphi ESR automotive RADAR is used for mid and long range depth and velocity estimation, and is even more robust towards changes in illumination and

weather than the LIDAR. A Logitech C920 color camera is used to detect and distinguish between different obstacle types, but is significantly more sensitive towards changes in illumination. Finally, a thermal camera is useful for capturing heat radiation from humans and animals. However, since only static, non-living obstacles are present in the dataset, this sensor is excluded from the paper. Together, the sensors both complement and overlap each other in terms of detection capabilities and robustness. A Vectornav VN-100 Inertial Measurement Unit (IMU) and a Trimble AG GPS361 Real Time Kinematic (RTK) GPS unit are used for pose estimation. Offline calibration is performed by hand by estimating extrinsic parameters of sensor positions. The specific sensor platform used for the experiments is presented and explained in detail in a previous paper (Christiansen et al. 2015).

2.2. Detection Algorithms

In the following sections, the algorithms used to produce classifications and their conversions to ISMs are described.

2.2.1. LIDAR

A single LIDAR scan provides a 3D point cloud consisting of depth measurements distributed 360° horizontally around the vehicle. For each point, we calculate 13 features using statistics from a local neighborhood (Kragh, Jørgensen, and Pedersen 2015). These features describe the height, shape, orientation and reflectance of the structure and help distinguish between points representing three classes: “ground”, “vegetation”, and “object”. A Support Vector Machine (SVM) classifier with probability estimates (Wu, Lin, and Weng 2004) is then trained to classify individual points into these classes. Figure 1 (left) shows an example of pseudo-colored probability estimates of the “object” class.

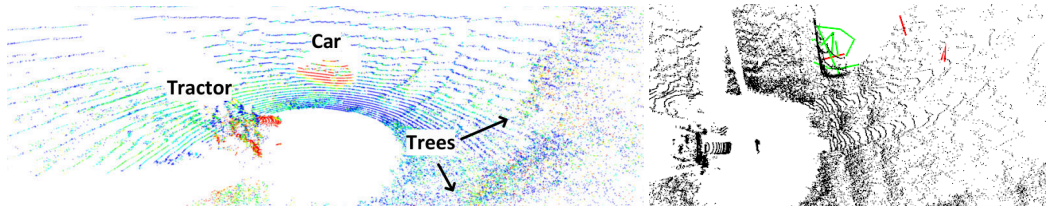


Figure 1. Left: Point cloud with pseudo-colored probability estimates of “object” class illustrating low (blue) and high (red) probabilities. Right: RADAR tracks overlaid on point cloud. Green are confirmed tracks and red are unconfirmed.

2.2.2 RADAR

The automotive RADAR combines mid- and long-range functionality simultaneously, so that it can detect close-distance objects with a horizontal field of view (FOV) of ±45° and far-distance objects with a narrow FOV of ±10°. The RADAR itself provides a processed list of up to 32 tracked objects, each with an angle and a range. However, most of these represent internal noise in the RADAR and therefore need to be processed further. For that, we apply the Kuhn-Munkres assignment algorithm (KMA), tracking detections from subsequent frames (Munkres 1957). Only detections that are less than 2 m apart from one frame to the next are associated. A track i is described by its current position and its track length L_i and is confirmed when $L_i \geq L_{min} = 3$. All confirmed tracks are then converted to detection probabilities:

$$P_{radar,i} = \frac{L_i - L_{min}}{L_i}$$

2.2.3 Color Camera

For the color camera, we apply three detection algorithms; Locally Decorrelated Channel Features for Pedestrian detection (PED) (Nam, Dollár, and Han 2014), You Only Look Once (YOLO) (Redmon et al. 2016), and Fully Convolutional Network for Semantic Segmentation (SS) (Long et al. 2015).

PED is a state-of-the-art pedestrian detector trained on the INRIA dataset (Dalal and Triggs 2005). PED uses three color and seven edge feature channels followed by a local decorrelation step creating 40 decorrelated feature channels. The algorithm uses an AdaBoost (Freund and Schapire 1996) based classifier and detects humans at multiple locations and scales using a speed efficient multiscale sliding window approach.

YOLO is a deep convolutional neural network (CNN) for object detection trained on 20 object classes on the Pascal Visual Object Classes (VOC) dataset (Everingham, Eslami, and Gool 2013). In this work, the 20 objects are mapped to three object classes: “human”, “vehicle”, and “unknown”.

In agriculture, elements such as the field and shelterbelts cannot naturally be delimited by a bounding box as normally provided by object detection algorithms. SS is a semantic segmentation method, meaning that each pixel in the image is classified as an object class. The algorithm is trained to recognize 60 object classes in the PASCAL-Context dataset (Mottaghi et al. 2014). As described in (Christiansen et al. 2016), these element classes can be remapped to a few agricultural classes. In this work, the classes are remapped to “unknown”, “grass”, “ground”, “human”, “shelterbelt”, “vehicle”, and “water”. An example of the outputs from the algorithms described above is presented in two cropped images in Figure 2.

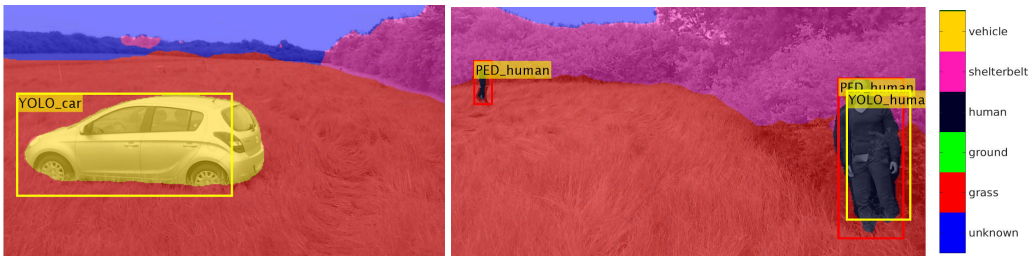


Figure 2. Example output of camera algorithms. PED detects both humans. YOLO is able to detect the vehicle and a human, but fails to detect the more distant human. SS detects both humans, the car, sky, ground and most of the shelterbelt. However, SS fails to detect the shelterbelt at far distance and around the human.

PED and YOLO algorithms output bounding box coordinates that are converted to a new image for each object class with a rectangle filled with a confidence measure of a detection. SS outputs an image for each object class, where each pixel contains a confidence measure of classification.

2.3. Mapping

Within this publication, two challenges are faced by mapping the algorithms' detections into a map representation of the vehicle's environment: First, by locating and mapping the detections into a map, evaluation against a ground truth map is easily applicable. Second, the map representation serves as the common way of fusing detections of a single algorithm temporally, and spatially across different modalities. A technique which suits these requirements is the Occupancy Grid Mapping (OGM).

2.3.1. Occupancy Grid Mapping

Two-dimensional occupancy grids were originally introduced by Elfes (Elfes 1990). In this representation, the environment is subdivided into a regular array or a grid of rectangular cells. The resolution of the environment representation directly depends on the size of the cells. In addition to this discretization of space, a probabilistic measure of occupancy is associated with each cell. This measure takes on any real number in the interval [0, 1] and describes one of the two possible cell states: occupied or unoccupied. An occupancy probability of 0 means definitely unoccupied space, and a probability of 1 means definitely occupied space. A value of 0.5 refers to an unknown state of occupancy.

The occupancy grid is an efficient approach for representing uncertainty, fusing multiple sensor measurements, and to incorporate different sensor models (Winner 2015). To learn an occupancy grid M given sensor information z, different update rules exist (Hähnel 2004). For our approach, we use the Bayesian update rule which is applied to every cell $m \in M$ as follows: Given the positions x_t of the vehicle at each point in time t, suppose $x_{1:t} = x_1, ..., x_t$ are the positions of the vehicle at the individual steps in time, and $z_{1:t} = z_1, ..., z_t$ are the perceptions of the environment. Occupancy probability grids determine for each cell c of the grid the probability that this cell is occupied by an obstacle. Thus, occupancy probability grids seek to estimate

$$P(m|z_{1:T}, x_{1:T}) = \prod_{t=1}^{T} \frac{P(m|z_t, x_t)}{1-P(m|z_t, x_t)} = \prod_{t=1}^{T} Odds(m|z_t, x_t) .$$

This equation already describes the online capable, recursive update rule that populates the current measurement z_t to the grid, where $P(m|z_t, x_t)$ is the so called inverse sensor model (ISM). The ISM is used to update the OGM in a Bayesian framework, which deduces the occupancy probability of a cell, given the sensor information.

2.3.2. Inverse Sensor Modelling

The ISM implements the inverse measurement model, which deduces from the sensor measurement to the occupancy probability at the particular cell. It is commonly used for sensors with a planar sensor lobe oriented parallel to the ground. In that case, a quite simplistic model can be applied, e.g. for a laser range finder. Each cell m that is covered by the beam of the observation z and whose distance to the sensor is shorter than the measured one, is supposed to be unoccupied. The cell in which the beam ends (the measurement point) is supposed to be occupied, and everything behind is unknown (Stachniss 2009). For our implementation, however, the cameras, LIDAR, and RADAR are non-planar, as their sensor lobes are tilted. Every non-planar sensor, compared to planar operating sensors, can only be evaluated at the measurement point, and thus do not provide any information in front of the measurement. Each sensor-algorithm combination requires its own ISM, converting from the algorithm's output to a 2D measurement grid representation. For this, a geometric interpretation is needed in order to transform features from the sensor frame to the vehicle frame.

2.3.2.1 ISM for LIDAR

From the SVM classifier, a 3D point cloud with class probabilities is provided for each class: "ground", "vegetation",

and “object”. A 2D class probability grid is created for each class by projecting all points onto a locally estimated plane and averaging over class probabilities of points lying within a grid cell. From these class probability grids P^*_{class}, two ISM obstacle layers are produced: “object” and “vegetation”. Figure 4 (left) illustrates an example of the “object” layer. The calculation of the log odds ratio of “object” combines the probability of the cell m being an object and the cell not being ground:

$$\begin{aligned} logOdds(P_{object}(m)) &= logOdds(P^*_{object}(m)) + logOdds(1 - P^*_{ground}(m)) \\ &= log(P^*_{object}(m)) - log(1 - P^*_{object}(m)) + log(1 - P^*_{ground}(m)) - log(P^*_{ground}(m)) \end{aligned}$$

2.3.2.2 ISM for RADAR

An ISM obstacle layer “radar” is produced by converting all confirmed detections from polar to cartesian coordinates and averaging over detection probabilities of tracks lying within a grid cell. This provides a probability grid $P^*_{ground}(m)$. The calculation of the log odds ratio of “radar” for cell m is then given by:

$$logOdds(P_{radar}(m)) = logOdds(P^*_{radar}(m)) = log(P^*_{radar}(m)) - log(1 - P^*_{radar}(m))$$

2.3.2.3 ISM for Camera - Inverse Perspective Mapping

Within this chapter, the projection of a camera image onto a planar ground map is described. We assume a pinhole model for the camera, a constant transformation between the camera frame and the vehicle’s footprint, and a flat world. To calculate the pixel-wise transformation from the camera frame into the vehicle frame, the inverse perspective mapping introduced by (Bertozzi and Broggi 1996) is applied.

Because of the flat world assumption, the projection is ill-defined for any detection that does not reside on the ground level. Kohlbrecher bypasses this problem by assuming every detected object to be grounded (Kohlbrecher 2011). In this way, an occupancy grid is generated by traversing through every column of a detection image starting from the bottom. This creates a ray in the occupancy grid, starting at the sensor position towards the horizon. When a detection ($P > 0.5$) occurs along this ray, the given cell is mapped accordingly and all subsequent cells are mapped as unknown ($P = 0.5$).

In this work, a positive detection pixel is extended by the estimated depth of a given obstacle before mapping unknown pixels. Figure 3 illustrates an example of this procedure. In the center image, a positive detection (white blob) of a vehicle seen by the SS algorithm is shown along with the estimated horizon. At the right, the same image converted through inverse perspective mapping to an occupancy grid is visualized, showing how the vehicle is assumed to have a depth of 2 meters.

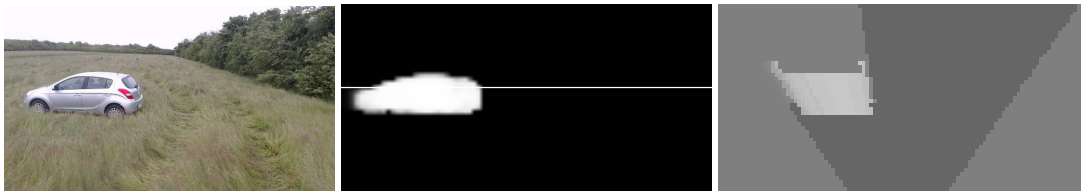


Figure 3. Left: Input image. Center: Horizon and detection of vehicle with semantic segmentation. Right: Inverse perspective mapping showing vehicle, FOV and unknown areas both behind the vehicle and outside the FOV.

2.3.3 Grid Map Representation

Different approaches exist for handling the residency of a map. For spatially limited applications, commonly one global map is used. To reduce the memory consumption, so called topo-metric maps are used as well, where the map size is reduced to e.g. rooms which are interconnected by a graph (Hähnel 2004). For automotive applications, temporary maps have proven their worth. They are build up by different sensors for a short time scenery of the environment (Winner 2015). This paper formulates an independent and global coordinate system which holds multiple two-dimensional grid maps for small areas. The whole area is divided into patches, and for each timestep only one patch, namely the Region-Of-Interest (ROI) is loaded. As depicted in Figure 4 (center and right), the patches overlap at the point where the vehicle crosses the border from the inner to outer ROI to the outer margin. If the vehicle passes this border, a new patch map is loaded. This provides two advantages: First, the memory consumption is reduced to a minimum and second, drift over multiple maps can be reduced by realigning all maps subsequently. Our solution can be compared to the patch map approach by (Konrad et al. 2011). Konrad aligns all maps vertically and horizontally with an overlap at their margins. Compared to this, our approach is able to respect former recorded data by transforming it into the upcoming ROI.

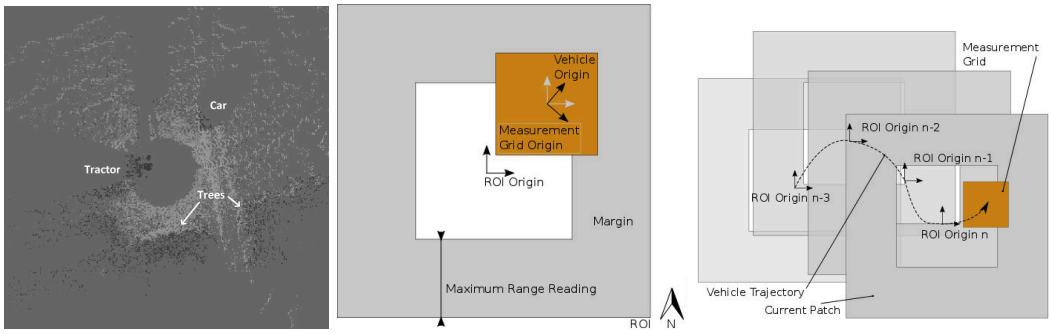


Figure 4. Left: Inverse sensor model as measurement grid of LIDAR for class "object". Center: Current patch as Region-Of-Interest. Right: Overlaid patches along a vehicle's trajectory

2.3.4 Mapping Uncertainty

Every ISM is influenced by the vehicle's pose uncertainty. This includes the latitude and longitude and the roll/pitch/yaw angles. Furthermore, because of the flat-plane assumption, the error caused by the assumed sensor height above the ground is respected as well. All uncertainties of every grid cell are modeled by a two-dimensional Gaussian function. To respect all Gaussian uncertainties in the ISM, all cell neighbours have to be taken into account. Thus, first an ISM without position uncertainties is created and then convolved by a Gaussian kernel $F \in \Re^{IxJ}$. To respect the fact that we deal with probabilities inside the ISM, we define the convolution function P^* for a single probability P of a cell $m_{x,y}$ at point (x,y) in the grid M as follows:

$$P^*(m_{x,y}) = logOdds^{-1} \sum_{i=x-I/2}^{x+I/2} \sum_{j=y-J/2}^{y+J/2} logOdds(F(i,j)(P(m_{i,j}) - 0.5) + 0.5)$$

3. Results and Discussion

3.1. Dataset

The evaluation of the grid mapping is performed on a dataset recorded at Research Centre Foulum, Denmark, in June 2015. The sensor platform described in section 2.1 is mounted in front of a tractor in a grass mowing scenario, recording over a 15 minute traversal in the field. Apart from naturally occurring elements in the field (shelterbelts, grass, ground, and water flooding), static obstacles (wells, a car, barrels, and adult and kid mannequin dolls) are placed and measured with precise GPS positions. The dataset also includes a single moving object (walking pedestrian). A ground truth map is generated by recording the field and obstacles with a Phantom 2 drone and manually annotating with per-pixel labeling. Figure 5 shows the orthophoto of the field with overlaid ground truth annotations.

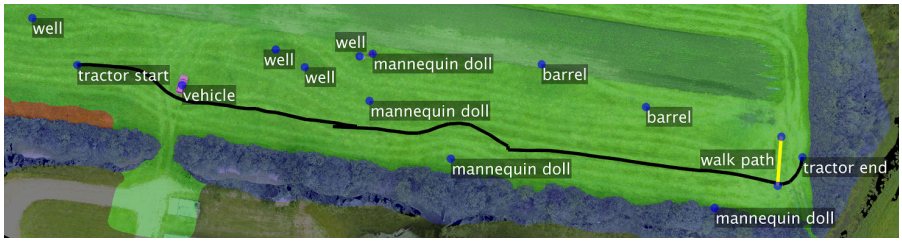


Figure 5. Orthophoto with static objects, tractor trajectory (black line) and human walk path (yellow line). An overlay shows the ground truth of vegetation (blue), ground (green) and non-traversable ground (red).

3.2. Evaluation and Results

To obtain the mapping results, the ISM methods are applied to their specific sensors to extract the measurement grids. To locate the measurement grid inside the current patch and globally, the extended Kalman filter by (Moore and Stouch 2016) is used, taking GPS, IMU, and GPS carrier measurements (Bevly and Cobb 2010) into account. As proposed in (Korthals, Skiba, and Krause 2016), multiple layers N of maps are needed to respect a diverse and heterogeneous sensor setup. This is used to overcome the drawback of the Bayesian update equation, which does not respect different sensor impacts or update rates. Thus, across each of the $N = 15$ sensor-algorithm-class sets, fusion is performed at a later stage by composing cell probabilities. In our implementation, two different fusion techniques are applied: First, the fusion based on a Superbayesian Independent Opinion Pool formula P_B (Pathak et al. 2007). It is applicable for the case when

separate occupancy grids with identical feature representations (e.g. set of maps for class "obstacle") are maintained. Second, a non-Bayesian fusion methods by taking the maximum P_M is applied to heterogeneous feature representations (e.g. set of maps for "vehicle" and "human"). It is worth mentioning that these fusion techniques are again cell-wise and therefore online applicable.

$$P_B(m) = \frac{\prod_N P_n(m)}{\prod_N P_n(m) + \prod_N (1 - P_n(m))}, \qquad P_M(m) = max_n\, P_n(m)$$

As evaluation metrics, precision, recall, F1 score, accuracy, True-Positive-Rate (TPR) and False-Positive-Rate (FPR) of the Receiver-Operator-Characteristic (ROC), and normalized entropy are calculated for all detected cells. For the given algorithms and sensors, the fusion and evaluation scores are not directly applicable. Even if the Bayesian framework allows the representation of the presence and absence of a feature, some algorithms do not make use of it. To name two examples, the LIDAR allows the deduction of free or occupied space based on its physical measurement principle. On the other hand, a camera based algorithm is fairly good for detecting the presence of a class, but easily fails in detecting the absence, due to e.g. a possible lack in the training set. Thus, the metrics recall, F1 score, TPR, and FPR can be calculated for LIDAR based detections, but not for camera and RADAR based detections. To give a better interpretation, the normalized entropy H_N of all true negative and true positive classified cells is used to calculate the remaining uncertainty normalized by a completely unknown map:

$$H(P(M)) = -\sum_{c \in M} P(c)\, log\,(P(c)) + (1 - P(c))\, log\,(1 - P(c)), \qquad H_N(P(M)) = H(P(M))/H(P(M) \equiv 0.5)$$

This gives a quantitative value of the information gain among different setups where the range of the normalized entropy reaches from 0, meaning that there is no unknown space left, to 1, meaning the map is completely unknown.

Layers produced by the same sensor are fused by the maximum method to get a competitive fusion across algorithms, and the outcome of these layers is fused by the Superbayesian method to get a complementary fusion across different sensors as shown in Figure 6.

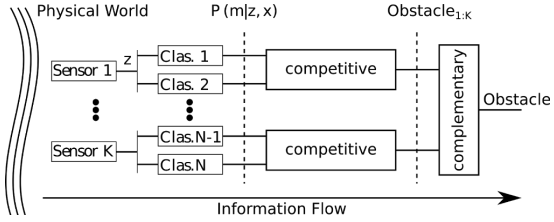


Figure 6. Fusion framework

Table 1. List of sensor setups. 1-3 use competitive fusion across classes, whereas 4-7 use complementary fusion.

Setup	Fusion	Sensors	Detection Algorithm	Input Classes	Output Classes
		Camera	SS	shelterbelt, human, vehicle	
1	Competitive	Camera	YOLO	human, vehicle	obstacle_C
		Camera	PED	human	
2	Competitive	LIDAR	SVM	object, vegetation	obstacle_L
3	Competitive	RADAR	KMA	radar	obstacle_R
4	Complementary	Camera, LIDAR	-	obstacle_C, obstacle_L	obstacle
5	Complementary	LIDAR, RADAR	-	obstacle_L, obstacle_R	obstacle
6	Complementary	Camera, RADAR	-	obstacle_C, obstacle_R	obstacle
7	Complementary	Camera, LIDAR, RADAR	-	obstacle_C, obstacle_L, obstacle_R	obstacle

Table 2. Evaluation scores for the different sensor setups (ill-defined scores omitted by "-")

Setup	Fusion	Precision	Recall	F1 score	Accuracy	TPR	FPR	Entropy
1	Maximum	0.889	-	-	0.889	-	-	0.984
2	Maximum	**0.897**	0.922	0.910	0.957	0.922	**0.0320**	0.821
3	Maximum	0.789	-	-	0.789	-	-	0.991
4	Superbayes	0.896	0.941	0.918	0.960	0.941	0.0342	0.819
5	Superbayes	0.889	0.944	0.916	0.960	0.944	0.0357	0.820
6	Superbayes	0.827	-	-	0.827	-	-	0.979
7	Superbayes	0.889	**0.958**	**0.922**	**0.961**	**0.958**	0.0376	**0.818**

For the evaluation, a constant map resolution of 10 cm per cell is used. To measure the impact of each sensor, all permutations of the sensors (camera, LIDAR, and RADAR) are performed as shown in Table 1. Particularly for the camera based detection, only classes representing objects are taken into account. For setup 1, 2, and 3, the fusion P_M is applied competitively, outputting "obstacle_C", "obstacle_L" and "obstacle_R" for camera, LIDAR, and RADAR respectively. These outputs are then fed into the complementary fusion P_B, outputting "obstacle". The results for all different setups are shown in Table 2. The first noticeable fact is the decrease of entropy for every complementary fusion. This shows, that with the introduction of new sources of information, the unknown area is reduced. Thus, the lowest entropy is evaluated for setup 7. The same is the case for the other scores, where setup 7 performs the best. The only exceptions arise for precision and FPR. For precision, the LIDAR performs better, but also has a bad recall resulting in the worst F1 score. This coincides with the FPR, as the number of misclassifications may rise with more sensors coming into play due to the fact, that in the evaluation scenario the sensor lobes do not fully overlap at all positions. Therefore, wrong classifications can not be corrected by sensor fusion.

As can be seen in Figure 7, misclassifications occur mainly at object borders. Due to the fact that the errors are evenly distributed around them, it can be assumed that they are caused by statistical errors from the sensors, the detection algorithm, or the vehicle's position uncertainty. To quantify this error, the standard deviations of all distinctive misclassified regions across obstacle borders are averaged with the result of $\sigma = 0.332$ m. In Figure 8, the final fused detection of all obstacle layers can be seen. To highlight one example, the car is almost perfectly detected with the only exception of the tail. Having in mind that the upper right edge of the car has not been seen by any sensor, the result of the fusion concept is even more convincing.

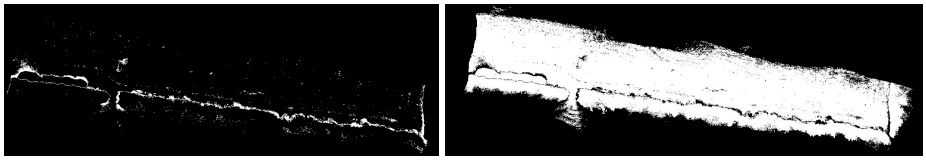


Figure 7. Binary mask created by setup 7 of false (left) and correct (right) classifications

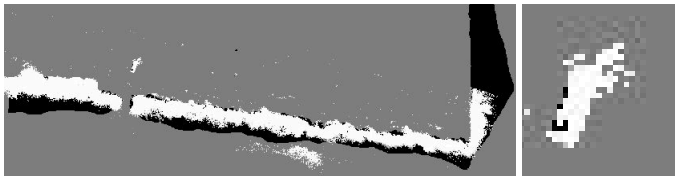


Figure 8. Left: Ground truth (black) with overlaid obstacle detection (white) by setup 7. Right: Magnified area of the car

4. **Conclusions**

In this work, we have presented a global mapping approach fusing information from a monocular color camera, a RADAR, and a LIDAR. For each sensor, we have introduced detection algorithms, mapping from raw sensor data to a number of 2D grid-based obstacle interpretations of the environment, such as "human", "vehicle", and "vegetation". These representations are first fused competitively for each sensor to provide a sensor-specific obstacle representation. Then, complementary fusion is used to fuse across sensor modalities, providing a final combined obstacle interpretation.

Based on data from a grass mowing scenario with various static obstacles, we have evaluated the proposed mapping approach for all combinations of sensors. We have shown that any combination of sensors performs better than the same sensors individually, and that we achieve a mapping accuracy for detected cells of 96% and an F1 score of 92%, when combining information across all three sensors. Future work will focus on introducing dynamic obstacles and training the fusion algorithm to weigh information from sensors and algorithms individually. Also, a more comprehensive evaluation from different fields and sensor setups is planned, investigating generalization performance of the proposed method.

Acknowledgements

This research is sponsored by the Innovation Fund Denmark as part of the project "SAFE - Safer Autonomous Farming Equipment" (project no. 16-2014-0).

This research and development project is funded by the German Federal Ministry of Education and Research (BMBF) within the Leading-Edge Cluster "Intelligent Technical Systems OstWestfalenLippe" (it's OWL) and managed by the Project Management Agency Karlsruhe (PTKA). The author is responsible for the contents of this publication.

References

Ahtiainen, J., T. Peynot, J. Saarinen, S. Scheding, and A. Visala. 2015. "Learned Ultra-Wideband RADAR Sensor Model for Augmented LIDAR-Based Traversability Mapping in Vegetated Environments." In *Information Fusion (Fusion), 2015 18th International Conference on*, 953–60.

Bertozzi, M., and A. Broggi. 1996. "Real-Time Lane and Obstacle Detection on the GOLD System." *Proceedings of Conference on Intelligent Vehicles*. doi:10.1109/IVS.1996.566380.

Bevly, D. M., and S. Cobb. 2010. *GNSS for Vehicle Control*. GNSS Technology and Applications Series. Artech House.

Christiansen, P., M. K. Hansen, K. A. Steen, H. Karstoft, and R. N. Jørgensen. 2015. "Advanced Sensor Platform for Human Detection and Protection in Autonomous Farming." In *Precision Agriculture '15*, 291–98.

Christiansen, P., R. Sørensen, S. Skovsen, C. D. Jæger, R. N. Jørgensen, H. Karstoft, and K. A. Steen. 2016. "Towards Autonomous Plant Production Using Fully Convolutional Neural Networks." In . Aarhus University.

Dalal, Navneet, and Bill Triggs. 2005. "Histograms of Oriented Gradients for Human Detection." In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*. doi:10.1109/CVPR.2005.177.

Elfes, Alberto. 1990. "Occupancy Grids: A Stochastical Spatial Representation for Active Robot Perception." In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*.

Everingham, Mark, Sma Eslami, and Luc Van Gool. 2013. "The Pascal Visual Object Classes Challenge–a Retrospective." *Homepages.Inf.Ed.Ac.Uk*. doi:10.1007/s11263-014-0733-5.

Freund, Yoav, and Robert E. Schapire. 1996. "Experiments with a New Boosting Algorithm." In *ICML*, 96:148–56.

Hähnel, Dirk. 2004. "Mapping with Mobile Robots."

Kohlbrecher, Stefan. 2011. "Grid-Based Occupancy Mapping and Automatic Gaze Control for Soccer Playing Humanoid Robots." *... Humanoid Soccer Robots ...*, no. October.

Konrad, Marcus, Magdalena Szczot, Florian Schüle, and Klaus Dietmayer. 2011. "Generic Grid Mapping for Road Course Estimation." *IEEE Intelligent Vehicles Symposium, Proceedings*, no. Iv: 851–56.

Korthals, Timo, Andreas Skiba, and Thilo Krause. 2016. "Evidenzkarten-Basierte Sensorfusion Zur Umfelderkennung Und Interpretation in Der Ernte." In *Informatik in Der Land-, Forst Und Ernährungswirtschaft*, 15–18.

Kragh, Mikkel, Rasmus N. Jørgensen, and Henrik Pedersen. 2015. "Object Detection and Terrain Classification in Agricultural Fields Using 3D Lidar Data." In *Computer Vision Systems*, 188–97. Lecture Notes in Computer Science. Springer International Publishing.

Long, Jonathan, Long Jonathan, Shelhamer Evan, and Darrell Trevor. 2015. "Fully Convolutional Networks for Semantic Segmentation." In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. doi:10.1109/cvpr.2015.7298965.

Moore, Thomas, and Daniel Stouch. 2016. "A Generalized Extended Kalman Filter Implementation for the Robot Operating System." In *Intelligent Autonomous Systems 13*, edited by E. Menegatti, N. Michael, K. Berns, and H. Yamaguchi, 335–48. Advances in Intelligent Systems and Computing 302. Springer International Publishing.

Mottaghi, Roozbeh, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. 2014. "The Role of Context for Object Detection and Semantic Segmentation in the Wild." In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 891–98. IEEE.

Munkres, James. 1957. "Algorithms for the Assignment and Transportation Problems." *Journal of the Society for Industrial and Applied Mathematics* 5 (1): 32–38.

Nam, Woonhyun, Piotr Dollár, and Joon Hee Han. 2014. "Local Decorrelation For Improved Detection." *Advances in Neural Information Processing Systems*, 1–9.

Pathak, Kaustubh, Andreas Birk, Jann Poppinga, and Sören Schwertfeger. 2007. "3D Forward Sensor Modeling and Application to Occupancy Grid Based Sensor Fusion." *Proceedings of the ... IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE/RSJ International Conference on Intelligent Robots and Systems* 2: 2059–64.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. "You Only Look Once: Unified, Real-Time Object Detection." http://arxiv.org/abs/1506.02640v3.

Reina, Giulio, and Annalisa Milella. 2012. "Towards Autonomous Agriculture: Automatic Ground Detection Using Trinocular Stereovision." *Sensors* 12 (9). Molecular Diversity Preservation International: 12405–23.

Stachniss, Cyrill. 2009. *Robotic Mapping and Exploration*.

Winner, Hermann. 2015. *Handbuch Fahrerassistenzsysteme - Grundlagen, Komponenten Und Systeme Für Aktive Sicherheit Und Komfort*.

Wu, Ting-Fan, Chih-Jen Lin, and Ruby C. Weng. 2004. "Probability Estimates for Multi-Class Classification by Pairwise Coupling." *Journal of Machine Learning Research: JMLR* 5 (December). JMLR.org: 975–1005.

Paper 7

Multi-Modal Detection and Mapping of Static and Dynamic Obstacles in Agriculture for Process Evaluation

Timo Korthals, Mikkel Fly Kragh, Peter Christiansen, Henrik Karstoft, Rasmus Nyholm Jørgensen, and Ulrich Rückert

Peer reviewed
Accepted for publication in Frontiers in Robotics and AI, Research Topic: Multi-modal Sensor Fusion, March 2018

frontiers

Multi-Modal Detection and Mapping of Static and Dynamic Obstacles in Agriculture for Process Evaluation

Timo Korthals [1*], **Mikkel Kragh** [2*], **Peter Christiansen** [2*], **Henrik Karstoft** [2], **Rasmus N. Jørgensen** [2], **and Ulrich Rückert** [1]

[1] *Cognitronics & Sensor Systems, Bielefeld University, Inspiration 1, D-33619 Bielefeld, Germany*
[2] *Department of Engineering, Aarhus University, Finlandsgade 22, DK-8200 Aarhus N, Denmark*

Correspondence*:
Timo Korthals
tkorthals@cit-ec.uni-bielefeld.de

Mikkel Kragh
mkha@eng.au.dk

Peter Christiansen
repetepc@gmail.com

ABSTRACT

Today, agricultural vehicles are available that can automatically perform tasks such as weed detection and spraying, mowing, and sowing while being steered automatically. However, for such systems to be fully autonomous and self-driven, not only their specific agricultural tasks must be automated. An accurate and robust perception system automatically detecting and avoiding all obstacles must also be realized to ensure safety of humans, animals, and other surroundings. In this paper, we present a multi-modal obstacle and environment detection and recognition approach for process evaluation in agricultural fields. The proposed pipeline detects and maps static and dynamic obstacles globally, while providing process-relevant information along the traversed trajectory. Detection algorithms are introduced for a variety of sensor technologies including range sensors (lidar and radar) and cameras (stereo and thermal). Detection information is mapped globally into semantical occupancy grid maps and fused across all sensors with late fusion, resulting in accurate traversability assessment and semantical mapping of process-relevant categories (e.g. crop, ground, and obstacles). Finally, a decoding step uses a Hidden Markov Model to extract relevant process-specific parameters along the trajectory of the vehicle, thus informing a potential control system of unexpected structures in the planned path. The method is evaluated on a public dataset for multi-modal obstacle detection in agricultural fields. Results show that a combination of multiple sensor modalities increases detection performance, and that different fusion strategies must be applied between algorithms detecting similar and dissimilar classes.

*First three authors contributed equally to this work.

Keywords: Occupancy Grid Maps, Mapping & Localization, Obstacle Detection, Precision Agriculture, Sensor Fusion, Multi-Modal Perception, Inverse Sensor Models, Process Evaluation

1 INTRODUCTION

In recent years, autonomous robots and systems have influenced the automation of various agricultural tasks. Numerous scientific approaches have shown that adapting robotic advances can improve workflow, minimize manual labor, and optimize yield. Today, however, conventional scenarios still have the human operator in a centralized position of the farming process, supported by various non-centralized controls units. Due to the global trend in automation, the operator will evidently become an observer in upcoming farming scenarios and to a greater extent manage than operate the process. One key aspect of reaching this goal is to ensure safe operation of driverless systems by perceiving the environment from which potential obstacles are detected and avoided. No sensor can single-handedly guarantee this safety in diverse agricultural environments, and thus a heterogeneous and redundant set of perception sensors and algorithms are needed for this purpose.

Contrary to self-driving cars whose primary purpose is to travel from A to B, an autonomous farming vehicle must also process the traversed area along its way. Common agricultural tasks are harvesting, mowing, pruning, seeding, and spraying. For these tasks, a simple representation of the environment into traversable and non-traversable areas is insufficient. Instead, an agricultural vehicle requires a distinction between e.g. traversable areas like road and soil, and processable areas like grass, crops, and plants. Therefore, obstacle detection in an agricultural context does not simplify to purely identifying objects that protrude from the ground. High grass or crop may appear non-traversable while actually being processable, whereas flat obstacles such as plant seedlings may appear traversable while being non-traversable. A need therefore exists for a system that can detect and recognize a large variety of object categories, while at the same time combine the extensive and perhaps unmanageable amount of information into process-specific parameters relevant for either the driver or an autonomous controller.

This paper presents a multi-modal obstacle and environment detection and recognition approach for process evaluation in agricultural fields. The proposed architecture describes a perception pipeline from data acquisition to classification of process-relevant properties along the vehicle path. Detection algorithms are presented for lidar, radar, stereo camera, and thermal camera, individually. Information from all detections is mapped into a global 2D grid-based representation of the environment and fused across object categories, detection algorithms, and sensor modalities. Finally, relevant properties for processing the field such as traversability and yield information along planned trajectories are decoded. The proposed method is evaluated on a public grass mowing dataset recorded in Lem, Denmark, October 2016. The dataset includes both static and dynamic (moving) obstacles such as humans, vehicles, vegetation, barrels, and buildings as well as structures in the environment such as the grass field and roads.

To the knowledge of the authors, no similar architectures or baselines targeting agricultural applications have previously been published. The proposed architecture therefore represents a novel set of procedures to perform acquisition, detection, fusion, mapping, and process evaluation in a multi-modal setup for an unstructured environment in agriculture. As such, the contributions of the paper are:

- An architecture for multi-modal obstacle and environment detection covering detection algorithms, mapping, fusion across sensors and object classes, and path decoding.

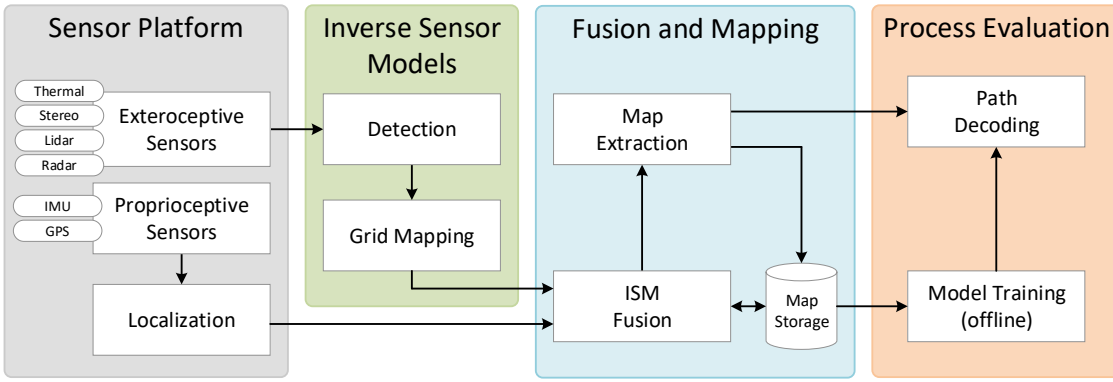


Figure 1. System architecture including information flow.

- A process evaluation method combining mapped environment detections over time into agriculturally relevant properties using a Hidden Markov Model.
- An evaluation on a public agricultural dataset including lidar, radar, stereo camera, and thermal camera sensor data recorded during grass mowing.

The authors' approach extends agricultural technology without replacing current work habits, and allows incorporation of state-of-the-art algorithms for comprehensive environment detection and recognition via an efficient mapping approach. Furthermore, it allows for easy changeability and extendability, which is needed in a daily agricultural scenario. In comparison to model-based or parametrized approaches, the non-parametric two-dimensional occupancy grid mapping has more desirable properties for agricultural scenarios, where mainly the vegetated area is of interest. Analytical solutions as well as relevant heuristics have been applied to build the inverse sensor models (ISM) which incorporate the sensor information as well as its localization.

The proposed architecture is depicted in Figure 1. A sensor platform is mounted on a tractor traversing a field along a preplanned trajectory. A number of exteroceptive sensors collect synchronized perception data used for object detection, whereas proprioceptive sensors are used for global localization of the vehicle. For each sensor modality, an inverse sensor model (ISM) includes an algorithm for detecting a number of object categories (e.g. *human*, *vegetation*, and *building*) and a mapping to align detection information from various algorithms using a 2D occupancy grid map (OGM) representation in the local sensor frame. Detection algorithms include deep learning methods for object detection, semantic segmentation, and anomaly detection on color images, dynamic thresholding on thermal images, point-wise feature extraction and classification of lidar point clouds, and tracking of radar detections. In the fusion and mapping step, OGMs for all sensors and object categories are first localized globally and then updated temporally with the occupancy grid map algorithm by late fusion on a decision level. Finally, they are fused spatially to extract a global map of the environment. We present both binary (occupied/unoccupied) and semantical (object category-specific) maps, allowing further processing in subsequent algorithms. A final decoding step operates on the fused semantical maps and applies a Hidden Markov Model to extract relevant process-specific parameters (e.g. harvesting, mowing, or weed-spraying) along the predefined trajectory of the vehicle. The final output could be used to alert a driver with human-understandable information, or directly by a control system for completely autonomous operation.

The paper is divided into 6 sections. Section 2 introduces related work on obstacle detection in agricultural applications. Section 3 presents the proposed method consisting of each of the four building blocks from Figure 1. Section 4 presents the experimental dataset and results for static and dynamic obstacle and environment detection as well as decoding of process-relevant parameters. Section 5 provides a discussion of the overall approach, while section 6 concludes the paper and suggests future work.

2 RELATED WORK

Robotic automation is emerging for numerous agricultural tasks. The main objective is to reduce production costs and manual labor, while increasing yield and raising product quality (Luettel et al., 2012; Bechar and Vigneault, 2017). A significant milestone is to make robots navigate autonomously in dynamic, rough, and unstructured environments, such as agricultural fields or orchards. To some extent, this has been possible for around two decades with automated steering systems utilizing global navigation systems (Abidine et al., 2004). To eliminate the need for a human operator, however, strict safety precautions are required including accurate and robust risk detection and obstacle avoidance.

Today, only small robots are commercially available that incorporate obstacle avoidance and operate fully autonomously in various agricultural domains (Lely, 2016; Harvest Automation, 2012). Commercialized self-driving tractors or harvesters, however, currently only exist as R&D projects (Case IH, 2016; ASI, 2016; Kubota, 2017).

In scientific research, the concept of an autonomous farming vehicle with obstacle avoidance dates back to 1997 where a camera was used as an anomaly detector to identify structures different from crop (Ollis and Stentz, 1997). Since then, several systems have been proposed for detecting and avoiding obstacles (Cho and Lee, 2000; Stentz et al., 2002; Griepentrog et al., 2009; Moorehead et al., 2012; Emmi et al., 2014; Ball et al., 2016).

A simplified representation of the environment into traversable and non-traversable regions is common for autonomous navigation (Papadakis, 2013). A path may be non-traversable if it is blocked by obstacles, or if the terrain is too rough or steep. Similarly, anomaly or novelty detection is used to find anything that does not comply with normal appearance, and is thus used to detect obstacles (Sofman et al., 2010; Ross et al., 2015; Christiansen et al., 2016a). However, for many agricultural tasks such as harvesting, mowing and weed spraying, further distinction between obstacles and traversable vegetation is necessary. In one application, apparent obstacles such as crops or high grass may be traversable, whereas in another, small plants at ground level may represent obstacles and thus be non-traversable. Distinction into object, vegetation, and ground is common (Wellington and Stentz, 2004; Lalonde et al., 2006; Bradley et al., 2007; Kragh et al., 2015), whereas a few approaches explicitly recognize classes such as humans, vehicles and buildings (Yang and Noguchi, 2012; Christiansen et al., 2016b).

In the literature, obstacle detection systems often rely on a single sensor modality (Rovira-Mas et al., 2005; Reina and Milella, 2012; Fleischmann and Berns, 2015). These systems, however, are easily affected by varying weather and lighting conditions and thus present single points of failure. Christiansen et al. (2017) discusses advantages and disadvantages of various sensor technologies. For instance, a color camera captures visual information similar to humans and can be used to recognize visually distinctive objects. Similarly, a thermal camera captures heat radiation and can distinguish living obstacles such as humans and animals from the background. However, cameras in general are unable to reliably detect object positions and are easily interfered by direct sunlight and changes in weather conditions. On the other hand, lidar and radar sensors are robust to varying weather and lighting conditions and recognize structural differences

with high precision. However, the lack of visual information only allows for a few distinguishable object classes. Therefore, a safety system must have a heterogeneous and complementary sensor suite with multiple sensing modalities that have an overlapping frustum[1] and complement each other in terms of detection capabilities and robustness. Sensor fusion is the concept of combining information from multiple sources to reduce uncertainty in locality and class affiliation. Early fusion combines raw data from different sensors, whereas late fusion integrates information at decision level. In both cases, sensor data need to be compatible.

Lidar, radar, and stereo cameras are all range sensors operating in the domain of metric 3D coordinates. Lidar and radar have been fused with early fusion using a joint extrinsic calibration procedure (Underwood et al., 2010) and with late fusion for augmented traversability assessment (Ahtiainen et al., 2015). Similarly, lidar and stereo camera have been fused with late fusion for traversability assessment (Reina et al., 2016). Often, a grid-based representation such as occupancy grid maps (Elfes, 1990) is used, allowing simple probabilistic fusion and subsequent path planning on the late fused decision level. Monocular cameras operate in the domain of non-metric pixels and can be fused directly under assumption of negligible parallax errors. Examples are available of color and thermal camera fusion for object detection using both early (Davis and Sharma, 2007) and late (Apatean et al., 2010) fusion.

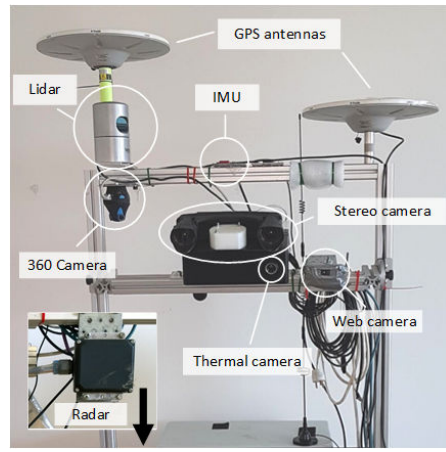
Fusion across domains is possible only when a transformation between them exists. By projecting 3D points onto corresponding 2D images, range sensors can be fused with cameras. With this approach, lidar and color cameras have been combined for semantic segmentation and object recognition using both early (Dima et al., 2004; Wellington et al., 2005; Häselich et al., 2013) and late (Laible et al., 2013; Kragh and Underwood, 2017) fusion. Similarly, image data in pixel-space have been transformed to metric 3D coordinates with inverse perspective mapping (Bertozzi and Broggi, 1998; Konrad et al., 2012). Here, a ground plane assumption is used to invert the perspective effect applied during image acquisition, such that image data are compatible with e.g. lidar and radar data.

In this paper, sensor data from both lidar, radar, stereo camera, and thermal camera are fused with a probabilistic 2D occupancy grid map. This data representation has been chosen as its non-parametric property allows the representation of diffuse agricultural environments. Further, it simplifies path planning and is already a standard in the automotive industrial research (Garcia et al., 2008; Bouzouraa and Hofmann, 2010; Konrad et al., 2012; Winner, 2015). Traditionally, occupancy grid maps represent traversable and non-tranversable areas in a binary decision. The occupancy grid mapping used in this paper, however, is applied in a much richer fashion, due to the extension to multiple semantical layers. Thus, techniques for finding an optimal path, like the A* search algorithm, cannot be directly applied. Further, the finding of an optimal path online in agricultural processes is not mandatory, due to the fact that a full area coverage is aimed, which is inherently defined by the topology and shape of the field. The quantification of the area which lies ahead, and therefore the prediction of process characteristics, is of higher interest. While the direct deduction from the semantical grid maps becomes unfeasible, a so-called decoding for inferring process-relevant information is introduced.

In this work, generative models for inferring process-relevant information out of the mapped sensors' detections are used. Generative models have a number of applications in prediction, missing data imputation or probabilistic inference (Rabiner, 1989; Hinton and Salakhutdinov, 2006). One mathematical framework of generative models is the Hidden Markov Model (HMM) which is able to respect the time-domain and noisy sensor data of a process. Applications to robotics and grid maps have shown the incorporation of

[1]The sensor frustum is the perceptible volume of a sensor, also referred to as the field of view or lobe.

(2a) Platform attached to tractor-mounted mower.



(2b) Sensor setup.

Figure 2. Recording platform. Reprinted from Kragh et al. (2017) with permission.

learning and decoding of hidden property information from the environment which makes HMMs a suitable approach to infer properties out of the semantical grid maps (Stachniss, 2009; Walter et al., 2013; Vasquez et al., 2017).

3 METHOD

In the following, each step from the system architecture in Figure 1 is explained in detail. Section 3.1 describes the recording setup including sensor specifications. Section 3.2 describes the fusion and mapping approach that takes in inverse sensor models and combines these to generate fused obstacle maps. Section 3.3 describes the inverse sensor models, consisting of sensor-specific detection algorithms and transformations to 2D occupancy grid maps. Finally, section 3.4 describes the process evaluation that uses the fused maps to decode process-relevant properties along the trajectories of the tractor.

3.1 Sensor Platform

The sensor suite presented by Kragh et al. (2017) was used to record multi-modal sensor data. The dataset has recently been made publicly available[2]. It includes lidar, radar, stereo camera, thermal camera, IMU, and GNSS[3]. The sensors were fixed to a common platform and interfaced to the Robot Operating System (ROS) (Koubaa, 2016). A tractor-mounted setup and a close-up of the platform are shown in Figure 2.

The exteroceptive sensors and their properties are listed in Table 1. Proprioceptive sensors used for localization included a Vectornav VN-100 IMU and a Trimble BD982 dual antenna GNSS system. All sensors were synchronized in ROS. Lidar, stereo camera, and thermal camera were registered before recording in a semi-automatic calibration procedure (Christiansen et al., 2017). All remaining sensors were registered by hand, by estimating extrinsic parameters of their positions. Global localization from IMU and GNSS was obtained with the robot_localization package (Moore and Stouch, 2014) available in ROS, by simply concatenating the world referenced position and orientation. The overall localization accuracy was

[2] https://vision.eng.au.dk/fieldsafe/

[3] Global Navigation Satellite System

Table 1. Sensors. Adapted from Kragh et al. (2017) with permission.

Sensor	Model	Resolution	FOV (°)	Range (m)	Data rate (fps)
Stereo camera	Multisense S21, CMV2000	1024 x 544	85 x 50	1.5 – 50	10
Web camera	Logitech HD Pro C920	1920 x 1080	70 x 43	n/a	20
360°camera	Giroptic 360cam	2048 x 833	360 x 292	n/a	30
Thermal camera	Flir A65, 13 mm lens	640 x 512	45 x 37	n/a	30
Lidar	Velodyne HDL-32E	2172 x 32	360 x 40	1 – 100	10
Radar	Delphi ESR	32 targets/frame	90 x 4.2 20 x 4.2	0 – 60 0 – 174	20

thus determined by the sensor accuracies of the GNSS ($8\,\mathrm{mm}$ and $15\,\mathrm{mm}$ standard deviations for horizontal and vertical positions, and $< 0.5°$ for yaw) and IMU ($1.0°$ standard deviations for roll and pitch).

3.2 Fusion and Mapping

Occupancy grid maps are used in static obstacle detection for robotic systems, which is a well-known and a commonly studied scientific field (Hähnel, 2004; Thrun et al., 2005; Stachniss, 2009). They are components of almost all navigation and collision avoidance systems designed to maneuver through cluttered environments. Another important application is the creation of obstacle maps for traversing unknown areas and the recognition of known obstacles, thereby supporting localization. Recently, occupancy grid maps have been applied to combine lidar and radar in automotive applications with the goal of creating a harmonious, consistent, and complete representation of the vehicle's environment as a basis for advanced driver assistance systems (Garcia et al., 2008; Bouzouraa and Hofmann, 2010; Winner, 2015).

3.2.1 Occupancy Grid Mapping

Two-dimensional occupancy grid maps (OGM) were originally introduced by Elfes (1990). In this representation, the environment is subdivided into a regular array or a grid of quadratic cells. The resolution of the environment representation directly depends on the size of the cells. In addition to this compartmentalization of space, a probabilistic measure of occupancy is associated with each cell. This measure takes any real number in the interval $[0, 1]$ and describes one of the two possible cell states: unoccupied or occupied. An occupancy probability of 0 represents a space that is definitely unoccupied, and a probability of 1 represents a space that is definitely occupied. A value of 0.5 refers to an unknown state of occupancy.

An occupancy grid is an efficient approach for representing uncertainty, combining multiple sensor measurements at the decision level, and for incorporating different sensor models (Winner, 2015). To learn an occupancy grid M given sensor information z, different update rules exist (Hähnel, 2004). For the authors' approach, a Bayesian update rule is applied to every cell $m \in M$ at position (w, h) as follows: Given the position x_t of a vehicle at time t, let $x_{1:t} = x_1, \ldots, x_t$ be the positions of the vehicle's individual steps until t, and $z_{1:t} = z_1, \ldots, z_t$ the environmental perceptions. For each cell m of the occupancy probability grid $P\left(m|z_{1:t}, x_{1:t}\right)$ represents the posterior probability that this cell is occupied by an obstacle. Thus, occupancy probability grids seek to estimate

$$P\left(m|z_{1:T}, x_{1:T}\right) = \mathrm{Odd}^{-1}\left(\prod_{t=1}^{T} \mathrm{Odd}\left(P\left(m|z_t, x_t\right)\right)\right), \quad \mathrm{Odd}\left(P\left(m|z_t, x_t\right)\right) = \frac{P\left(m|z_t, x_t\right)}{1 - P\left(m|z_t, x_t\right)}. \quad (1)$$

This equation already describes the online capable, recursive update rule that populates the current measurement z_t to the grid, where $P\left(m|z_{1:t}, x_{1:t}\right)$ is the so-called inverse sensor model (ISM). The ISM

is used to update the OGM in a Bayesian framework, which deduces the occupancy probability of a cell, given the sensor information.

3.2.2 Extension to Agricultural Applications

Contrary to robotic or automotive applications, OGM techniques are not directly applicable to agricultural applications. Common applications want to detect non-traversable areas or objects occupying their paths. Such unambiguous information is used to quantify the whole environment sufficiently for all derivable tasks such as path planning and obstacle avoidance. When assumptions like a flat operational plane or minimum obstacle heights are made, the projection of the sensor's frustum to the ground plane is sufficient for all tasks.

In agricultural applications, a crucial task is to quantify the environment as the machines act on and process it. This involves features such as processed areas, processability, crop quality, density, and maturity level in addition to traversability. In order to map these features, single occupancy grid maps are no longer sufficient. Instead, semantical occupancy grid maps (SOGM) that allow different classification results to be mapped are used. Furthermore, sensor frustums are no longer oriented parallel to the ground, but rather oriented at a downward angle to gather necessary crop information (Korthals et al., 2017b).

The extension to SOGM or inference grids is straightforward and defined by an OGM M with W cells in width, H cells in height, and N semantical layers (see Figure 3a):

$$M : \{1, \dots, W\} \times \{1, \dots, H\} \rightarrow m = [0, 1]^N \ . \tag{2}$$

Compared to a single layer OGM which allows the classification into three states $\{$occupied, unoccupied, unknown$\}$, the SOGM supports a maximum of 3^N different states allowing much higher differentiability in environment and object recognition. The corresponding ISMs are fused by means of the occupancy grid map algorithm to their nth associated semantical occupancy grid.

The location of information in the maps is required to be completed by *mapping under known poses* approaches (Thrun et al., 2005). The ISMs are mapped locally in the maps while the maps themselves are globally referenced enabling consistent storing and loading of information. Further, it allows smooth local mapping in the short term without discrete jumps caused by global positioning systems using a Global Navigation Satellite System (GNSS) (Korthals et al., 2017b).

3.2.3 Mapping Capabilities

SOGMs contain a generic representation of the environment. However, for many applications, only part of this vast amount of information is required. Therefore, in the following, we introduce three methods of fusing SOGMs. The first two methods are cell-wise layer fusions given in Equation 3 and 4, while the third method is a cell-clustering technique working across layers given in Equation 5. These are used in the evaluation for binary traversability assessment, class-specific obstacle mapping, and process evaluation.

The first approach introduced in Equation 3 is based on a super Bayesian independent opinion pooling P_{B} (Pathak et al., 2007). It is applicable for the case when separate SOGMs with identical feature representations (same object classes) are maintained. Second, Equation 4 introduces a non-Bayesian maximum pooling fusion method P_{M} is applied to heterogeneous feature representations (varying object classes) (Liggins et al., 2001). The fusion techniques are cell-wise and therefore do not introduce any

clustering:

$$P_{\mathrm{B}}(m) = \frac{1}{1 + \prod_n \frac{1-P(m_n)}{P(m_n)}}, \tag{3}$$

$$P_{\mathrm{M}}(m) = \max_n P(m_n) . \tag{4}$$

Unlike single-layer OGM approaches, an SOGM incorporates multiple OGMs with varying classes residing in the map storage. For many applications cell-wise consideration, which is the disregarding of the cells' surroundings, is not a feasible approach due to noisy or sparse data and potential positional offsets between layers. Thus, clustering on SOGMs was introduced by Korthals et al. (2017a) using a Supercell Extracted Variance Driven Sampling (SEVDS) algorithm, which tends to find clusters that consist of mainly non-contradicting cells:

$$H(c) = D(c) + \Gamma G(c) \text{ with } D(c) = \sum_{n=1}^{N} e_n(\mathrm{var}(h(c))) . \tag{5}$$

In Equation 5, c is the supercell of interest and G is the contour function, which can be smoothed via the scalar factor Γ. The distribution term D of a supercell c is defined as the sum of Eigenvalues e of the covariance matrix of the probability histogram $h(c)$ (see Figure 3b and Figure 3c). The contour term G is taken from Van den Bergh et al. (2015) and evaluates cell-wise updates that penalize irregular shapes, e.g. a single cell extending into an adjacent supercell. A scalar factor of $\Gamma = 1$ is used as in the original paper.

As depicted in Figure 3c, for every found supercell, a triple $\mathcal{C} = (\mathbf{T}_c, \mathbf{L}_c, \mathbf{P}_c)$ consists of its centroid location $\mathbf{T}_c$, a list of adjacent supercells $\mathbf{L}_c$, and a feature vector $\mathbf{P}_c \in \mathbb{R}^N$, with N being the number of SOGM layers. $\mathrm{Odd}(\mathbf{P}_c)$ is calculated as:

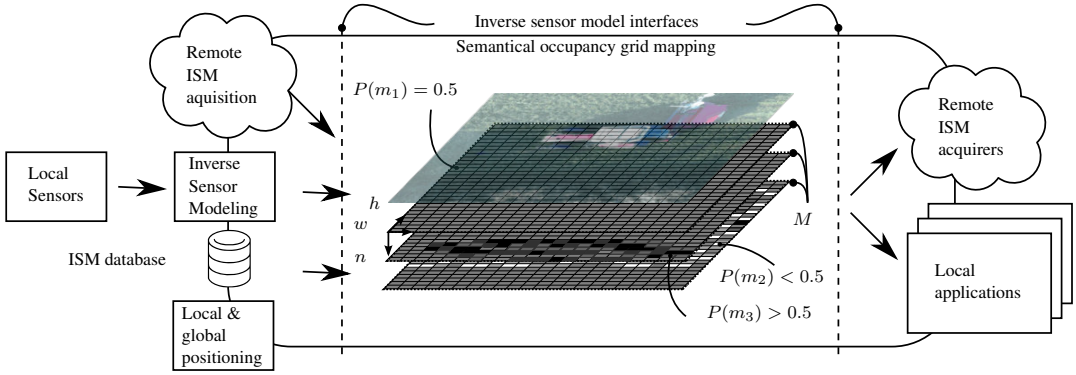
$$\mathrm{Odd}(\mathbf{P}_c) = \left(\prod_{m \in c_1} \mathrm{Odd}(P(m)), \ldots, \prod_{m \in c_N} \mathrm{Odd}(P(m)) \right)^{\mathsf{T}} . \tag{6}$$

3.2.4 Recency Weighting for Dynamic Obstacles

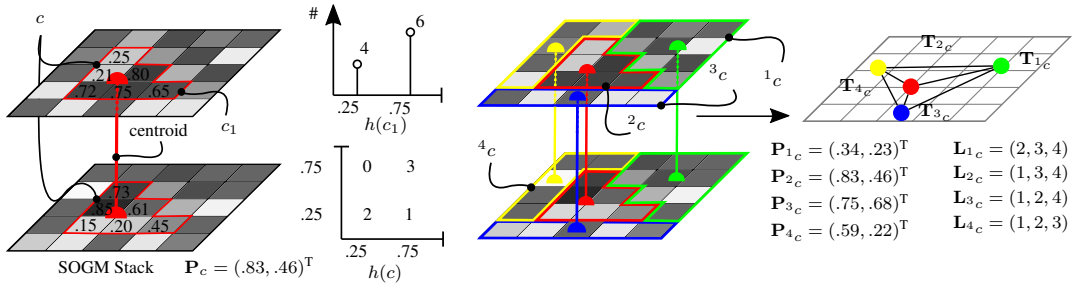
When evaluating the detection dynamic obstacles, static obstacle detections are ignored by introducing recency weighting to the mapserver via two new parameters. A *ForgetValue* indicates the amount of temporal memory in the map. A value of 0 indicates no forgetting, such that all information remains in the map, once it is introduced. A value of 1, however, indicates total forgetting (no memory), such that the map is cleared every time the forgetting is applied. The second parameter is a *ForgetRate* that indicates the rate at which the forgetting is applied. A rate of 2 means that two times every second, all cells in the map are updated with respect to the *ForgetValue*:

$$P(m_t) = (P(m_{t^-}) - 0.5) \cdot (1 - \textit{ForgetValue}) \sum_{n \in \mathbb{N}} \gamma \left(t - \frac{n}{\textit{ForgetRate}} \right) + 0.5 . \tag{7}$$

First, $P(m_{t^-})$ is centralized at 0 where t^- addresses the cell property just before the update. γ indicates the discrete Dirac function which builds up the sampling function with its sampling rate *ForgetRate*. With every forgetting step, the updated posterior probability converges to 0.5 which indicates no knowledge over



(3a) Semantical occupancy grid mapping framework.



(3b) Supercell with $N = 2$ layers and corresponding histograms with $K = 2$ bins.

(3c) Conversion of supercells to a graph of centroids labeled with feature vectors.

Figure 3. Semantical OGM framework and supercell clustering.

the cell m. Thus, Equation 7 is a basic exponential smoothing filter with $P(m_{t-})$ being the start excitation (Biber, 2005).

3.3 Inverse Sensor Models

In the following, individual inverse sensor models (ISM) are introduced and explained in detail for each of the sensors. An ISM consists of an algorithm for detecting a number of object categories and a mapping to align detection information using a 2D occupancy grid map (OGM) in the local sensor frame.

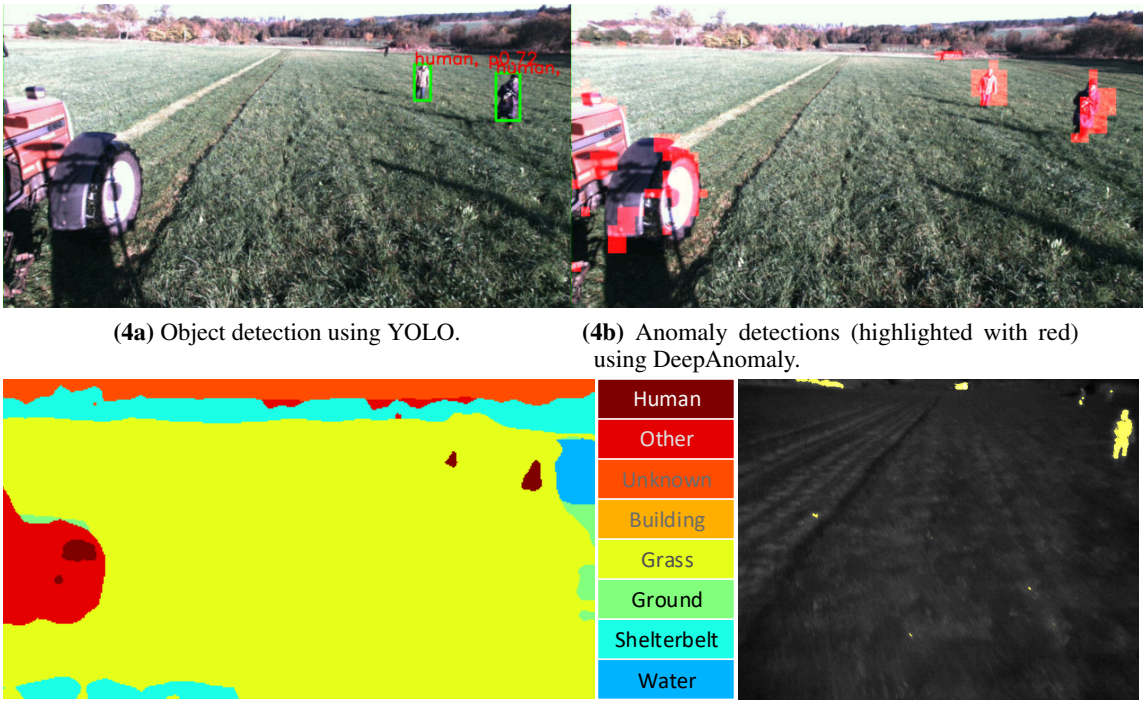
3.3.1 Cameras

In this section, multiple ISMs are described for the stereo camera and thermal camera. First, the individual detection algorithms operating on image data are explained. Then, two procedures for aligning detections to OGMs are proposed.

3.3.1.1 Detection Algorithms

A total of four detection algorithms for the stereo camera have been used; Locally Decorrelated Channel Features (LDCF) for pedestrian detection (Nam et al., 2014), an improved version of You Only Look Once (YOLO) (Redmon and Farhadi, 2016; Redmon et al., 2016) for object detection, a Fully Convolutional Neural Network (FCN) for semantic segmentation (Long et al., 2015), and DeepAnomaly (Christiansen et al., 2016a) for anomaly detection. The algorithms all use a single color image from the stereo camera. For the thermal camera, a heat detection algorithm (HeatDetection) is used to detect objects that are warm

compared to the background using a dynamically adjusted threshold (Christiansen et al., 2014). Figure 4 presents examples of output predictions from the detection algorithms.



(4a) Object detection using YOLO.

(4b) Anomaly detections (highlighted with red) using DeepAnomaly.

(4c) Semantic segmentation using FCN.

(4d) Thermal camera detections (highlighted with yellow) using HeatDetection.

Figure 4. Camera detections for stereo and thermal camera. Written and informed consent was obtained from all depicted individuals.

LDCF is a pedestrian detection algorithm delimiting instances by bounding boxes with fixed aspect ratios. The model is trained on the INRIA Person Dataset (Dalal and Triggs, 2005). The detector is publicly available in a MATLAB-based framework by Dollar (2015) and has been converted to C++ and wrapped in a ROS-package[4] (Kragh et al., 2016).

YOLO is a deep learning-based object detector delimiting instances by bounding boxes of variable aspect ratios. The detector is developed in the deep learning framework Darknet (Redmon, 2013) and trained on ImageNet (Berg and Deng, 2015) and Microsoft COCO (Lin et al., 2014) for detecting 80 object categories. For running the algorithm within the proposed framework, a ROS-package[5] has been developed which also applies a remapping of the 80 object classes into three classes (human, object, and unknown).

FCN uses the backbone of VGG (Simonyan and Zisserman, 2014) to make a fully convolutional semantic segmentation algorithm that classifies all pixels in an image. The model is developed in Caffe (Jia et al.,

[4]ROS package available at `https://github.com/PeteHeine/pedestrian_detector_ros.git`

[5]ROS package available at `https://github.com/PeteHeine/yolo_v2_ros`

2014) and is publicly available[6]. The model is trained on the 59 most frequent classes of the Pascal Context dataset (Mottaghi et al., 2014). Unlike the more popular Pascal VOC dataset (Everingham et al., 2013) with only 20 object classes, Pascal Context provides full image annotations of 407 classes. In Christiansen et al. (2016b), the 59 object classes are remapped to only 11 classes to investigate semantic segmentation in an agricultural context. In Kragh et al. (2016), the detector has been wrapped in a ROS-package[7]. In the current work, predictions are remapped to six classes (human, object, grass, ground, vegetation, and undefined).

DeepAnomaly is a deep learning-based detection algorithm for detecting anomalies (Christiansen et al., 2016a). The backbone is AlexNet (Krizhevsky et al., 2012) trained on ImageNet, and the anomaly detector is modeled using 150 images from the dataset in Christiansen et al. (2017). The output consists of coarse predictions of the whole image.

HeatDetection uses a heat detection principle from Christiansen et al. (2014) for detecting warm objects using a thermal camera. The median temperature is determined for all image pixels of the current image, and the dynamic threshold is defined $3.0\,^{\circ}\mathrm{C}$ above the median temperature. In this work, the median temperature is determined for the bottom $80\,\%$ of the image to not include image sections of the sky. Subtracting the image by the dynamic threshold and clipping values below zero results in a heat map of how much each pixel has exceeded the dynamic threshold. A ROS-package is publicly available[8].

3.3.1.2 Mapping of Detections to OGM

Camera detections are mapped to an OGM representation (Korthals et al., 2017b) using two procedures as presented in Figure 5. The top branch denoted *Bounding Boxes to OGMs* is for mapping detections represented by bounding boxes. The bottom branch denoted *Segmentations to OGMs* is for mapping segmented image detections. Finally, a few exceptions exist for DeepAnomaly and two FCN classes where segmented elements are converted to bounding box representations using a connected component module before mapping to OGM. The code has been made publicly available as ROS packages[9,10]. Below, the two branches are described in more detail.

Bounding Boxes to OGMs

This procedure maps detections to OGMs by first converting 2D bounding boxes to 3D cylinders. First, the distance to an object is estimated using depth from stereo matching. The distance is defined as the median depth inside the bounding box. The estimated distance is assigned to each bounding box corner and mapped to 3D using conventional camera geometry. Bounding box corners are converted to a cylinder represented by a center position, width, and height. Finally, 3D detections are mapped to an OGM as the output of the top branch in Figure 5.

Various heuristics are used for modeling the OGM's uncertainties. Areas outside the camera's field of view (FOV) are set to 0.5. Areas inside the FOV with no detections w.r.t. m are set to 0.4 indicating lower probabilities of occupancy. Detections w.r.t. m are given a value between 0.5 and 0.8 to indicate that the areas are occupied by the corresponding detections. A value of 0.5 represents the minimum prediction

[6]Model is available at `https://github.com/shelhamer/fcn.berkeleyvision.org`

[7]ROS package available at `https://github.com/PeteHeine/fcn8_ros`

[8]ROS package available at `https://github.com/PeteHeine/dynamic_heat_detection`

[9]ROS package available at `https://github.com/PeteHeine/image_inverse_sensor_model2`

[10]ROS package available at `https://github.com/PeteHeine/image_boundingbox_to_3d`

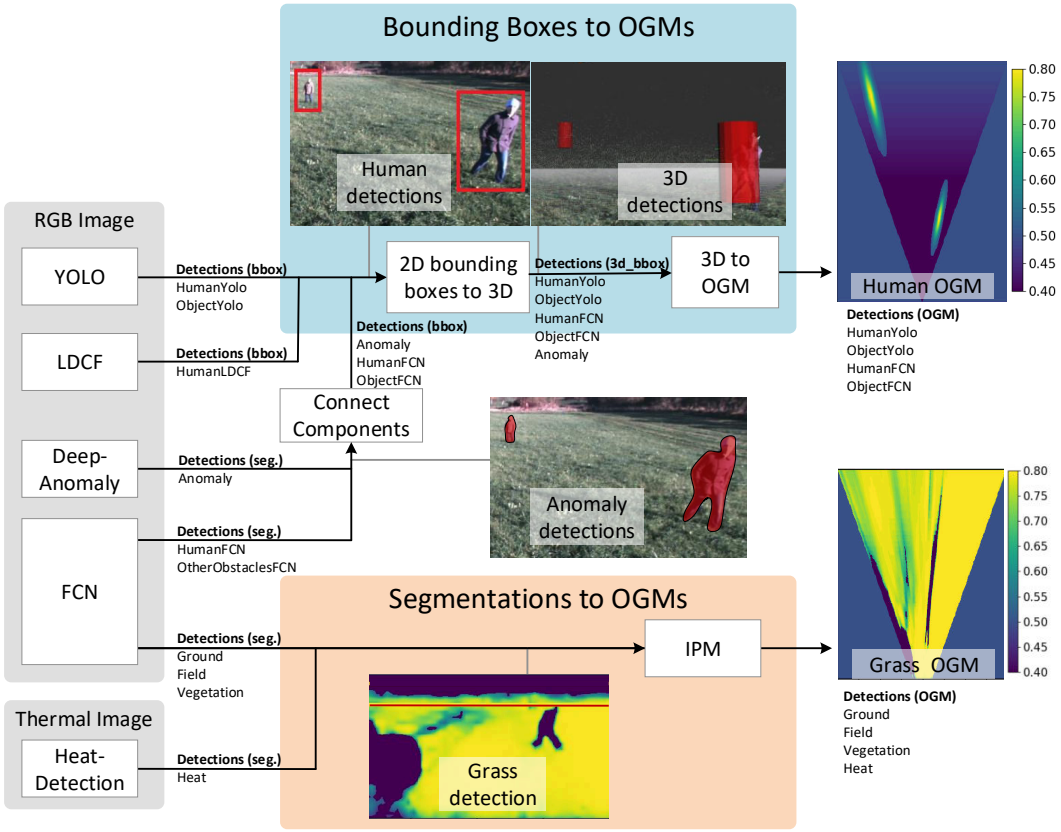


Figure 5. Converting detections to OGMs. Written and informed consent was obtained from all depicted individuals.

or class probability by a detection algorithm, whereas a value of 0.8 represents the maximum. Values in between are scaled linearly. A maximum value of 0.8 was chosen to avoid early saturation under fusion.

Imprecise localization of a detection is modeled by a Gaussian distribution. For a camera, the uncertainty of distance (radial coordinate) and angle (angular coordinate) to the object are independent. This is incorporated by modeling each polar coordinate (radial and angular) with independent uncertainties. In Figure 5, the localization uncertainty caused by the radial coordinate is larger than the uncertainty caused by the angular coordinate.

A detection algorithm is less likely to detect distant obstacles or to guarantee that an obstacle is not there. To model this, the certainty of not detecting an obstacle is reduced linearly by the distance from the nearest to the most distant grid cells. In Figure 5, the probability increases linearly with distance from 0.4 to 0.5.

Segmentations to OGMs

Inverse perspective mapping (IPM) is used for mapping image segmentations to a grid map. IPM projects an image from the camera frame to the ground plane surface using a geometrical transformation (Bertozzi and Broggi, 1998; Konrad et al., 2012). The purpose of IPM is to remove/inverse the perspective effect by changing the viewpoint from the camera to a bird's-eye view. Areas outside the camera FOV are set to 0.5.

Areas inside the FOV with no detections are set to 0.4. Detections are given a value between 0.5 and 0.8 to indicate that the areas are occupied.

The IPM algorithm is able to approximate the actual mapping for flat elements on the surface such as grass. However, elements protruding or positioned above the ground surface (e.g. humans and many obstacles) are imprecisely mapped. For this reason, segmentations of anomalies, humans, and other obstacles are converted to bounding boxes using a connected component module as illustrated in Figure 5. The OGM for a grass-segmented image is presented in the bottom of the figure.

3.3.2 Lidar

The inverse sensor model for the lidar sensor consists of a detection algorithm and a mapping to align detection information to a local 2D occupancy grid map (OGM) in the sensor frame. The detection algorithm operates directly on 3D point clouds with approximately $70\,000\,\mathrm{points/frame}$ generated at $10\,\mathrm{fps}$ by the Velodyne HDL-32E lidar. First, 13 features are calculated per point using neighborhood statistics that depend on local point densities (Kragh et al., 2015). Second, a Support Vector Machine (SVM) classifies each point as either *ground*, *vegetation*, or *object*. It further assigns probability estimates (Wu et al., 2004) to each class to describe the certainty of each classification. The SVM classifier was trained on the same data used in Kragh et al. (2015).

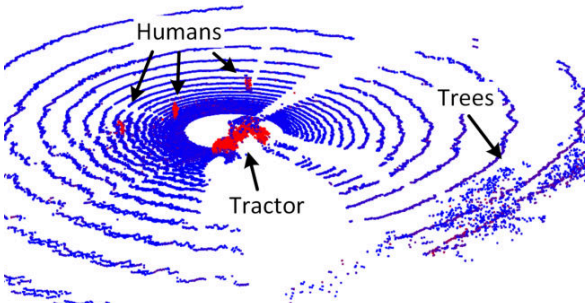
The mapping from detection probabilities to a local 2D grid is handled by projecting and resampling 3D points into 2D grid cells. For each 2D grid cell, class probabilities of all 3D points whose flattened projection lies inside are averaged and normalized such that the three class probabilities sum to 1. This results in three 2D probability grids: P^*_{object}, $P^*_{vegetation}$, and P^*_{ground}. The three classes are combined into two OGMs (lidar-SVM-object and lidar-SVM-vegetation) by incorporating the *ground* probabilities into the *object* and *vegetation* classes probabilistically with Bayesian fusion. For each grid cell m in an OGM, the log odds ratio of e.g. the *object* class is:

$$\begin{aligned}\mathrm{logOdd}\left(P_{object}\left(m\right)\right) &= \mathrm{logOdd}\left(P^*_{object}\left(m\right)\right) + \mathrm{logOdd}\left(1 - P^*_{ground}\left(m\right)\right) \\ &= \log\left(P^*_{object}\left(m\right)\right) - \log\left(1 - P^*_{object}\left(m\right)\right) \\ &\quad - \log\left(P^*_{ground}\left(m\right)\right) + \log\left(1 - P^*_{ground}\left(m\right)\right) .\end{aligned} \tag{8}$$

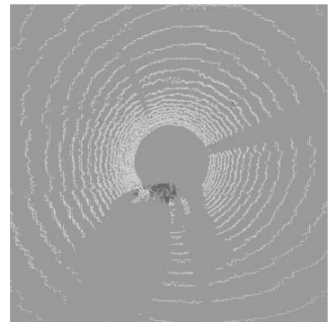
Figure 6a shows an example of a point cloud colored by *object* probabilities from the SVM classifier, while Figure 6b shows the corresponding *object* OGM.

3.3.3 Radar

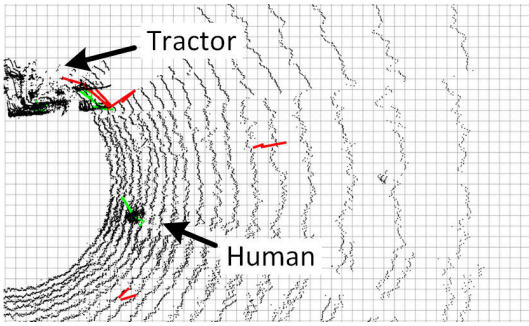
The Delphi ESR automotive radar provides a list of up to 32 targets for each frame. Each target is represented by an angle, a range, and an amplitude. Most targets, however, represent internal noise in the radar and have low amplitudes. Simply filtering out these targets with a threshold eliminates radar returns from low-reflective objects such as humans and animals. Therefore, instead the approach from the authors' previous paper (Kragh et al., 2016) was used in combination with a tracking algorithm between subsequent frames known as the Kuhn-Munkres assignment algorithm (Munkres, 1957). Only radar targets that are less than $2\,\mathrm{m}$ apart between two consecutive frames are associated. A track i is described by its current position and its track length L_i. It is confirmed when $L_i > L_{\min} = 3\,\mathrm{m}$ and converted to a detection



(6a) Point cloud with pseudo-colored probability estimates of the *object* class. Blue and red denote low and high probabilities, respectively.



(6b) Resulting lidar OGM for the *object* class illustrating low (bright) and high (dark) probabilities.



(6c) Radar detection example with confirmed (green) and unconfirmed (red) radar tracks overlaid on point cloud.

(6d) Resulting radar OGM.

Figure 6. Lidar and radar detections and OGMs.

pseudo-probability by:

$$P_{radar,i} = 0.5 + 0.5\frac{L_i - L_{\min}}{L_i}\,. \tag{9}$$

The addition of 0.5 makes the detector report only positive information of occupancy, thus not indicating absence of objects. The mapping from detection probabilities to a local 2D grid is handled by converting from polar to cartesian coordinates and resampling into 2D grid cells. For each 2D grid cell, class probabilities of all detections lying inside are averaged. This results in a 2D probability grid P^*_{radar}. Finally, the log odds ratio for each grid cell m in the radar OGM (radar-tracking) can be expressed as:

$$\text{logOdd}\left(P_{radar}\left(m\right)\right) = \log\left(P^*_{radar}\left(m\right)\right) - \log\left(1 - P^*_{radar}\left(m\right)\right)\,. \tag{10}$$

Figure 6c shows an example of confirmed (green) and unconfirmed (red) radar tracks overlaid on the corresponding point cloud, while Figure 6d shows the resulting radar OGM.

3.4 Process Evaluation

Farming scenarios are commonly well-defined and the trajectories are always planned in advance to yield optimal efficiency. However, the field may consist of many different properties that can only be revealed by

sensing the current environment. Common properties are *cropable*, *traversable*, or *non-traversable*, where of course the yield itself is of special interest.

The environment of the field is made up of structures in space that are sensed by diverse sensors. While the well-defined vehicle trajectory traverses this area, this path is of particular interest to forecast implement parameters or steering suggestions. Further, due to imperfections in sensor calibration, registration, and synchronization, areas of detections may not always overlap and will therefore always have spots where only certain sensors sense a property. This phenomenon evolves along the frustum and therefore along the planned trajectory. Thus, changes in real-world scenes are sequential in space, and the sequential nature can be used to learn property relationships between the various semantical occupancy grid map (SOGM) layers to analyze scenes. In this section, a hierarchical model that maps an observed SOGM along a trajectory to properties is presented.

Figure 7a shows the kind of structured information that is envisioned parsing from the trajectory over an SOGM. The lowest level corresponds to the feature vectors extracted from Equation 6. The middle layer corresponds to a property (e.g. *cropable*), and the top root node represents the trajectory. The cost of obtaining such hierarchical annotations would be very high due to the complexity of the annotation task. Typically, agricultural datasets are not labelled with all desired properties. As a result, models for learning such structures should also be able to operate in an unsupervised framework.

The problems to address are twofold. *Learning*: In order to categorize or classify mappings along the trajectory into properties, statistical characterizations of the patterns of observation sequences must be learned. *Classification*: Given observations along a trajectory, an algorithm is needed to classify these into properties.

3.4.1 A Generative Model for Inducing Properties over SOGMs

For the given task of path traversal, a hierarchical approach is targeted that not only models the single property at a certain location, but also the whole object itself. The probability making observation $\mathcal{O}$ with property w can be expressed as the joint probability

$$P\left(\mathcal{O}, w; \lambda\right) = \prod_{i=1}^{I} P\left(w_i\right) \prod_{j=1}^{J} P\left(\mathcal{O}_j | w_i; \lambda\right) \tag{11}$$

with the hidden variable w, $P(w)$ being the discrete property probability, and λ being the generative property model for the observed feature vector $\mathcal{O}$. The amount of properties along a path are enumerated by I while the length of a single property is denoted by J.

The inter-property model $\lambda_w = (S, \mathcal{O}, A, \Phi, \Pi)$ is a corresponding Hidden Markov Model (HMM) with states s, observations $\mathcal{O}$, transition probability A, emission probability Φ, and start probability Π for every single property w. The emission probability is modeled as a beta mixture model (BMM) over the N semantical occupancy grids with δ as normalization weight and the beta function $\mathcal{B}$ with its parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$:

$$\Phi\left(\boldsymbol{\delta}, \boldsymbol{\alpha}, \boldsymbol{\beta}\right) = \sum_{n=1}^{N} \delta_n \mathcal{B}\left(\alpha_n, \beta_n\right) \ . \tag{12}$$

At the lowest level of the hierarchical structure specified by the model in Figure 7a is a sequence of probabilistic feature vectors. In reality, there are infinitely many feature vectors. Moreover, due to

imperfections in localization and mapping, regions among semantical layers may not overlap perfectly and can be noisy as depicted in Figure 7a.

As discussed before, it is expected that trajectories are composed of a sequence of semantically meaningful properties that manifest themselves in various property-compositions. The feature vectors themselves can be directly modelled as a BMM as stated in Equation 12. While a direct classification might be suitable, a sequence along the trajectory (which is along the field of view) may represent a true underlying property even better and can only be revealed when taking spatially earlier readings into account. Therefore, a generative model is introduced in Figure 7b, where the interpretable properties generate probability feature vectors (the features of a supercell).

The distribution of properties in the field will be stochastic in nature (e.g. a trajectory may contain segments of crop, weed, non-traversability), and the distribution of the feature vectors themselves is beta-distributed and property-dependent. While the number of such properties is expected to be very large, it is assumed that for a given dataset a limited number of properties can describe the property space fairly well.

The generative model is shown in Figure 7b. K properties in the vocabulary and feature vectors $\mathbf{P} \in \mathbb{R}^N$ are as observations are assumed (that is $\mathbf{P}_c$ from Equation 6). A set of T trajectories can be generated as follows: for each trajectory t, I properties are drawn from a unigram distribution U. We then draw J feature-vectors from the specific generative property-model. Thus, in this model, each trajectory is a bag of properties and each occurrence of a property is a sequence of feature-vectors. The resulting hirachical model is shown as a concatenation of 7d, as an ergodic model for the inter-properties, and 7c, as left-right model for the inter-property realization.

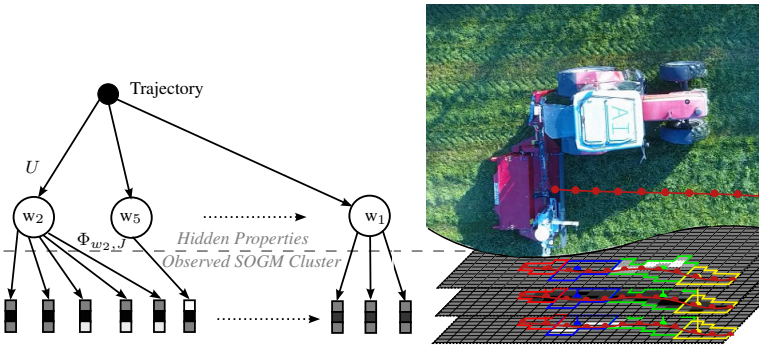
3.4.2 Model Estimation and Decoding

An HMM, as shown in Figure 7c, for each of the I properties is produced. It is modeled as a left-right structure comprising L states with an additional exit transition for each state to follow the aforementioned idea of non-perfectly overlapping detections. Thus, property burn-in, settling, and burn-out behaviours can be modeled in the beginning, middle, and end of the trajectory. Therefore, a minimum of three states s are necessary to model these behaviours for every property w. Since properties may have very diverse features in the start and end sequence, all states have their own emission probability.

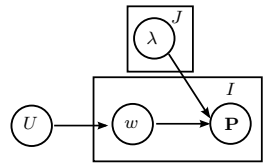
The HMMs for the properties are now put together as shown in Figure 7d. For the sake of simplicity, a black circle represents the hub for all property transitions in the ergodic model. $P(w_k)$ represents the probability of the property w_k. This approach is trained in a supervised fashion and thus, the objective function for one property w tends to find the most likely model λ_w^*, given an observation sequence $\mathcal{O} = (\mathbf{P}_1, \ldots, \mathbf{P}_J)$ and its corresponding ground truth (GT) sequence $\mathcal{S} = (s_1, \ldots, s_J)$:

$$\lambda_w^* = \text{argmax}_{\lambda_w} P(\mathcal{O}, \mathcal{S}|\lambda_w). \tag{13}$$

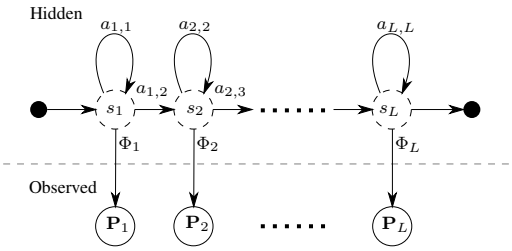
Equation 13 can be estimated by *instance counting*, which counts the hidden state transitions and output states, and uses the relative frequencies as estimates for the transition probabilities of λ_w. The inter-property model can be trained in the same way. Given the GT, the parameters α and β can be directly determined by the *Method of Moments*. For decoding, the likelihood $P(\mathcal{O}|\lambda_w)$ that a given model λ_w has produced a given observation sequence $\mathcal{O}$ is calculated by the Viterbi algorithm (Rabiner, 1989).



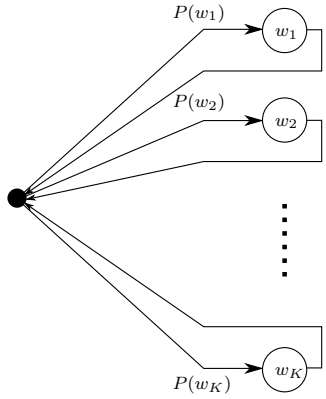
(7a) Conceptual representation of the proposed framework with the generative sampling on the left and a corresponding scenario with observations along the red tractor trajectory on the right.



(7b) Generative model for property identification. Only $\mathbf{P}$ is observed.



(7c) Left-right structure of one intra-property model λ_w for the inter-property model (exit transition per state is not visualized).



(7d) Ergodic inter-property model of the HMM.

Figure 7. Generative model and Hidden Markov Model framework for identifying properties in the mapped data.

4 EVALUATION

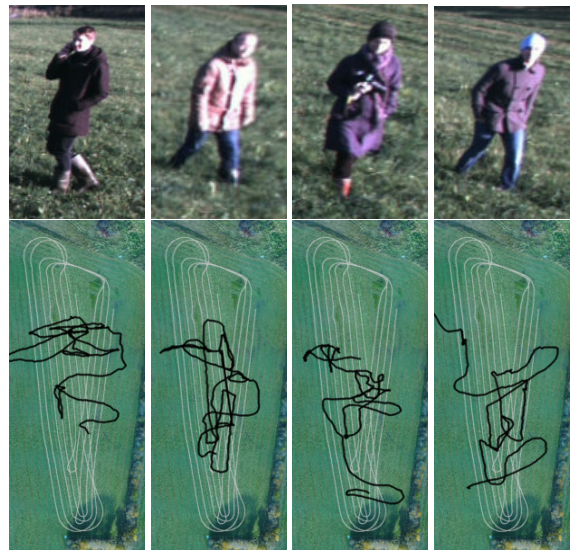
In this section, we evaluate the proposed architecture for obstacle detection, recognition, and mapping on static and dynamic obstacles, individually. Further, we evaluate the process evaluation on the mapped data with a spatial resolution of $10\,\mathrm{cm}$ per cell.

4.1 Dataset

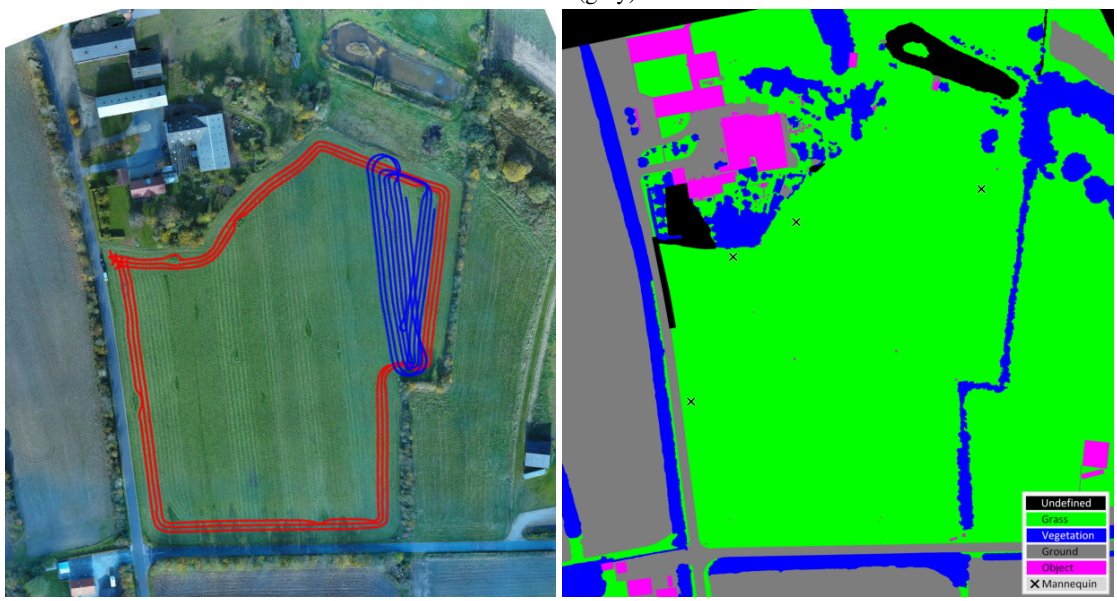
The publicly available FieldSAFE dataset (Kragh et al., 2017) for multi-modal obstacle detection in agricultural fields was used for the evaluation. The dataset includes two hours of recording during mowing of a grass field in Denmark. Figure 8a illustrates examples of static obstacles in the dataset, whereas Figure 8b shows examples of dynamic obstacles (humans) and their GT traversed paths overlaid on the path of the tractor. Figure 8c shows a static orthophoto of the field together with pixel-wise manually labeled

GT classes. In the following section, the annotated orthophoto is used as ground truth for evaluating the proposed architecture.

(8a) Examples of static obstacles.



(8b) Examples of moving obstacles (from the stereo camera) and their paths (black) overlaid on tractor path (grey).



(8c) Colored and labeled orthophotos. Left: orthophoto with tractor tracks overlaid. The red track includes only static obstacles, whereas the blue track also has moving obstacles. Right: annotated orthophoto with pixel-wise labels.

Figure 8. FieldSAFE dataset. Adapted from Kragh et al. (2017) with permission. Written and informed consent was obtained from all depicted individuals.

4.2 Static Scenario

Two different evaluations have been performed: evaluation **A** for detecting process-relevant classes exclusively, and evaluation **B** for detecting occupied areas with respect to traversability.

For evaluation **A**, GT labels were grouped into four different process-relevant classes (*Vulnerable obstacles*, *Processable*, *Traversable*, and *Non-traversable*). The *Vulnerable obstacles* class included GT label *Mannequin* and covered regions with which a collision must be avoided under any circumstance. The *Processable* class included GT label *Grass* and represented the crop. The *Traversable* class included GT labels *Grass* and *Ground* and represented areas that could be traversed by the vehicle. Finally, the *Non-traversable* class included GT label *Vegetation* and represented areas that must be avoided to not damage the vehicle. For evaluating the process-relevant detection, each of the four classes was considered in its own property map. Included GT classes were marked as *occupied*, whereas all other classes were treated as *unknown*.

For evaluation **B**, GT labels were grouped into three different properties (*occupied*, *unoccupied*, and *unknown*) according to their traversability. The labels *Vegetation*, *Mannequin*, and *Object* were combined to the *occupied* property. The label *Undefined* was considered an *unknown* property, whereas the remaining classes *Ground* and *Grass* were combined to the *unoccupied* property.

To quantify the detection of static obstacles and to compare it against the GT data from subsection 4.1, the evaluation pipeline from Figure 9a was applied. The mapserver's maps, which contain all fused classifier information, were stored as explained in Korthals et al. (2017a). The single maps were stitched together, such that they meet the size and resolution of the GT data. Afterwards, different combinations of the maps were applied as represented in Table 2 to achieve the corresponding results in the evaluation step.

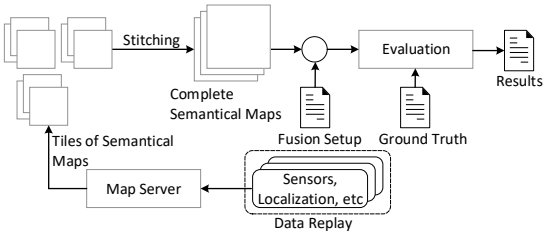
It is worth noticing that the mapping technique is very prone to misclassification, which can be caused for example by sun blinded cameras or systematic errors. To address the second case, a blind spot has been applied at the location of the tractor so that the mapping of self-classification, heavily caused by the radar, was overcome. This approach has been applied to all the following evaluations as well.

The resulting tri-state maps from GT data and mapping were compared tile-wise against each other, such that the true positives (TP), false positives (FP), and false negatives (FN) could be calculated for the entire map.

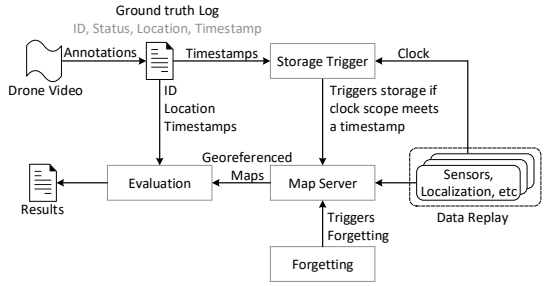
To do so, the binary mapping $G : m \to \{0, 1\}$ is defined which converts the cell m to an indicator. Further, G_{GT} refers to the map constructed from the GT data, and G_{M} that maps the cell m, given the estimated posterior $P(m)$ evaluated on the subset of seen cells $M' = \{m \in M | P(m) < 0.5 - \epsilon \vee P(m) > 0.5 + \epsilon\}$. Thus, M' refers to all observed cells which properties are known. To overcome floating-point quantization noise, a slack variable with $\epsilon = .01$ was introduced to the evaluation:

$$G_{\mathrm{M}}(m) = \begin{cases} 1, & \text{if } P(m|z_{1:T}, x_{1:T}) > 0.5 \\ 0, & \text{otherwise} \end{cases}, \quad G_{\mathrm{GT}}(m) = \begin{cases} 1, & \text{if } m \textit{ occupied} \\ 0, & \text{if } m \textit{ unoccupied} \end{cases}. \tag{14}$$

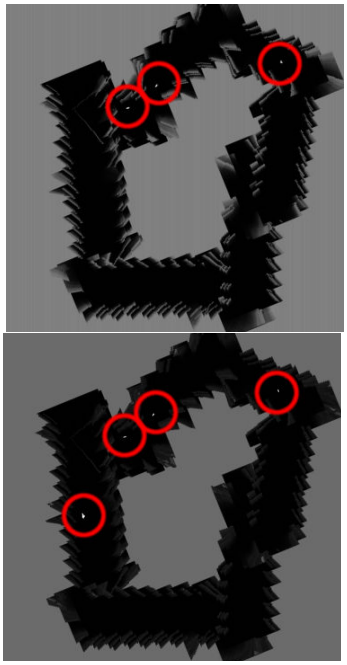
The function G_{M} only takes the estimated map, and G_{GT} only takes the GT map into account. TP, FP, and FN can then be calculated by cell-wise multiplication between the estimated map G_{M} and the GT map G_{GT}:



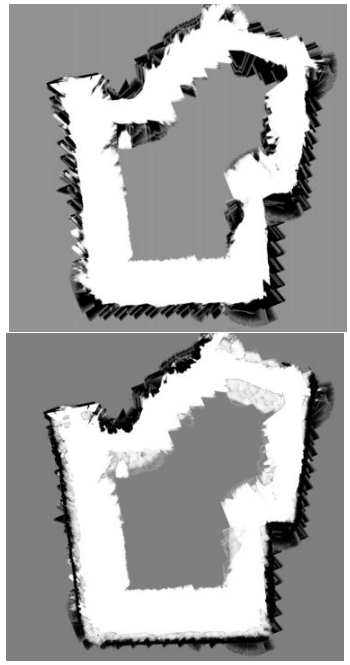
(9a) Evaluation pipeline from static recording to evaluation with stitching.



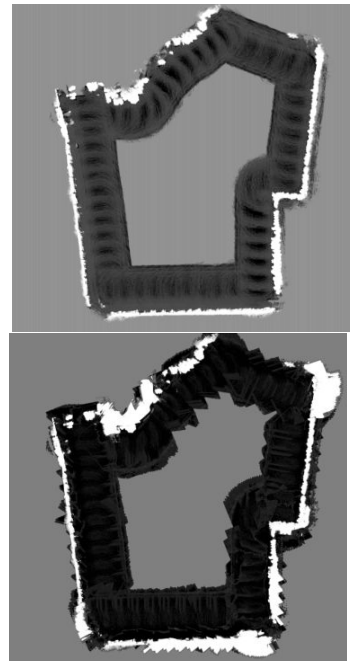
(9b) Evaluation pipeline from dynamic recording using drone video and recorded data as input.



(9c) cam-YOLO-human (top) and fused human class (bot.).



(9d) cam-FCN-ground (top) and fused ground class (bot.).



(9e) lidar-SVM-veg. (top) and fused vegetation class (bot.).

(9f) radar-tracking (left), Bayesian fusion among class (mid.), and complete fused map (right).

Figure 9. Examples for different stitched mapping results for different evaluations of Table 2a (9c-9e), 2b (9f), and evaluation pipelines (9a, 9b). Red circles emphasize correct object/mannequin detections. Grayscale encoding: black $\widehat{=}$ *occupied*, white $\widehat{=}$ *unoccupied*, gray $\widehat{=}$ *unknown*.

$$\mathrm{TP} = \sum_{m \in M'} G_{\mathrm{GT}}(m)\, G_{\mathrm{M}}(m)\ , \mathrm{FP} = \sum_{m \in M'} (1 - G_{\mathrm{GT}}(m))\, G_{\mathrm{M}}(m)\ , \mathrm{FN} = \sum_{m \in M'} G_{\mathrm{GT}}(m)\, (1 - G_{\mathrm{M}}(m))\ . \tag{15}$$

The Precision, Recall, F_1 score, and entropy H were calculated as follows:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{FP} + \mathrm{TP}}\,,\ \mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{FN} + \mathrm{TP}}\,,\ \mathrm{F}_1 = 2\frac{\mathrm{Recall} \cdot \mathrm{Precision}}{\mathrm{Recall} + \mathrm{Precision}}\,, \tag{16}$$

$$H(P(M)) = -\sum_{m \in M} P(m) \log P(m) + (1 - P(m)) \log (1 - P(m))\ . \tag{17}$$

Table 2a shows the results of evaluation **A**, i.e. detecting process-relevant classes exclusively. The results are grouped by the process-relevant classes, and the three columns show individual algorithm detection results, fusion across algorithms, and fusion across sensors, respectively. Here, both competitive (Bayesian) fusion and complementary (max-pooling) fusion were applied for the two fusion scenarios.

Table 2b shows the results of evaluation **B**, i.e. detecting occupied areas with respect to traversability. The first column shows individual detection results for each of the algorithms. These are grouped by object categories such that different algorithms from different sensors that detect similar classes are grouped together. In the second column, algorithms from each group of categories are fused with competitive (Bayesian) fusion. For classifiers detecting the same object classes, competitive fusion increases the precision while maintaining information gain (entropy). In the third column, detections from all sensors (and algorithms) are fused with complementary (max-pooling) fusion. For classifiers detecting different object classes, complementary fusion increases recall while maintaining precision. In practice, this results in a more complete detection of the environment.

Figures 9c – 9e show an excerpt from the corresponding evaluation in Table 2a. The constructed maps were built from traversing the depicted red track in Figure 8c. The gray area represents unknown or not-seen areas, white denotes a vote for, and black against the desired class. Figure 9c shows the single cam-YOLO-human classification in the top image, whereas the bottom image consists of the combination of all camera-based human classifications. While the single classifier already showed plausible results with correct human classifications highlighted with red circles, it still missed some detections. The combination of classifiers overcame this issue and also increased certainty for classifications where no humans resided. Figure 9d shows the ground and crop classifications of cam-FCN-ground in the top image and the corresponding combination in the bottom. While the camera-based classification showed significant noise at the borders, the classifiers supplemented each other to achieve a denoised and extended classification of the ground. Figure 9e shows the lidar-SVM-vegetation classification in the top image and a combination with camera-based classifiers at the bottom. The lidar already achieved results that were qualitatively close to the GT data. While in the fused result artifacts resulting from the ISM approach are visible at the outer borders, the overall score increased due to the gain of new information and increased certainty of already perceived information. Figure 9f shows an excerpt from the evaluation in Table 2b where a classical retrieval of an occupancy grid map was aimed. The radar classification depicted on the left provided a quite clean obstacle detection which in combination with the remaining object classifiers in

Table 2. Evaluation of static obstacle detection and mapping. The vertical lines encapsulate groups of algorithms on the left side and present their fused results on the right hand side. Values in percentages.

(2a) Evaluation **A**. Process-relevant object detection for single classifiers, classifier combinations, and sensor combinations.

	Single classifiers				Fusion among class.					Fusion among sensors				
Classifier	F_1	Prec.	Rec.	H	Fus.	F_1	Prec.	Rec.	H	Fus.	F_1	Prec.	Rec.	H
Vulnerable Obstacles (Mannequin)														
cam-LDCF-human	1.3	0.7	25.9	83.2	max.	3.2	1.6	73.4	86.2					
cam-FCN-human	3.4	1.7	73.6	75.6	bay.	12.6	7.1	57.4	84.3					
cam-YOLO-human	11.7	6.9	36.1	75.5										
Processable (Grass)														
cam-FCN-grass	85.2	94.2	77.8	75.2										
Traversable (Grass & Road & Ground)														
cam-FCN-grass	83.4	96.3	73.6	75.2	max.	84.6	96.0	75.6	75.3	max.	90.1	89.2	91.0	92.3
cam-FCN-ground	24.0	96.8	13.7	75.1	bay.	82.0	97.2	71.0	75.2	bay.	87.7	90.8	84.8	92.2
lidar-SVM-ground	89.7	89.4	90.1	81.1										
Non-Traversable (Vegetation)														
lidar-SVM-veg.	83.6	81.4	86.0	87.9						max.	84.3	80.1	89.1	92.3
cam-FCN-veg.	46.6	32.2	84.7	81.2						bay.	84.8	81.3	88.7	92.3

(2b) Evaluation **B**. Traversability assessment of static obstacles for single classifiers, classifier combinations, and sensor combinations.

	Single classifiers				Bayesian among class.				Max-pooling among class.			
Classifier	F_1	Prec.	Rec.	H	F_1	Prec.	Rec.	H	F_1	Prec.	Rec.	H
cam-FCN-human	3.8	25.3	2.1	75.6								
cam-LDCF-human	0.7	3.7	0.4	83.2								
cam-YOLO-human	1.2	6.8	0.7	75.5								
radar-tracking	2.6	3.5	2.1	15.9	13.0	67.4	7.2	89.2				
thermal-HeatDetection	7.3	16.6	4.7	88.6								
lidar-SVM-object	7.8	66.8	4.1	89.7								
cam-FCN-object	4.1	30.8	2.2	76.3					88.8	88.3	89.4	92.5
cam-YOLO-object	2.0	3.9	1.3	75.6								
cam-DeepAnomaly	2.0	3.8	1.4	75.6	22.3	72.3	13.2	89.5				
radar-tracking	2.6	3.5	2.1	15.9								
lidar-SVM-object	7.8	66.8	4.1	89.7								
lidar-SVM-veg.	83.5	81.4	85.8	87.9	84.6	88.3	81.6	92.3				
cam-FCN-veg.	46.7	32.2	84.4	81.2								

the middle led to a richer and more precise result. Finally, the fusion of all classifiers and sensors on the right resulted in a quite complete occupancy map.

4.3 Dynamic Scenario

To evaluate the detection of dynamic obstacles, the mapserver was applied in exactly the same way as for the static scenario. However, instead of evaluating a stitched map combining information from traversing the entire field, the mapserver was queried temporally for each available timestamp t in the GT data. In order to evaluate only the detection of dynamic and non-static obstacles, recency weighting as introduced in 3.2.4 was applied. The *ForgetValue* and *ForgetRate* are evaluated exhaustively at the end of this section. However, for the following evaluations, a high *ForgetRate* of 6 and a high *ForgetValue* of 0.8 were used,

as these values ensured a responsive mapping where only recent measurements were taken into account. In this way, the mapserver continuously updated the positions of moving objects, while still allowing an appropriate amount of information fusion of non-synchronized sensors.

Contrary to the static evaluation where GT annotations were dense and pixel-wise, the GT annotations of dynamic obstacles were point-based (Kragh et al., 2017). Therefore, tile-wise comparison between GT data and the fused map was unfeasible. Instead, point-wise GT annotations were compared to clusters of detections for each timestamp. Figure 10a illustrates the dynamic evaluation scenario. First, the different mapserver layers were fused. The resulting tri-state (occupied, unoccupied, unknown) likelihood map was then clustered for each state with 8-connected clustering. Clusters smaller than *MinClusterSize* were pruned to suppress noise. Finally, TP, FP, and FN were accumulated over time in the GT data by comparing the detected clusters c_j with index j and the GT positions p_i with index i:

$$\begin{aligned} \mathrm{TP} &= \sum_t \mathrm{TP}_t, \quad \mathrm{TP}_t = |\{p_i | \exists c_j : p_i \in c_j\}| \ , \\ \mathrm{FP} &= \sum_t \mathrm{FP}_t, \quad \mathrm{FP}_t = |\{c_j | p_i \notin c_j\}| \ , \\ \mathrm{FN} &= \sum_t \mathrm{FN}_t, \quad \mathrm{FN}_t = |\{p_i | p_i \notin c_j\}| \ . \end{aligned} \tag{18}$$

Regions that remained unknown ($P(m) = 0.5$) did not affect the evaluation, and only detected clusters and GT positions inside the sensor frustum were taken into account. Similar to the static scenario, precision, recall, and F_1-score metrics were calculated using Equation 17. Figure 10b shows an example from the dynamic evaluation. The GT positions are denoted by colored circles, while the detected clusters are represented by white regions beneath. In the depicted example, one true-positive, one false-positive, and one false-negative was counted due to the fact that the yellow and red positions were inside the sensors' frustum.

Figure 9b illustrates the evaluation pipeline for the temporal sequences. In an offline-procedure, all necessary GT information like person identifiers (ID), their status (visible/non-visible and standing/sitting/lying), the geo-referenced locations, and the timestamps was extracted. Afterwards, the mapserver ran in a common setup with the forgetting feature, where for every given GT timestamp the current maps of the mapserver were extracted. In an evaluation step, the maps were clustered and compared to the GT information to achieve the presented results in Figure 10c and 10d.

Table 3. Listing of setups and the detection algorithms they comprise.

Class	object	heat	object	objects/human	human	human	anomaly
Algorithm	detection	DynamicHeat	SVM	FCN	LDCF	YOLO	DeepAnomaly
Setup	9 (Radar)	3 (IR)	4 (Lidar)	5	6	7	8
	1			2 (Camera)			

Table 3 lists 9 different sensor/algorithm setups that were evaluated. Setup 1 includes all sensors and algorithms, setup 2 includes all stereo camera algorithms, whereas setup 3 – 9 concern individual sensors and detection algorithms.

Figure 10c shows precision, recall, and F_1-scores for setup 3 – 9, when varying the *MinClusterSize* used in the clustering. Figure 10c (right) shows results for clustering without subsequent dilation, whereas Figure 10c (left) introduces dilation by the vehicle radius of all clusters as is common in robotic navigation and planning algorithms (Dudek and Jenkin, 2010). In the current evaluation, dilation effectively mitigated the influence of localization inaccuracies and resulted in better scores. Objects that were detected and mapped with slight displacements from their GT positions were thus more likely to be included by dilated clusters. This indicated that a large part of false negative detections were located close to GT positions. Setup 4 (lidar) and 9 (radar) had undefined F_1-scores for *MinClusterSize*s above $0.7\,\mathrm{m}$ and $0.6\,\mathrm{m}$, respectively. This was caused by the two sensors providing precise 3D measurements, which made their detections precisely located and narrow in space. Since the human objects had small footprints, no clusters with areas above these values were generated. For the same reason, a *MinClusterSize* of $0.5\,\mathrm{m}$ was chosen as a compromise, such that most of the noisy sensor readings were filtered out, while small and correct detection footprints from humans were still kept.

Table 4 shows precision, recall, and F_1-scores for the fusion setups 1 and 2 using *MinClusterSize* $= 0.5$ and no subsequent cluster dilation. For setup 1 (all sensors and algorithms), complementary (max-pooling) fusion performed much better than competitive fusion. This was caused by the fact that detections from different sensors did not overlap perfectly due to localization errors. Competitive fusion therefore falsely combined non-overlapping detections, whereas the complementary fusion tolerated the localization issues by effectively summing all detection contributions. For setup 2 (camera-based detection), however, competitive (Bayesian) fusion was superior to complementary fusion. This was caused by the fact that the same camera was used by all algorithms, thereby mitigating localization errors and ensuring overlapping detections.

Table 4. Sensor fusion of setup 1 and 2 with different fusion strategies.

Setup	Fusion	F_1 (%)	Precision (%)	Recall (%)
1	Max	70.81	57.23	92.86
	Bayes	42.58	39.76	45.83
2	Max	57.32	51.14	65.22
	Bayes	61.22	56.96	66.18

Figure 10d shows precision, recall, and F_1 scores for setup 1 (all sensors), when varying the *ForgetRate* $(1 - 6)$ and *ForgetValue* $(0.1 - 0.9)$ of the mapserver. Similar to the above cases, *MinClusterSize* $= 0.5$ and no subsequent cluster dilation was applied. Clearly, all scores were dramatically influenced by the two parameters. A *ForgetValue* of 0.8 and *ForgetRate* of 6 seemed to be the best compromise between memory and responsiveness, such that only the most recent measurements were taken into account. A too large *ForgetValue* (close to 1) resulted in no memory, meaning that valuable information from previous frames was not taken into account. Contrarily, a too small *ForgetValue* (close to 0) resulted in too long memory (approaching the static scenario), effectively letting outdated information of obstacle positions stay in the map. Similarly, a too small *ForgetRate* resulted in too long memory, whereas the performance seemed to approach an upper limit with larger *ForgetRate*s.

4.4 Process Evaluation

The above-mentioned approaches were able to classify single observations point-wise and did not take into account surrounding classifications when determining classes of current observations. In agricultural processes, however, observations obtained from the surroundings are typically identical and homogeneous in particular. Furthermore, certain transitions between classes are rather unlikely. For example, if the classification of the current pose is *Grass*, it is rather unlikely that the classification of the next pose is *Ground*. From the processable class *Grass* to the traversable class *Road*, there are commonly ground, borders, or trenches. To utilize these dependencies between the individual classes, we use a Hidden Markov Model (HMM) and calculate the belief $P(\mathcal{O}, w; \lambda)$ about the class model λ from Equation 11. The observation per pose is the feature vector from Equation 6 of homogeneous clusters extracted via SEVDS (see Equation 5). The sequence of observations along some trajectory contains the upcoming poses of the vehicle as depicted in Figure 7a. A consequence of having metric grid maps is that via the shape constraints in Equation 5, implicit sizes of clusters can be given. This influences the step size of every pose to decode along the trajectory, so that empirically shape parameters for given step sizes can be found.

To compare the capabilities of the HMM against the static scenario from subsection 4.2, the same process-relevant classes (denoted in brackets) were chosen to train four different models ($K := 4$ w.r.t. Equation 11): Vulnerable Obstacles (Mannequin), Processable (Grass), Traversable (Ground), and Non-Traversable (Vegetation). It is worth mentioning that the class *Grass* was removed from the model *Traversable* to make it mutually exclusive against the model *Processable*.

The entire training was performed in a supervised fashion on the mapped data from the static scenario. All inter-property models as depicted in 7c had five hidden states ($L := 5$ w.r.t. Equation 11) due to the fact, that less states result in worse performance and more states do not show any improvements. The minimum amount of states can be explained by the necessary modeling of the burn-in and out behaviours as stated in 3.4.2, while more states do not improve the performance as the models tends to exit after the fifth state. Further, the training set was extracted out of randomly generated trajectories, while the test set represented trajectories driven by the vehicle. It was desired to forecast the class along the trajectory for as long as possible, but the maximum length was constrained by two factors: First, the applied mapserver only had a locally bounded area, where the maximum allowed range reading was equal to the size of the outer boundary minus the inner boundary (Kragh et al., 2016). For the presented experiments, the boundaries were set to $35\,\mathrm{m}$ and $10\,\mathrm{m}$ respectively, which resulted in a maximum forecast of $25\,\mathrm{m}$. Second, not all sensors exploited this maximum range reading and further, closer areas tended to be more precise in information due to the nature of the occupancy grid mapping algorithm. Thus, to have a fair comparison, the decoding was done for a close range starting from the tractor at $0\,\mathrm{m}$ to $12.5\,\mathrm{m}$ (Figure 11a) and a far range extending the former range from $12.5\,\mathrm{m}$ to $25\,\mathrm{m}$ (Figure 11b).

For training, the HMM was initialized as follows: All start and property probabilities were uniformly distributed with an additive Gaussian noise. The transmission probabilities of the property models were randomly initialized. The beta distribution mean for emission was set by k-means++ (Arthur and Vassilvitskii, 2007), while the variance was kept constant. Training and decoding was performed on all available detection algorithms as presented in Table 2b.

5 DISCUSSION

The proposed architecture is an extension of the authors previous paper on occupancy grid mapping in agriculture (Kragh et al., 2016). The current study has unified the system architecture and extended the

previous approach by a class-specific evaluation of static obstacles plus a method for detecting and mapping dynamic obstacles over time. Further, this paper has introduced a process evaluation method combining mapped environment detections over time into agriculturally relevant properties.

The provided evaluation measured the end-to-end ability of both fused and individual algorithms to detect and map elements with the provided architecture. That is, detections were not evaluated in local sensor frames, but were instead evaluated after projection to local 2D grids and after global mapping. A deficiency of such an evaluation was that it did not clarify why a given algorithm or sensor performed badly. The end-to-end detection error may have originated from multiple sources such as sensor noise, detection or local localization errors by algorithms, errors in intrinsic and extrinsic calibration parameters, inaccurate grid map representations, robot localization errors, and errors in the ground truth annotations. To isolate and quantify these error sources, GT data would be necessary for each link in the chain. However, annotations of obstacles were only available as global GPS-coordinates and not in the local vehicle frame or sensor frames (e.g. pixel-wise or bounding box annotation in camera images).

After fusing all sensors, the complete architecture reached an F_1-score of $88.8\,\%$ in static traversability assessment (Table 2b) and $70.8\,\%$ in dynamic obstacle detection (Table 4). The presented performance measures are useful for showing relative improvement with fusion and for comparing proposed methods. The metrics, however, can not quantify the safety-level of the system in real operation. A very low F_1-score for e.g. camera-based human detectors in Table 2a and Table 2b suggests that the combined localization and detection is of insufficient performance. However, an actual safety system should not be evaluated on F_1-scores of a map, but instead on e.g. the decoded process-relevant properties along the traversed trajectory as in Figure 11. As of today, no self-driving cars are certified for full autonomy, and, to the authors knowledge, no regulations describe exactly what detection accuracy, precision, frequency etc. would be required for certification. Instead, self-driving car manufacturers document their traveled distances during testing without incidents and without human intervention. An actual certification might end up building on measures like these. And most likely, autonomous vehicles in agriculture will follow and possibly extend the regulations of self-driving cars, once available.

As shown in Table 2a and Table 2b, classification performance generally increased as more sensors were introduced. However, different sensors detecting the same class may not always lead to a significant increase in accuracy. In fact, this was the case for the radar. The fusion of all sensors in Table 2b gave an F_1-score of $88.873\,\%$. The same setup without the radar gave an F_1-score of $88.871\,\%$, which was hardly a significant improvement. The specific radar and detection algorithm pair could thus be left out of the fusion setup, as it did not contribute with more information. On the other hand, even with insignificant improvements, additional sensors may still provide a more robust and redundant setup, thus mitigating single points of failure. And with another radar, specifically targeting agricultural scenarios (e.g. by penetrating vegetation), actual improvements in accuracy may be possible.

The results in Figure 10c (right) showed that the F_1-score could be improved significantly by introducing a cluster dilation corresponding to the vehicle size in the dynamic evaluation. Effectively, the dilation mitigated the influence of localization and demonstrated the potential of the detectors when being less sensitive to localization errors. An optimized localization, a model-based approach, or temporal tracking of detected clusters would therefore potentially increase the combined detection and localization results.

As previously mentioned, localization errors could also originate from inaccurate grid map representations in the ISMs. This could be caused by extrinsic and intrinsic calibration errors for each sensor, such that detections in the local sensor-frames were incorrectly transformed to the vehicle-frame.

Multiple heuristic models were introduced in the ISM to convert detections into occupancy probability estimates. Heuristic model parameters have been selected to model both detection and localization uncertainties for a given algorithm. In future work, these issues could be addressed by supervised training of a function approximator for mapping detections from local sensor-frames to the vehicle-frame as well as converting detection certainties to occupancy probabilities. Effectively, this could limit the number of heuristics and improve both localization and detection accuracy. One example is the heuristic model used for converting 2D bounding boxes to an OGM using a stereo camera as explained in paragraph 3.3.1.2. The uncertainty for localizing an object is modelled using assumed radial and the angular variances. However, the true radial and angular variances can be estimated more accurately from sensor calibrations. A more extensive approach would be to train the ISMs end-to-end, such that environment detections were directly output in local vehicle coordinates. However, this would contradict our applicable architecture approach that allows easy setup of different sensor combinations and would require a much larger dataset for training.

The semantical occupancy mapping technique used competitive (Bayesian) fusion for similar modalities followed by complementary (max-pooling) fusion for dissimilar modalities. This was both an intuitive and a reasonable procedure for fusing information and was demonstrated to increase the F_1-score. More advanced procedures such as instance boosting could be trained to learn the optimal combination of semantical maps. Such procedures would expectedly be less prone to misclassifications such as cameras blinded by the sun or potential systematic errors. Comprising the three possible levels of fusion, which are fusion on raw data, feature level, or decision level, our approach focuses on decision level fusion. Other approaches like Kalman filtering techniques tend to work on raw data and at feature level which might result in better fusion results, but also demand more effort in designing the filters themselves. Furthermore, varying setups cannot be considered easily due to necessary redesign of the filter. On the other hand, model-free approaches like particle filters have proven their capabilities also for occupancy grid map approaches by Korthals et al. (2016), but would need deeper insights into the sensors' design to build up proper fusion. As stated before, this approach pursues the easy changeability and extendability of sensors and other information sources. Considering this condition, the occupancy grid mapping technique tends to be the most versatile approach, which allows the combination and incorporation of information also after the sensor data has been mapped.

Process evaluation was implemented to be executed at runtime. The Hidden Markov Model (HMM) was applied to decode the most recent SOGM along the upcoming vehicle trajectory. This approach allowed the process evaluation along a vehicle trajectory to predict and steer machine parameters for upcoming situations. The training and decoding was performed such that intra-property-HMMs modeled the process-relevant classes for a grass mowing scenario which were linked together in a inter-property-HMM. Results showed a detection rate of over $90\,\%$ for every class in near-field situations, whereas the detection rate degraded noticeably in far-field situations. The drop performance can be explained by the map-building process. Far-field areas have only been observed a few times and are therefore prone to classification errors. Near-field areas have been observed more often and are less sensitive to similar noise. Furthermore, detection algorithms are expected to perform better at short range. Thus, the proposed HMM approach for combining the classifications inside the SOGM has proven its capabilities to learn the process's statistics and correct combination of SOGMs to predict the correct classes. Other approaches such as boosting are applicable for classifier fusion as well. However, the structure of HMM is better suited for modelling the statistics and consecutiveness of the given processes.

However, with our proposed architecture pipeline and information processing we have shown that with each combination of classifiers, an overall increase of the F_1-score can be reached. With up to $88.8\,\%$

in a $10\,\mathrm{cm}$ cell-wise, globally mapped evaluation for obstacle scenarios, our approach represents a state-of-the-art solution for environment classification in agricultural scenarios. Similar results were achieved for mapping semantical classes so that further mowing processes can be prospectively controlled by this information. Finally, the proposed application of HMMs to decode process-relevant information directly from the SOGMs has shown that our architecture is online applicable.

6 CONCLUSION AND FUTURE WORK

In this work, we have presented an information processing architecture for global mapping and process evaluation in an agricultural grass mowing scenario. The proposed architecture consists of four components: Sensor Platform, Inverse Sensor Models, Fusion and Mapping, and Process Evaluation. The sensor platform comprises all applied sensors for localization and environment data acquisition, such as stereo camera, radar, lidar, and thermal camera. The inverse sensor models (ISMs) describe the sensors' data processing for detecting and localizing process-relevant properties and objects in the environment, like grass, vegetation, and humans. The ISMs are 2D grid-based, non-parametric representations of the detection outputs. Fusion and mapping is performed on the ISMs which are referenced and fused based on the occupancy grid mapping algorithm into a semantical occupancy grid map (SOGM) stack. Process evaluation applies a Hidden Markov Model-based approach to first train and then quantify the environment along the vehicle's trajectory to reveal process-relevant information out of the SOGMs.

To evaluate the capabilities of the mapping approach, we compared the mapping and fusion of ISMs in a static and dynamic obstacle scenario against the FieldSAFE dataset. For both scenarios, we reported detection results for individual classifiers, fusion among classifiers, and fusion among sensors. In the static case, detection and localization results improved when introducing information fusion, first through competitive fusion among classifiers detecting similar classes, and second through complementary fusion among sensors and algorithms detecting different classes. For detecting humans in the dynamic evaluation, only classifiers that were able to detect these were fused accordingly, before a grid cell clustering was applied to retrieve consistent human hypotheses. Further, the SOGM method was extended with forgetting capabilities to adapt the mapping approach to dynamic environments. Similar to the static evaluation, a combination of multiple sensors led to an overall improvement in detection of dynamic obstacles.

In future work, we want to incorporate geodata acquired by satellites, drones, or planes from which we directly derive process-relevant information into the detection pipeline. This approach will overcome issues like complex sensor registration, weather conditions, and false detections for static properties and objects in the environment, and will therefore improve and harden our setup. Further, we want to apply supervised training of the mapping from sensor-frames to the vehicle-frame for ISMs, thereby reducing heuristics and improving global localization.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

The overall contribution and workload have been distributed between TK, MK, and PC in a close collaboration. TK is responsible for subjects related to fusion, mapping, process evaluation and for evaluating the

ability of the whole architecture to detect static and dynamic obstacles and perform process evaluation. MK and PC designed and implemented detection algorithms and inverse sensor models. RNJ contributed with insight into the domain of agriculture and preparing field experiments. HK contributed with insight into machine learning and detection algorithms. UR contributed with insight into robotics and systems engineering.

FUNDING

This research is sponsored by the Innovation Fund Denmark as part of the project "SAFE-Safer Autonomous Farming Equipment" (Project No. 16-2014-0). This research/work was supported by the Cluster of Excellence Cognitive Interaction Technology "CITEC" (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG) and by the German Federal Ministry of Education and Research (BMBF) within the Leading-Edge Cluster "Intelligent Technical Systems OstWestfalenLippe" (it's OWL) and managed by the Project Management Agency Karlsruhe (PTKA). We acknowledge support for the Article Processing Charge by the Deutsche Forschungsgemeinschaft and the Open Access Publication Fund of Bielefeld University.

REFERENCES

Abidine, A. Z., Heidman, B. C., Upadhyaya, S. K., and Hills, D. J. (2004). Autoguidance system operated at high speed causes almost no tomato damage. *California Agriculture* 58, 44–47. doi:10.3733/ca.v058n01p44

Ahtiainen, J., Peynot, T., Saarinen, J., Scheding, S., and Visala, A. (2015). Learned ultra-wideband radar sensor model for augmented lidar-based traversability mapping in vegetated environments. In *Information Fusion (Fusion), 2015 18th International Conference on* (IEEE), 953–960

Apatean, A., Rusu, C., Rogozan, A., and Bensrhair, A. (2010). Visible-infrared fusion in the frame of an obstacle recognition system. In *Automation Quality and Testing Robotics (AQTR), 2010 IEEE International Conference on* (IEEE), vol. 1, 1–6

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 1027–1035

ASI (2016). Autonomous Solutions. `https://www.asirobots.com/farming/`. Accessed: 2017-08-09

Ball, D., Upcroft, B., Wyeth, G., Corke, P., English, A., Ross, P., et al. (2016). Vision-based Obstacle Detection and Navigation for an Agricultural Robot. *Journal of Field Robotics* 33, 1107–1130. doi:10.1002/rob.21644

Bechar, A. and Vigneault, C. (2017). Agricultural robots for field operations. Part 2: Operations and systems. *Biosystems Engineering* 153, 110–128. doi:10.1016/j.biosystemseng.2016.11.004

Berg, A. and Deng, J. (2015). Imagenet large scale visual recognition challenge 2015. *Challenge*

Bertozzi, M. and Broggi, A. (1998). GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Trans. Image Process.* 7, 62–81

Biber, P. (2005). Dynamic maps for long-term operation of mobile service robots. In *In Proc. of Robotics: Science and Systems (RSS*

Bouzouraa, M. E. and Hofmann, U. (2010). Fusion of occupancy grid mapping and model based object tracking for driver assistance systems using laser and radar sensors. In *2010 IEEE Intelligent Vehicles Symposium*. 294–300. doi:10.1109/IVS.2010.5548106

Bradley, D. M., Unnikrishnan, R., and Bagnell, J. (2007). Vegetation detection for driving in complex environments. In *Robotics and Automation, 2007 IEEE International Conference on* (IEEE), 503–508

Case IH (2016). Case IH Autonomous Concept Vehicle. `http://www.caseih.com/apac/en-in/news/pages/2016-case-ih-premieres-concept-vehicle-at-farm-progress-show.aspx`. Accessed: 2017-08-09

Cho, S. I. and Lee, J. H. (2000). Autonomous speedsprayer using differential global positioning system, genetic algorithm and fuzzy control. *Journal of Agricultural Engineering Research* 76, 111–119. doi:10.1006/jaer.1999.0503

Christiansen, P., Kragh, M., Steen, K. A., Karstoft, H., and Jørgensen, R. N. (2017). Platform for evaluating sensors and human detection in autonomous mowing operations. *Precision Agriculture* 18, 350–365. doi:10.1007/s11119-017-9497-6

Christiansen, P., Nielsen, L. N., Steen, K. A., Jørgensen, R. N., and Karstoft, H. (2016a). Deepanomaly: Combining background subtraction and deep learning for detecting obstacles and anomalies in an agricultural field. *Sensors* 16, 1904

Christiansen, P., Sørensen, R., Skovsen, S., Jæger, C. D., Jørgensen, R. N., Karstoft, H., et al. (2016b). Towards autonomous plant production using fully convolutional neural networks. In *International Conference on Agricultural Engineering 2016*. 1–8

Christiansen, P., Steen, K., Jørgensen, R., and Karstoft, H. (2014). Automated detection and recognition of wildlife using thermal cameras. *Sensors* 14, 13778–13793

Dalal, N. and Triggs, B. (2005). INRIA person dataset. *Online: http://pascal. inrialpes. fr/data/human*

Davis, J. W. and Sharma, V. (2007). Background-subtraction using contour-based fusion of thermal and visible imagery. *Computer Vision and Image Understanding* 106, 162 – 182. doi:http://dx.doi.org/10.1016/j.cviu.2006.06.010. Special issue on Advances in Vision Algorithms and Systems beyond the Visible Spectrum

Dima, C., Vandapel, N., and Hebert, M. (2004). Classifier fusion for outdoor obstacle detection. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004* (IEEE), vol. 1, 665–671 Vol.1. doi:10.1109/ROBOT.2004.1307225

Dollar, P. (2015). Piotr's computer vision matlab toolbox

Dudek, G. and Jenkin, M. (2010). *Computational Principles of Mobile Robotics* (New York, NY, USA: Cambridge University Press), 2nd edn.

Elfes, A. (1990). Occupancy grids: A stochastic spatial representation for active robot perception. In *Proceedings of the Sixth Conference on Uncertainty in AI*. vol. 2929

Emmi, L., Gonzalez-De-Soto, M., Pajares, G., and Gonzalez-De-Santos, P. (2014). New trends in robotics for agriculture: Integration and assessment of a real fleet of robots. *The Scientific World Journal* 2014. doi:10.1155/2014/404059

Everingham, M., Eslami, S., and Gool, L. V. (2013). The pascal visual object classes challenge–a retrospective. *Homepages.Inf.Ed.Ac.Uk*

Fleischmann, P. and Berns, K. (2015). A Stereo Vision Based Obstacle Detection System for Agricultural Applications. *Field and Service Robotics* , 1–14

Garcia, R., Aycard, O., and Vu, T.-d. (2008). High Level Sensor Data Fusion for Automotive Applications using Occupancy Grids , 17–20

Griepentrog, H. W., Andersen, N. A., Andersen, J. C., Blanke, M., andT.E. Madsen, O. H., Nielsen, J., et al. (2009). Safe and reliable: further development of a field robot. *Precision agriculture '09* , 857–866

Hähnel, D. (2004). *Mapping with Mobile Robots*. Ph.D. thesis, University of Freiburg

Harvest Automation (2012). HV-100. https://www.public.harvestai.com. Accessed: 2017-08-09

Häselich, M., Arends, M., Wojke, N., Neuhaus, F., and Paulus, D. (2013). Probabilistic terrain classification in unstructured environments. *Robotics and Autonomous Systems* 61, 1051–1059. doi:10.1016/j.robot.2012.08.002

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science* 313, 504–507. doi:10.1126/science.1127647

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., et al. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22Nd ACM International Conference on Multimedia* (New York, NY, USA: ACM), MM '14, 675–678

Konrad, M., Nuss, D., and Dietmayer, K. (2012). Localization in digital maps for road course estimation using grid maps. In *2012 IEEE Intelligent Vehicles Symposium*. 87–92

Korthals, T., Barther, M., Schöpping, T., Herbrechtsmeier, S., and Rückert, U. (2016). Occupancy grid mapping with highly uncertain range sensors based on inverse particle filters. In *ICINCO 2016 - Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics*. vol. 2

Korthals, T., Exner, J., Schöpping, T., and Hesse, M. (2017a). Semantical Occupancy Grid Mapping Framework. In *European Conference on Mobile Robotics* (IEEE)

Korthals, T., Kragh, M., Christiansen, P., and Rückert, U. (2017b). Towards Inverse Sensor Mapping in Agriculture. In *IROS 2017 Workshop on Agricultural Robotics: learning from Industry 4.0 and moving into the future* (Vancouver)

Koubaa, A. (2016). *Robot Operating System (ROS): The Complete Reference*, vol. 1 (Springer International Publishing). doi:10.1007/978-3-319-26054-9

Kragh, M., Christiansen, P., Korthals, T., Jungeblut, T., Karstoft, H., and Jørgensen, R. N. (2016). Multi-Modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture. In *International Conference on Agricultural Engineering* (International Commission of Agricultural and Biosystems Engineering)

Kragh, M., Jørgensen, R. N., and Pedersen, H. (2015). Object Detection and Terrain Classification in Agricultural Fields Using 3D Lidar Data. In *Computer Vision Systems : 10th International Conference, ICVS 2015, Proceedings*, vol. 9163. 188–197. doi:10.1007/978-3-319-20904-3_18

Kragh, M. and Underwood, J. (2017). Multi-modal obstacle detection in unstructured environments with conditional random fields. *CoRR* abs/1706.02908

Kragh, M. F., Christiansen, P., Laursen, M. S., Larsen, M., Steen, K. A., Green, O., et al. (2017). Fieldsafe: Dataset for obstacle detection in agriculture. *Sensors* 17. doi:10.3390/s17112579

Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* , 1097–1105

Kubota (2017). Kubota. http://www.kubota-global.net/news/2017/20170125.html. Accessed: 2017-08-16

Laible, S., Khan, Y. N., and Zell, A. (2013). Terrain classification with conditional random fields on fused 3D LIDAR and camera data. In *2013 European Conference on Mobile Robots* (IEEE), 172–177. doi:10.1109/ECMR.2013.6698838

Lalonde, J.-F., Vandapel, N., Huber, D. F., and Hebert, M. (2006). Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics* 23, 839–861. doi:10.1002/rob.20134

Lely (2016). Lely Discovery Collector. https://www.lely.com/the-barn/housing-and-caring/discovery-collector. Accessed: 2017-08-09

Liggins, M. E., Hall, D. L., and Llinas, D. (2001). *Handbook of multisensor data fusion*

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft COCO: Common objects in context. *arXiv preprint arXiv ...* cs.CV, 1–15

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3431–3440

Luettel, T., Himmelsbach, M., and Wuensche, H.-J. (2012). Autonomous Ground Vehicles—Concepts and a Path to the Future. *Proceedings of the IEEE* 100, 1831–1839. doi:10.1109/JPROC.2012.2189803

Moore, T. and Stouch, D. (2014). A generalized extended kalman filter implementation for the robot operating system. In *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)* (Springer)

Moorehead, S. S. J., Wellington, C. K. C., Gilmore, B. J., and Vallespi, C. (2012). Automating orchards: A system of autonomous tractors for orchard maintenance. *Proc. IEEE Int. Conf. Intelligent Robots and Systems, Workshop on Agricultural Robotics* , 632

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., et al. (2014). The role of context for object detection and semantic segmentation in the wild. In *2014 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE), 891–898

Munkres, J. (1957). Algorithms for the Assignment and Transportation Problems. doi:10.1137/0105003

Nam, W., Dollár, P., and Han, J. H. (2014). Local decorrelation for improved detection. *Adv. Neural Inf. Process. Syst.* , 1–9

Ollis, M. and Stentz, A. (1997). Vision-based perception for an automated harvester. *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97* 3, 1838–1844. doi:10.1109/IROS.1997.656612

Papadakis, P. (2013). Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence* 26, 1373–1385. doi:10.1016/j.engappai.2013.01.006

Pathak, K., Birk, A., Poppinga, J., and Schwertfeger, S. (2007). 3D Forward sensor modeling and application to occupancy grid based sensor fusion. *IEEE International Conference on Intelligent Robots and Systems* 2, 2059–2064. doi:10.1109/IROS.2007.4399406

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*

Redmon, J. (2013). Darknet: Open source neural networks in c. *h ttp://pjreddie. com/darknet* 2016

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, Real-Time object detection

Redmon, J. and Farhadi, A. (2016). YOLO9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*

Reina, G. and Milella, A. (2012). Towards Autonomous Agriculture: Automatic Ground Detection Using Trinocular Stereovision. *Sensors* 12, 12405–12423. doi:10.3390/s120912405

Reina, G., Milella, A., Rouveure, R., Nielsen, M., Worst, R., and Blas, M. R. (2016). Ambient awareness for agricultural robotic vehicles. *Biosystems Engineering* 146, 114–132. doi:10.1016/j.biosystemseng.2015.12.010

Ross, P., English, A., Ball, D., Upcroft, B., and Corke, P. (2015). Online novelty-based visual obstacle detection for field robotics. *Proceedings - IEEE International Conference on Robotics and Automation* 2015-June, 3935–3940. doi:10.1109/ICRA.2015.7139748

Rovira-Mas, F., Reid, J., Han, S., et al. (2005). Obstacle detection using stereo vision to enhance safety of autonomous machines. *Transactions of the ASAE* 48, 2389–2397

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for Large-Scale image recognition , 1–13

Sofman, B., Bagnell, J. A., and Stentz, A. (2010). Anytime online novelty detection for vehicle safeguarding. *Proceedings - IEEE International Conference on Robotics and Automation* , 1247–1254doi:10.1109/ROBOT.2010.5509357

Stachniss, C. (2009). *Robotic Mapping and Exploration*. doi:10.1007/978-3-642-01097-2

Stentz, A., Dima, C., Wellington, C., Herman, H., and Stager, D. (2002). A system for semi-autonomous tractor operations. *Autonomous Robots* 13, 87–104. doi:10.1023/A:1015634322857

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics* (Cambridge, Mass.: MIT Press)

Underwood, J. P., Hill, A., Peynot, T., and Scheding, S. J. (2010). Error modeling and calibration of exteroceptive sensors for accurate mapping applications. *Journal of Field Robotics* 27, 2–20

Van den Bergh, M., Boix, X., Roig, G., and Van Gool, L. (2015). SEEDS: Superpixels Extracted Via Energy-Driven Sampling. *International Journal of Computer Vision* 111, 298–314. doi:10.1007/s11263-014-0744-2

Vasquez, A., Kollmitz, M., Eitel, A., and Burgard, W. (2017). Deep detection of people and their mobility aids for a hospital robot. *CoRR* abs/1708.00674

Walter, O., Korthals, T., Haeb-Umbach, R., and Raj, B. (2013). A hierarchical system for word discovery exploiting DTW-based initialization. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2013 - Proceedings*. 386–391. doi:10.1109/ASRU.2013.6707761

Wellington, C., Courville, A., and Stentz, A. T. (2005). Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection. In *Proceedings of Robotics: Science and Systems*. vol. 17, 251–60. doi:10.1.1.64.1208

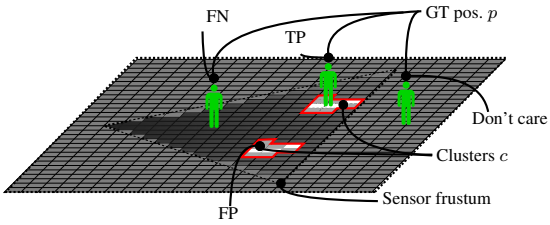
Wellington, C. and Stentz, A. (2004). Online Adaptive Rough-Terrain Navigation in Vegetation. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004* 1, 96–101 Vol.1. doi:10.1109/ROBOT.2004.1307135

Winner, H. (2015). *Handbuch Fahrerassistenzsysteme - Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort* (Wiesbaden: Vieweg+Teubner Verlag)

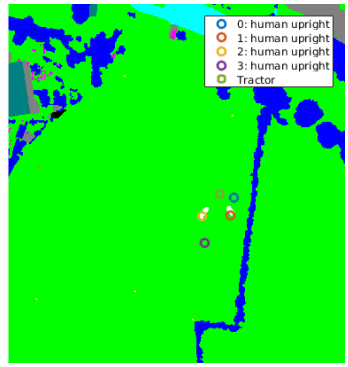
Wu, T.-F., Lin, C.-J., and Weng, R. C. (2004). Probability Estimates for Multi-class Classification by Pairwise Coupling. *Journal of Machine Learning* 5, 975–1005. doi:10.1016/j.visres.2004.04.006

Yang, L. and Noguchi, N. (2012). Human detection for a robot tractor using omni-directional stereo vision. *Computers and Electronics in Agriculture* 89, 116–125

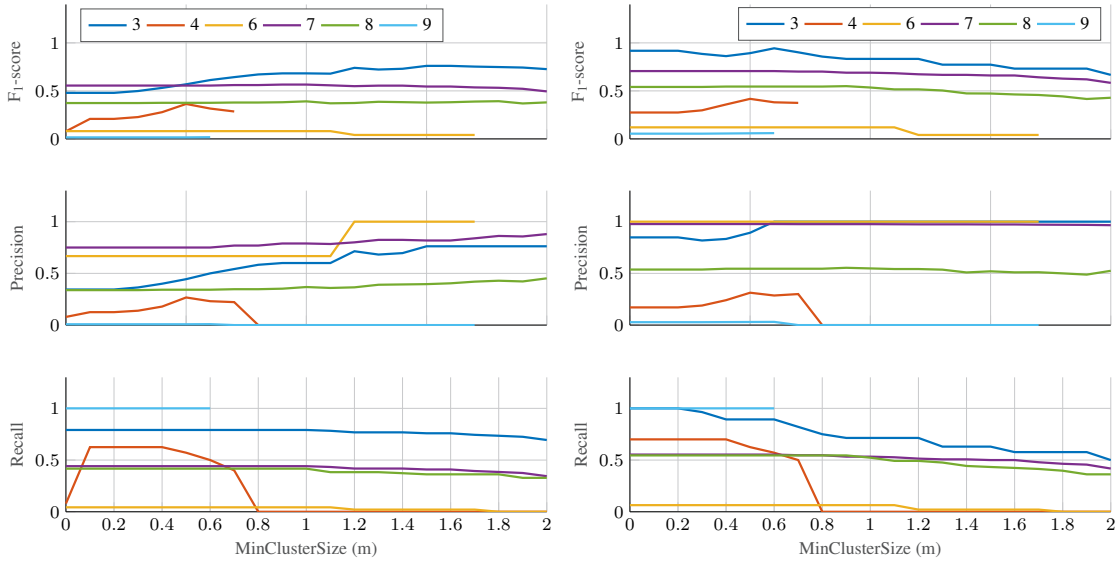
FIGURE CAPTIONS



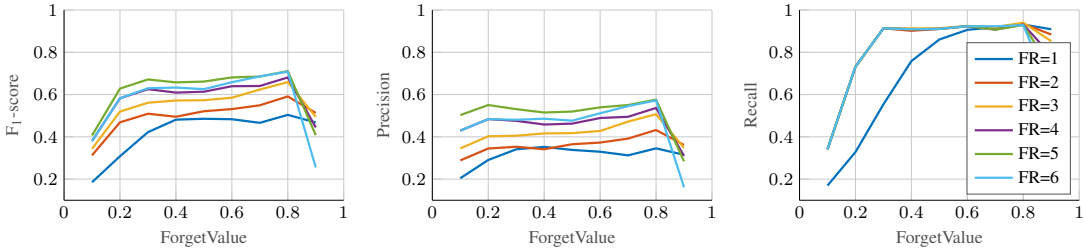
(10a) Dynamic evaluation of TP, FP, and FN acquisition.



(10b) Dynamic evaluation example. Valid clusters are colored white, and GT positions of humans and the tractor are overlaid.

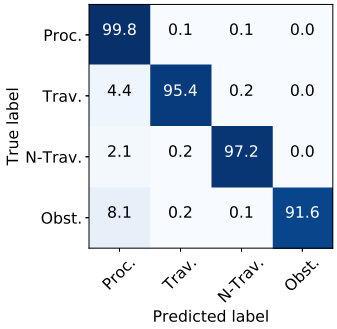


(10c) Precision, recall, and F_1-score over increasing minimum cluster size for different setups from Table 3. *ForgetRate* $= 6$ and *ForgetValue* $= 0.8$. Left: No dilation. Right: Dilation by vehicle radius of $2.5\,\mathrm{m}$.

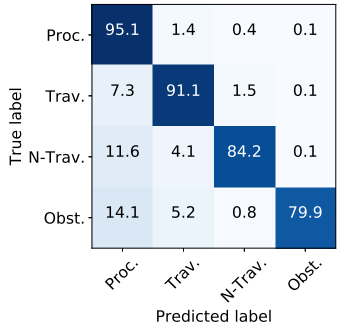


(10d) F_1-score over increasing *ForgetValue* with different *ForgetRate* (FR).

Figure 10. Evaluation of dynamic scenario.



(11a) Confusion matrix for near field.



(11b) Confusion matrix for far field.

Figure 11. Results for decoding the corresponding classes along the trajectory at close and far range.

Paper 8

Towards Inverse Sensor Mapping in Agriculture
Timo Korthals, Mikkel Fly Kragh, Peter Christiansen, Ulrich Rückert

Peer reviewed
Presented at the International Conference on Intelligent Robots and Systems (IROS), Workshop on “Agricultural Robotics: learning from Industry 4.0 and moving into the future”, September 2017, Vancouver, Canada

Towards Inverse Sensor Mapping in Agriculture

Timo Korthals[1], Mikkel Kragh[2], Peter Christiansen[2], and Ulrich Rückert[1]

***Abstract*— In recent years, the drive of the Industry 4.0 initiative has enriched industrial and scientific approaches to build self-driving cars or smart factories. Agricultural applications benefit from both advances, as they are in reality mobile driving factories which process the environment. Therefore, acurate perception of the surrounding is a crucial task as it involves the goods to be processed, in contrast to standard indoor production lines. Environmental processing requires accurate and robust quantification in order to correctly adjust processing parameters and detect hazardous risks during the processing. While today approaches still implement functional elements based on a single particular set of sensors, it may become apparent that a unified representation of the environment compiled from all available information sources would be more versatile, sufficient, and cost effective. The key to this approach is the means of developing a common information language from the data provided. In this paper, we introduce and discuss techniques to build so called inverse sensor models that create a common information language among different, but typically agricultural, information providers. These can be current live sensor data, farm management systems, or long term information generated from previous processing, static drone images, or satellites. In the context of Industry 4.0, this enables the interoperability of different agricultural systems and allows information transparency.**

I. Introduction

Agricultural vehicles are complex, mobile processors of biological products that operate in unstructured and constantly changing environment. While the operation of these vehicles was initially relatively simple, today their setup and use requires trained specialists due to the requirement of increasing efficiency and lowering overall costs. However, without automation and the augmenting of parameter optimization in the process chain, throughputs, and farming yields would be much smaller than usual. For instance, automated steering systems employed in harvesting use LiDAR systems to scan the area between the crop and stubble in order to automatically guide the harvester along the edge; and seed drills save GPS data and the machine parameters of sowing which are used later to minimize the utilization of fertilizer spreaders.

Focusing the automation and in particular its implementation, all applications follow the same paradigm of having a distinctive set of sensors, a processing unit, and an actuator interface to steer the vehicle or manipulate process parameters. While this approach allows simple, distributed and modular modification, with increases in automated functionality its installation and maintenance becomes unfeasible due to the sheer number of sensors and processing units required. Furthermore, the potential for sensor fusion is completely squandered. An alternative approach is pursued by the authors, that of building a common inner semantical representation of the environment based on occupancy grid maps, from which all further automation is derived [1], [2]. These grid maps are arranged in multiple overlapping layers, where each one is occupied by localized classifications.

While the authors have already provided a proof-of-concept of semantical grid mapping approaches in agriculture [3], requisite information and instructions for building sensor models based on sensors and other data sources is still lacking. In contrast to robotic and automotive approaches, where grid mapping based applications are well known, agricultural environments and applications especially vary greatly and therefore have to be treated accordingly. With respect to Fig. 1 and [4], this contribution focuses on the *Inverse Sensor Modeling* component.

The paper is organized as follows: Section II presents a brief introduction to occupancy grid maps, their extension to the semantical representation. Section III presents the gathered experience and approaches to building sensor models derived from previous agricultural research projects. Finally, Section IV presents further ideas and points to next steps in agricultural applications in Industry 4.0.

II. Related Work

Occupancy grid maps are used in static obstacle detection for robotic systems, which are a well-known and a commonly studied scientific field [5], [6], [7]. They are a component of almost all navigation and collision avoidance systems designed to maneuver through cluttered environments. Another important application is the creation of obstacle maps for traversing an unknown area and the recognition of known obstacles, so supporting the localization. Recently, occupancy grid maps have been applied to combine LiDAR and RADAR in automotive applications, with the goal of creating a harmonious, consistent and complete representation of the vehicle's environment as a basis for advanced driver assistance systems [8], [9], [10].

A. Occupancy Grid Mapping

Two-dimensional occupancy grid maps (OGM) were originally introduced by Elfes [11]. In this representation, the environment is subdivided into a regular array or a grid of quadratic cells. The resolution of the environment representation directly depends on the size of the cells. In addition to

[1]Bielefeld University, Cluster of Excellence Cognitive Interaction Technologies, Cognitronics & Sensor Systems, Inspiration 1, 33619 Bielefeld, Germany, http://www.ks.cit-ec.uni-bielefeld.de/, {tkorthals, rueckert} @cit-ec.uni-bielefeld.de

[2]Aarhus University, Department of Engineering, Finlandsgade 22, DK-8200 Aarhus N, Denmark http://eng.au.dk/, {mkha, pech} @eng.au.dk

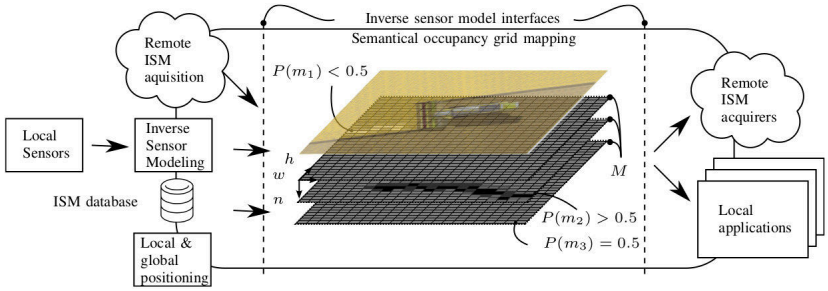


Fig. 1: Semantic occupancy grid mapping framework

this compartmentalization of space, a probabilistic measure of occupancy is associated with each cell. This measure takes any real number in the interval $[0, 1]$ and describes one of the two possible cell states: unoccupied or occupied. An occupancy probability of 0 represents a space that is definitely unoccupied, and a probability of 1 represents a space that is definitely occupied. A value of 0.5 refers to an unknown state of occupancy.

An occupancy grid is an efficient approach to representing uncertainty, combining multiple sensor measurements at the decision level, and to incorporating different sensor models [10]. To learn an occupancy grid M given sensor information z, different update rules exist [5]. For the authors' approach, a Bayesian update rule is applied to every cell $m \in M$ at position (w, h) as follows: Given the position x_t of a vehicle at time t, let $x_{1:t} = x_1, \ldots, x_t$ be the positions of the vehicle's individual steps until t, and $z_{1:t} = z_1, \ldots, z_t$ the environmental perceptions. For each cell m of the occupancy probability grid the probability that this cell is occupied by an obstacle. Thus, occupancy probability grids seek to estimate

$$P\left(m|z_{1:t}, x_{1:t}\right) = \mathrm{Odds}^{-1}\left(\prod_{t=1}^{T} \underbrace{\frac{P\left(m|z_t, x_t\right)}{1 - P\left(m|z_t, x_t\right)}}_{\mathrm{Odds}(P(m|z_t, x_t))}\right) \quad (1)$$

This equation already describes the online capable, recursive update rule that populates the current measurement z_t to the grid, where $P\left(m|z_{1:t}, x_{1:t}\right)$ is the so called inverse sensor model (ISM). The ISM is used to update the OGM in a Bayesian framework, which deduces the occupancy probability of a cell, given the sensor information.

B. Extension to Agriculture Applications

The adaptation of OGM techniques to agricultural applications appears to be merely a matter of time but is not that obvious and intuitive to apply on the second sight. Robotic and automotive applications have in common that they both want to detect non-traversable areas or objects occupying their path. Such unambiguous information is used to quantify the whole environment sufficiently for all derivable tasks, such as path planning or obstacle avoidance, to be completed. When assumptions like a flat operational plane or minimum obstacle heights are made, sensors frustums oriented parallel to the ground are sufficient for all tasks

In agricultural applications, obstacle recognition is not essential as they act on and process their environment. Therefore, quantification of the environment involves features such as processed areas, processability, crop quality, density, and maturity level in addition to traversability. In order to map these features, single occupancy grid maps are no longer sufficient and therefore, semantic occupancy grid maps that allow different classification results to be mapped are used. Furthermore, sensor frustums are no longer oriented parallel to the ground, but rather oriented at an angle to gather necessary crop information (cf. Fig. 2).

The extension to semantic occupancy grid maps (SOGM) or inference grids is straightforward and is defined by an OGM M with W cells in width, H cells in height, and N semantic layers (c.f. Fig. 1):

$$M : \{1, \ldots, W\} \times \{1, \ldots, H\} \rightarrow m = \{0, \ldots, 1\}^N \quad (2)$$

Compared to a single layer OGM which allows the classification into three classes $\left\{\text{occupied}, \overline{\text{occupied}}, \text{unknown}\right\}$, the SOGM supports a maximum of $\left|\left\{\text{occupied}, \overline{\text{occupied}}, \text{unknown}\right\}\right|^N = 3^N$ different classes allowing much higher differentiability in environment and object recognition. The corresponding ISMs are fused by means of the occupancy grid algorithm to their nth associated semantical occupancy grid.

The location of information in the maps is required to be completed by *mapping under known poses* approaches [6]. As proposed by REP-105[1] and realized by the authors in [4], information is mapped locally via Kalman filtered odometry and inertial navigation measurement. The maps themselves are globally referenced which on the one side allows smooth local mapping in the short term without the discrete jumps caused by global positioning systems using a Global Navigation Satellite System (GNSS), but also allows global consistent storing and loading of information.

While the actual features are very diverse of agriculture applications, this publication does not primarily focus on classification, but rather on geographical interpretation and sensor building.

[1] http://www.ros.org/reps/rep-0105.html

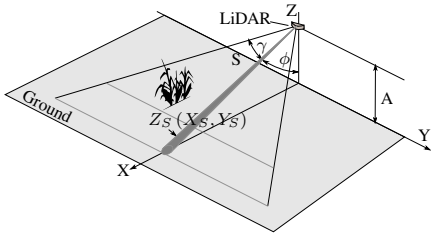


Fig. 2: Ground oriented LiDAR for crop rectification

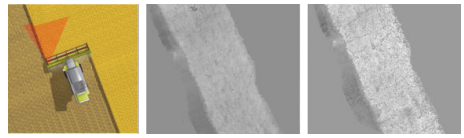


Fig. 3: Harvesting scenario (left), resulting SOGM from crop classification ISM with (middle) and without (right) error propagation

III. Explicit ISM Generation for Specific Sensors

A. Local Sensor Based ISM

1) LiDAR based Mapping: LiDAR sensors measure the distance to an object and depending on their capabilities, also the reflectance. The distance can directly be used to deduce free (s.t. the area between the measured distance and the sensor) and occupied space (s.t. the location of measured distance) in a planar environment. This is commonly utilized for robotic and automotive tasks, where a well-known inverse sensor modelling technique directly derives the corresponding ISM. In agriculture, however, it is common for LiDAR sensors to face downwards as shown in Fig. 2, in order to detect the soil or crop that needs to be processed. This results in the circumstance that the measurement can only be taken at the corresponding target point, and no implications can be done along the measurement.

Naively mapping the related classification in the point of measurement in the vehicles coordinate frame would result in scattered maps from which further applications are hardly derivable (c.f. Fig. 3). Therefore, the actual Gaussian measurement uncertainty σ_S needs to be introduced as in the common planar model, but with its appropriate error propagation. Assuming σ_ϕ, σ_ξ, σ_γ beeing gaussian noise in the angular positioning caused by vehicle's steering, and σ_x, σ_y, σ_z to be the positioning caused by vibrations of the vehicle it is possible to calculate the resulting full covariance matrix $\sum_{X_S}$ at the point of interest as follows: First, the transformation of the scalar distance measurement S in the LiDAR frame to the euclidean point X_S in the vehicle frame is

$$X_S = \begin{pmatrix} c_\phi c_\gamma \\ c_\xi s_\gamma + c_\gamma s_\phi s_\xi \\ s_\xi s_\gamma - c_\gamma s_\phi c_\xi \end{pmatrix} S + \mathrm{T}(x, y, z) \tag{3}$$

where T is the translation between the sensor and the vehicle frame. For error propagation, the functions need to be linearized by calculating the Jacobian:

$$J^{\mathrm{T}} = \begin{pmatrix} c_\phi c_\gamma & c_\xi s_\gamma + c_\gamma s_\phi s_\xi & s_\xi s_\gamma - c_\gamma s_\phi c_\xi \\ -S s_\phi c_\gamma & S c_\gamma c_\phi s_\xi & -S c_\gamma c_\phi c_\xi \\ -S c_\phi s_\gamma & S c_\xi c_\gamma - S s_\gamma s_\phi s_\xi & S s_\xi c_\gamma + S s_\gamma s_\phi c_\xi \\ 0 & -S s_\xi s_\gamma + S c_\gamma s_\phi c_\xi & S c_\xi s_\gamma + S c_\gamma s_\phi s_\xi \end{pmatrix}^{\mathrm{T}} \tag{4}$$

$$\sum_{X_S} = J \,\mathrm{diag}(\sigma_s^2, \sigma_\phi^2, \sigma_\gamma^2, \sigma_\xi^2) J^{\mathrm{T}} + \mathrm{diag}(\sigma_x^2, \sigma_y^2, \sigma_z^2) \tag{5}$$

The Jacobian is a function of its arguments $J(S, \phi, \gamma, \xi)$, which means that it is required to be evaluated for every new sensor measurement. Equation 5 describes the full covariance matrix which can be applied to calculate the uncertainty distribution for every measurement.

Two assumptions have been made in this model to make the error model tractable: first, that the uncertainty in angular movements resides in the coordinate frame of the laser scanner and second, that the uncertainty in translation is uncorrelated from the angular ones. The assumptions do not fully hold, due to the fact that rolling, pitching and yawing do not occure in the laser scanner frame, but in some other arbitrary frame, depending on the current ground conditions and vehicle's steering. To simplify the model even more, the uncertainty in z can be omitted, because in the later sensor modeling component, only the projection into the xy-plane is important. Further, rolling is omitted as it is negligible in comparison to the other influences [12]:

$$\begin{aligned} X'_S &= \begin{pmatrix} c_\phi c_\gamma \\ s_\gamma \\ -c_\gamma s_\phi \end{pmatrix} S + \mathrm{T}(x, y, z) \\ J' &= \begin{pmatrix} c_\phi c_\gamma & -S s_\phi c_\gamma & -S c_\phi s_\gamma \\ s_\gamma & 0 & S c_\gamma \end{pmatrix} \\ \sum_{X'_S} &= J' \,\mathrm{diag}(\sigma_s^2, \sigma_\phi^2, \sigma_\gamma^2) J'^{\mathrm{T}} + \mathrm{diag}(\sigma_x^2, \sigma_y^2) \end{aligned} \tag{6}$$

The influences of error propagation are depicted in Fig. 3 where a two class classifier for crop derives the ISMs which are mapped to the global coordinate system. The resulting map without error propagation is very sparse which makes further functionality derivation without heuristical post processing unfeasible. Introducing error propagation and respecting the model uncertainties, on the other hand, results in a much more sufficient and consistent map where further classification can easily be applied.

Further improvements in classification can be achieved by first mapping the raw LiDAR data to a globally referenced representation from which further ISMs with much higher quality can be derived. More advanced LiDAR systems scanning in multiple planes bypass the raw mapping and directly enable rich classifiers like Support Vector Machines to process the data as proposed by [3].

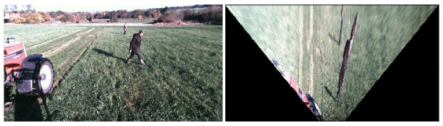


Fig. 4: Inverse Perspective Mapping of RGB image

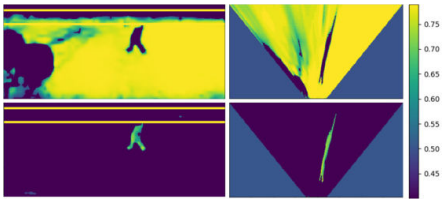


Fig. 5: (Left) Grass and human predictions in a mowing application classified by a fully convolutional network for semantic segmentation [15] and the corresponding ISMs generated by IPM (right)

2) Inverse Perspective Mapping: Inverse Perspective Mapping (IPM) is a geometrical transformation that projects an image to a ground plane surface as shown in Fig. 4. For a flat surface, the perspective effect is removed by transforming the viewpoint from a camera view to a birds eye view. This technique has been used in automotive applications where assumptions about camera pose and a flat world with respect to the street are sufficient [13], [12]. However, even slight deviations in camera inclination and height result in large errors, more advanced, adaptive techniques have been developed which calculate the camera pose online by using the borders of the road or lane markers [14].

However, an unstructured agricultural environments does permit such dynamic techniques and thus, they are either treated as a static scenario, where the camera pose relative to ground surface does not change, or the transformation between the extrinsic and flat plane is calculated dynamically with support of an inertial measurement unit (IMU). The whole IPM for mapping image coordinates $\mathbf{x}_\mathrm{P}|_\mathrm{px} = (u, v, 1)^\mathrm{T}$ to surface $\mathbf{x}_\mathrm{FP}|_\mathrm{m} = (x, y, z \equiv 0, 1)^\mathrm{T}$ is defined by three parameter transformations: the intrinsic ${}^\mathrm{P}\mathbf{T}_\mathrm{C}$ from the camera perspective to the camera frame, the extrinsic ${}^\mathrm{C}\mathbf{T}_\mathrm{V}$ from the camera frame to the vehicle frame, and ${}^\mathrm{V}\mathbf{T}_\mathrm{FP}$ which transforms from the vehicle frame to the flat plane (FP) frame. This leads to

$$\mathbf{x}_\mathrm{P}|_\mathrm{px} = {}^\mathrm{P}\mathbf{T}_\mathrm{C} \cdot {}^\mathrm{C}\mathbf{T}_\mathrm{V} \cdot {}^\mathrm{V}\mathbf{T}_\mathrm{FP} \cdot \mathbf{x}_\mathrm{FP}|_\mathrm{m} \tag{7}$$

To build the actual ISM, the image first needs to be classified and then transformed to the flat plane by means of Equation 7 (c.f. Fig. 5).

Values of an ISM are the probability of a grid cell being occupied by a giving classification. As indicated in Fig. 5, the area that is not visible by the camera is set to 0.5 to represent the fact no information is provided for areas that are not visible to the camera. Visible areas with no detections are set below 0.5 to indicate that the area is not expected to be occupied by the given class. Values above 0.5 indicate that the area is expected to be occupied by the given class.

Fig. 6: Input image (left), classification based on semantic segmentation (middle) and corresponding ISM with detection cut-off after class occurrence along the focal axis

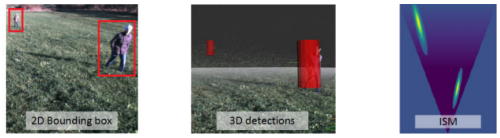


Fig. 7: Bounding box detection to ISM

For detecting flat class elements such as road-lane markings or grass, the IPM algorithm is able to provide good approximations of the actual inverse perspective mapping. Elevated elements violate the IPM ground plane assumption and will stretch elements unnaturally and incorrectly across large areas as indicated in Fig. 4.

To avoid the stretching artifacts of tall objects, different approaches are proposed. A naive approach for pixel based classifiers states that all objects classified as being other than ground are standing perpendicular on the ground. Therefore, one can perform a ray trace along the focal axis and mark all cells behind a detected object as unknown (c.f. Fig. 6) [16], [3].

Another approach generates three dimensional object location hypotheses by first estimating the distance to the corresponding detection. This can be achieved by either using the abovementioned naive approach or using a depth sensor like a stereo camera or LiDAR which is registered to the camera.

Second, when using classifiers like YOLO [17] which offers classified bounding boxes, the four bounding box corners are mapped to real world coordinates using the estimated distance to a detection and the intrinsic camera parameters. The bounding box position and extent are derived in 3D and is represented as depicted in Fig. 7 by cylinder specified by a center, height, and width.

Detections are mapped to values above 0.5 with a Gaussian distribution to indicate the existence of an obstacle with corresponding localization uncertainties. The localization uncertainty for the camera depends on the radial coordinate (distance to the object) and angular coordinate (angle to object), where accuracy degrades with increasing distance and angle. The procedure for converting a 2D bounding box to an ISM using distance estimates is presented in Fig. 7. Using the estimated distance of a detected object and the intrinsic camera parameters, the four bounding box corners are mapped to world coordinates.

Lastly, the concept of contradicting IPM is introduced for crop processing in harvesting scenarios. In comparison

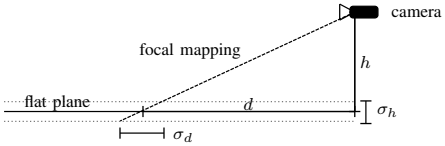


Fig. 8: Simplified error assumption in flat plane assumption according to height

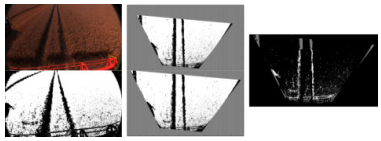


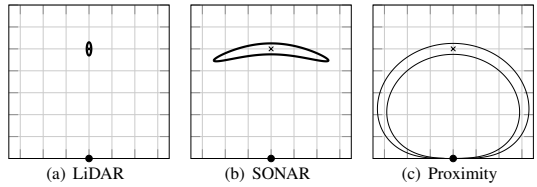
Fig. 9: RGB input image and scanline based classification for crop plane (left), inverse perspective mapping of classification for crop and ground plane (middle) and corresponding fused contradicting ISM (right)

with the abovementioned IPM scenarios, this discrimination is necessary as the camera rectifies no common ground in the lower areas of the image as depicted in Fig. 9 which refutes former assumptions. Neglecting this fact would result in drastically wrong localization of detections, as visualized in Fig. 8, which indicates that the localization error σ_d in depth d depends on the error σ_h of height h as follows:

$$\sigma_d = \frac{d}{h}\sigma_h. \tag{8}$$

If this simple error propagation is applied to a hypothetical example of small crop with for example a height of 0.5 meters and a camera installation height of 1.5 meters where a feature 10 meters away should be mapped, the resultant error is one of 3 meters. Therefore, two flat plane assumptions are calculated, one for the ground and one for the crop height resulting in two different ISMs. These can then be combined by Dempsters rule of combination leading to contradictions [18], which is visualized in Fig. 9. From the emerging contradictions in Fig. 9 (right), it can be seen that vehicle traces appear which are actually the contradicting occlusion in both IPMs.

3) Ambiguous Sensor Mapping: Ambiguous sensor readings originate from sensors with very bad angular or distance resolution by definition of the authors. As depicted in Fig. 10 LiDAR systems can achieve very accurate positioning and are therefore the preferred sensors for mapping. However, they are by far the most cost- and power intensive systems. Other sensing techniques are more cost and power efficient but are commonly neglected due to their high noise or inaccuracy. Nevertheless, the authors have demonstrated that even with poorly embedded sensors, sufficient environment detection can be achieved [19] by designing an inverse particle filter which samples from the sensors uncertainty distribution. At present, this technique has only been applied in laboratory conditions and therefore, real agricultural applications remain pending.



(a) LiDAR (b) SONAR (c) Proximity

Fig. 10: Standard error contour of qualitative sensor cones (•: Sensor position, x: Obstacle, -)

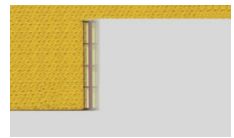


Fig. 11: Top view of crop field with an applied inverse sensor model for the cutter bar: gray shaded area being of high probability that the cutter bar has been applied on that region

B. Application Models

Application models are straight forward to implement and only depends on the localizing accuracy. Building such a model is only dependent on the geometrical shape of the agricultural implement. That means on the other hand, that ISM is a static and primitive shape in the local frame of the vehicle which leaves a probabilistic footprint where the implement has been applied to the crop as depicted in Fig. 11. When incorporating inaccurate localization, the shape needs to be transformed accordingly.

C. Map Services

Geodata acquired by satellites, drones, or planes with high recording frequencies as well as its partially free availability, make this information increasingly attractive for agriculture. In this context worth mentioning are the Sentinel program[2], the hyperspectral system EnMap[3], the RapidEye constellation[4] as well as the start-up companies Skybox Imaging[5] and Planet Labs[6]. In addition, the release of the long-standing Landsat archive now offers many opportunities for agricultural applications, such as the generation of profit potential maps. There is a trend towards direct access to such data and towards appropriate image excerpts using web servers or APIs. As part of spatial data infrastructures, data (e.g. land and terrain data) are published interoperably and often free of charge via web services. In particular, Annex III of the INSPIRE Directive[7] requires EU member states to provide data. However, for a precision farming service or a precision farming application further different data sources have to be

[2] http://www.esa.int/Our_Activities/Observing_the_Earth/Copernicus/Overview4/
[3] http://www.enmap.org/
[4] http://blackbridge.com/rapideye/
[5] http://www.skyboximaging.com/
[6] https://www.planet.com/
[7] http://inspire.ec.europa.eu/

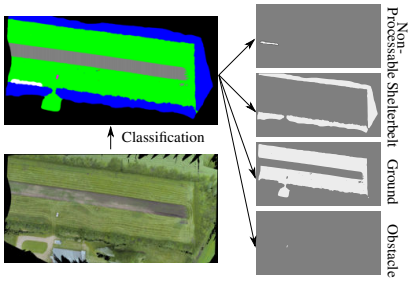


Fig. 12: Classification decomposition of hand labeled orthographic photograph [3]

linked (for example, weather data play a crucial role in most agricultural processes), or complex procedures and algorithms are required to derive the desired information from the data. Subsequent downstream services will continue to play an increasingly important role in agriculture. The European Union, for example, specifically supports the development of such services based on Copernicus data by SMEs. At the endpoint of the downstream services, information products (such as humidity maps, biomass maps and yield forecast maps) are often available, which can be integrated into other applications or devices. The combination and the inclusion of all the information sources and their derivation for the identification of machine parameters is one essential part which can be handled by ISMs. As an example, a static and classified drone image can be easily transferred to a semantic ISM by decomposing all classes and loading the appropriate area during operation (c.f. Fig. 12).

IV. Conclusion and Outlook

The authors have presented an information representation as semantic grids which can be maintained among different modalities and sources. It utilizes the idea of the ISOBUS standard, which was designed with machinery interoperability in mind, and allows every sensory source to publish or access its information in a general grid format. The main aspect of this contribution focused on different techniques, originating from literature, practical experiments, and experience, of actually building these representations.

As the acquisition and localization of data are sufficiently solved, further research will concentrate on planning and control of such diverse data. Furthermore, learning approaches have not been confronted in this application which directly maps a sensor reading to the appropriate locality and probability. These techniques were introduced by Thrun [6] and have been applied by the authors. However, following the engineering path of building inverse sensor models is far more robust and intuitive. At present, only a few approaches are known to the authors and therefore, more applications extending from direct control architectures up to holistic farm management systems are of great interest. Approaching rich control architectures in agricultural environments allows an interesting area of overlap between robotics and Industry 4.0 to emerge, s.t. simultaneously planning and processing. Mathematical frameworks exist, where in agriculture the particular issue will driven by the information representation and how it is incorporated into environmental processing.

Acknowledgment

This research was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG) and by the German Federal Ministry of Education and Research (BMBF) within the Leading-Edge Cluster "Intelligent Technical Systems OstWestfalenLippe" (it's OWL) and managed by the Project Management Agency Karlsruhe (PTKA).

References

[1] T. Korthals, A. Skiba, and T. Krause, "Evidenzkarten-basierte Sensorfusion zur Umfelderkennung und Interpretation in der Ernte," in *Informatik in der Land-, Forst und Ernährungswirtschaft*, 2016, pp. 15–18.

[2] ——, "Einsatz Event-Basierter Systemarchitektur für Erntemaschinen zur Elektronischen Umfelderkennung," in *74. Tagung LAND.TECHNIK*. VDI e.V., 2016.

[3] M. Kragh, P. Christiansen, T. Korthals, T. Jungeblut, H. Karstoft, and R. N. Jørgensen, "Multi-Modal Obstacle Detection and Evaluation of Occupancy Grid Mapping in Agriculture," in *International Conference on Agricultural Engineering*, Aarhus, 2016.

[4] T. Korthals, J. Exner, T. Schöpping, and M. Hesse, "Semantical Occupancy Grid Mapping Framework," in *European Conference on Mobile Robotics*. IEEE, 2017.

[5] D. Hähnel, "Mapping with Mobile Robots," Ph.D. dissertation, University of Freiburg, 2004.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, Mass.: MIT Press, 2005.

[7] C. Stachniss, *Robotic Mapping and Exploration*, 2009.

[8] R. Garcia, O. Aycard, and T.-d. Vu, "High Level Sensor Data Fusion for Automotive Applications using Occupancy Grids," no. December, pp. 17–20, 2008.

[9] M. E. Bouzouraa and U. Hofmann, "Fusion of occupancy grid mapping and model based object tracking for driver assistance systems using laser and radar sensors," in *2010 IEEE Intelligent Vehicles Symposium*, 2010, pp. 294–300.

[10] H. Winner, *Handbuch Fahrerassistenzsysteme - Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*. Wiesbaden: Vieweg+Teubner Verlag, 2015.

[11] A. Elfes, "Occupancy Grids: A Stochastical Spatial Representation for Active Robot Perception," in *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, 1990.

[12] M. Konrad, D. Nuss, and K. Dietmayer, "Localization in digital maps for road course estimation using grid maps," *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 87–92, 2012.

[13] M. Bertozzi and a. Broggi, "Real-time lane and obstacle detection on the GOLD system," *Proceedings of Conference on Intelligent Vehicles*, 1996.

[14] N. Simond and P. Rives, "Homography from a vanishing point in urban scenes," *International Conference on Intelligent Robots and Systems*, pp. 1005–1010, 2003.

[15] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.

[16] S. Kohlbrecher, "Grid-based occupancy mapping and automatic gaze control for soccer playing humanoid robots," ... *Humanoid Soccer Robots* ..., no. October, 2011.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Cvpr 2016*, pp. 779–788, 2016.

[18] G. Shafer, *A Mathematical Theory of Evidence*. Princeton: Princeton University Press, 1976.

[19] T. Korthals, M. Barther, and S. Herbrechtsmeier, "Occupancy Grid Mapping with Highly Uncertain Range Sensors based on Inverse Particle Filters," 2016.

Paper 9

Multi-Modal Semantic Segmentation in 3D with Range Images
Mikkel Fly Kragh, Martin Sand, Henrik Karstoft

Draft, February 2018

Multi-Modal Semantic Segmentation in 3D with Range Images

Mikkel Kragh, Martin Sand, Henrik Karstoft

Department of Engineering, Aarhus University
mkha@eng.au.dk

March 10, 2018

Abstract

In recent years, convolutional neural networks have shown great advances on different image recognition tasks. The networks allow for large model capacities, and the combination with continually growing datasets has led to remarkable improvements in classification performance. Extensions to 3D data formats such as point clouds from a lidar sensor have shown similar trends, in which self-learned features outperform hand-crafted features on most detection and classification tasks. However, whereas annotated 2D image datasets are widely available, only few datasets exist publicly with annotated 3D point clouds. In this paper, a semi-automated annotation process is proposed to acquire a large-scale point-wise labeled dataset. 3D point clouds from a Velodyne lidar are converted to 2D range images, and a state-of-the-art fully convolutional neural network is used for semantic segmentation. 3D points are further projected onto color and thermal images, allowing multi-modal fusion across three sensing modalities commonly used for obstacle detection. Preliminary results show that the classification performance is affected by class imbalance and annotation errors caused by label misalignment. However, the results further show that a custom loss function is partially able to mitigate the label misalignment, when class-specific costs are introduced during training. To date, fusion with color and thermal images has not improved classification performance, possibly due to calibration inaccuracies. Future work will therefore focus on sensor calibration and synchronization, as well as ground truth data refinement.

Keywords — deep learning, semantic segmentation, multi-modal, range images, lidar, color camera, thermal camera

I. Introduction

Within the past decade, deep learning has shown great advances on object recognition and semantic segmentation on 2D images. Based on large annotated datasets, convolutional neural networks (CNNs) automatically learn hierarchical feature representations specifically designed for classification and segmentation tasks. An extension to 3D is straightforward, as 2D convolutions and pooling operations can be replaced by 3D equivalents. However, the following increase in the number of parameters and memory consumption forces such networks to have much smaller input dimensions in order to run on even the best modern graphic cards. It further forces the 3D data to be grid-based like images. The Voxnet network (Maturana and Scherer, 2015) has succesfully applied a 3D CNN on the ModelNet dataset (Wu et al., 2015), however with an input resolution of only 32^3 voxels. The Octnet network (Riegler et al., 2017) has since then utilized an octree data structure to avoid

unnecessary computations and memory use of empty voxels. Effectively, this has increases the possible input dimensions to 256^3 on the same dataset.

Irregular sampled 3D data such as point clouds from a lidar can be voxelized to fit into a 3D grid. This has shown to work well with 3D CNNs on small segments (Maturana and Scherer, 2015), but requires an efficient region extraction mechanism that only forwards relevant segments to the CNN for classification. However, voxelization does not work well with sparse data such as Velodyne point clouds, since the point density decreases with distance and thus requires either very large voxels to work across the entire cloud or results in an increasing number of empty voxels with distance. Therefore, other data representations have been proposed for sparse point clouds that specifically address point neighborhoods without voxelization. Pointnet (Qi et al., 2017) treats a point cloud like an unordered set of points and proposes a network architecture with permutation-invariant operations. The network achieves object classification results on par with state-of-the-art voxelized networks, and can be further extended to semantic segmentation by concatenating global features with point-wise features. Another data representation exploits the data acquisition principle of a rotating lidar to generate 2D range images. A range image is a 2D grid-based representation of range measurements based on a projection of 3D points to a cylinder plane. It was first used in the context of deep learning for unsupervised feature learning (De Deuge et al., 2013). Since then, it has been used as input for fully convolutional networks (FCNs) (Li et al., 2016; Wu et al., 2017) to detect 3D bounding boxes of vehicles on the public KITTI dataset (Geiger et al., 2013). On the same dataset, Chen et al. (2017) have proposed the Multi-View 3D (MV3D) network, combining range images with bird's-eye view 2D grid maps, thus providing the network with multiple data representations of the point clouds. MV3D further fuses the two lidar views with an RGB image by projecting 3D region proposals to the 2D image followed by concatenation of features extracted from similar regions in the three views (range image, bird's-eye view, and RGB image).

Unlike 2D image recognition and semantic segmentation where multiple large-scale annotated datasets are available publicly (Deng et al., 2009; Mottaghi et al., 2014), only a few annotated public datasets with 3D points clouds exist. The semantic3d.net dataset (Hackel et al., 2017) provides large-scale manually annotated point-wise labels of various urban and suburban environments. The dataset is acquired using static surveying-grade laser scanners and thus provides extremely reliable range measurements. However, the point clouds are not easily converted to e.g. range images, and the dataset does not represent realistic point clouds acquired with moving multi-beam lidars. Therefore, most research on deep learning for point clouds acquired with autonomous vehicles use annotated bounding boxes of vehicles and pedestrians from the KITTI dataset. Recently, Wu et al. (2017) built a Velodyne lidar simulator into the video game Grand Theft Auto V to easily obtain more training data. Similarly, the open-source AirSim simulator (Shah et al., 2017) allows custom-built sensors such as a multi-beam lidar to be simulated in physically and visually realistic environments. However, even the most modern computer graphic engines still use relatively simple geometric models for objects such as cylinders for pedestrians (Wu et al., 2017).

Therefore, in this paper, a semi-automated annotation process is proposed to acquire a large-scale dataset with point-wise labels for 3D point clouds. The publicly available FieldSAFE dataset (Kragh et al., 2017) was used for this purpose, as it provides a GPS-referenced ground truth map of the environment with pixel-wise labels. By georeferencing 3D point clouds from consecutive lidar frames, ground truth labels are appended from the annotated map, thus providing a large-scale annotated dataset. The lidar point clouds are converted to 2D range images, and a state-of-the-art fully convolutional neural network is applied for semantic segmentation. As the FieldSAFE

dataset further includes a stereo and a thermal camera, the 3D points are projected onto color and thermal images. This provides additional range image channels and allows for comparison of multi-modal fusion across lidar, color camera, and thermal camera.

The main contributions of the paper are threefold:

- Multi-modal fusion of lidar, color camera, and thermal camera using a deep neural network for semantic segmentation on range images.
- Semi-automated annotation process utilizing the projection of georeferenced point clouds onto a manually labeled orthophoto.
- Evaluation of multi-modal semantic segmentation on a publicly available obstacle detection dataset in agriculture.

The paper is divided into 7 sections. Section 2 presents the dataset in both point cloud and range image format. Section 3 presents the network architecture for a deep convolutional neural network performing semantic segmentation on range images. Section 4 presents the training strategy and how to deal with class imbalance. Section 5 presents experimental results followed by a discussion in section 6. Ultimately, section 7 presents a conclusion and future work.

II. Dataset

For training and testing our method, we use the publicly available FieldSAFE dataset (Kragh et al., 2017) for multi-modal obstacle detection in agricultural fields. The dataset includes approximately two hours of recordings from a grass field in Denmark in October 2016. Both static and dynamic obstacles were present during the recordings. However, in this paper, we only use the section of the dataset containing static obstacles. Figure 1 illustrates example obstacles from the dataset.

The dataset includes calibrated and synchronized data from a Velodyne HDL-32E lidar, a Multisense S21 stereo camera, and a Flir A65 thermal camera.

Annotations, however, are not available for each sensor in its local sensor frame. Instead, an annotated, drone-recorded orthophoto relates global (geographic) coordinates to class labels. In order to acquire point-wise labels for all 3D point clouds, we therefore have to transform local sensor data into global coordinates and look-up class labels from the annotated ground truth map.

i. Ground Truth

Using the localization data of the tractor provided by the FieldSAFE dataset, all point clouds are georeferenced. That is, 3D points are transformed from the lidar frame to the world frame in UTM coordinates. Figure 2a shows global coordinates colored by their height above sea level. The point cloud is an accumulation of all frames during a single traversal along the edge of the field. As the grass field has a general slope in one direction, the coloring does not directly correspond to the height above ground level. Figure 2b shows the lidar reflectance/intensity which is a measure of the intensity of the reflected laser beams. It depends on the traveled distance of the light, the incident angle, and the material of the reflection. Figure 2c and 2d show the RGB colors and temperatures projected from the stereo camera and the thermal camera, respectively. As the two cameras only cover part of the field of view (FOV) of the lidar, these two point clouds have considerably fewer points than 2a and 2b.

Point-wise ground truth annotations are obtained by looking up class labels in the annotated ground truth map. This is accomplished using a nonreflective similarity transformation, converting UTM x- and y-coordinates to pixel coordinates:

$$H = \begin{bmatrix} -18.850 & -46.285 & 296310021.065 \\ 46.285 & -18.850 & 95754362.017 \\ 0.000 & 0.000 & 1.000 \end{bmatrix}. \quad (1)$$

Figure 1: *Examples of static obstacles. Adapted from Kragh et al. (2017) with permission.*

Figure 2e shows the resulting combined georeferenced point cloud colored by class labels projected from the ground truth map. Black pixels denote undefined areas that included moving objects during recording. Since the annotated drone orthophoto is static, these movements were not recorded in the ground truth map and should thus be ignored during training.

As the FieldSAFE dataset only includes a single field, the data must be split in order to obtain distinct subsets for training, validation, and test. However, no possible splits can avoid potential issues concerning inter-subset correlation and class imbalance. As a compromise, the dataset is therefore split geographically such that all classes are represented in all subsets, and such that inter-subset correlation is minimal. Figure 2f illustrates the splits into a training set (green), a validation set (blue), and a test set (red).

ii. Range Images

As described above, we apply a range image format to the individual point clouds as in Li et al. (2016), thus making point neighborhoods well-defined when processed by a subsequent 2D convolutional neural network.

The Velodyne HDL-32E lidar has 32 laser beams, each rotating 360° with 10 Hz. Each scan (360° rotation) can therefore be thought of as a range image. That is, a polar sampled cylinder image with range intensities. The (x, y, z) Cartesian coordinates are converted to polar (r, θ, ϕ) coordinates using:

$$r = \sqrt{x^2 + y^2 + z^2} \tag{2}$$

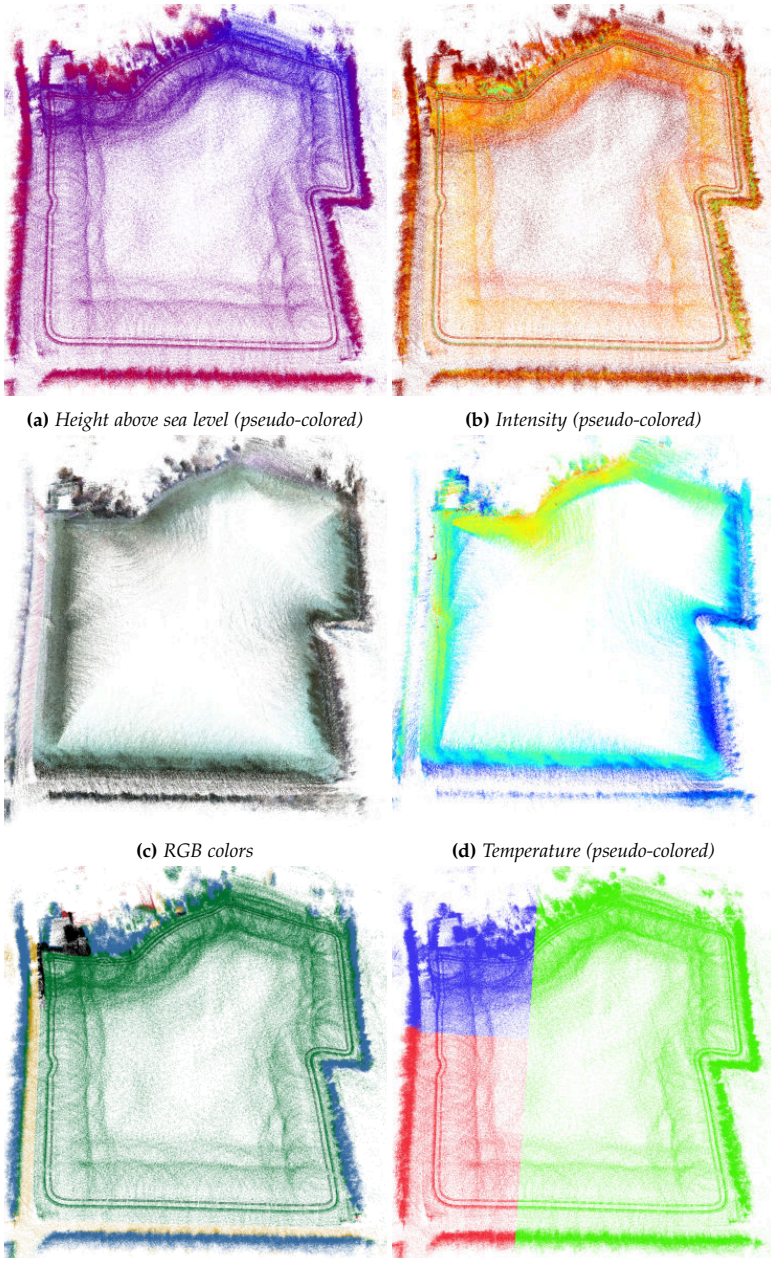
$$\theta = \text{atan2}\,(y, x) \tag{3}$$

$$\phi = \text{atan2}\left(z, \sqrt{x^2 + y^2}\right) \tag{4}$$

The range r defines the intensities of the range image, whereas the azimuth θ and elevation ϕ angle spans define the width and height, respectively.

With a horizontal FOV of 360° and an angular resolution of approximately 0.165°, the image width is 2176 pixels. The vertical FOV, however, is only 41.34° with a much smaller angular resolution of 1.33°. A vertical upsampling to the same resolution as the horizontal one of 0.165° results in an image height of 250 pixels. Since this is relatively close to $2^8 = 256$, which is a convenient size for deep learning algorithms, we allow a small deviation in the horizontal and vertical resolutions and thus force an image height of 256 pixels. Combined, this results in range images of size 2176×256 pixels. To avoid undefined pixels, a nearest neighbor interpolation is applied both horizontally and vertically, thus assigning each pixel by its nearest projected range value.

Figure 3 illustrates an example of a range image and its pixel-wise ground truth labels. Figure 3a shows the label image. Black pixels denote undefined areas, either due to the black, undefined regions in Figure 2e or due to a static mask which is applied on all labeled range images. The mask covers all regions in the range images where the tractor and sensor platform are visible. By regarding these as un-



(a) *Height above sea level (pseudo-colored)*

(b) *Intensity (pseudo-colored)*

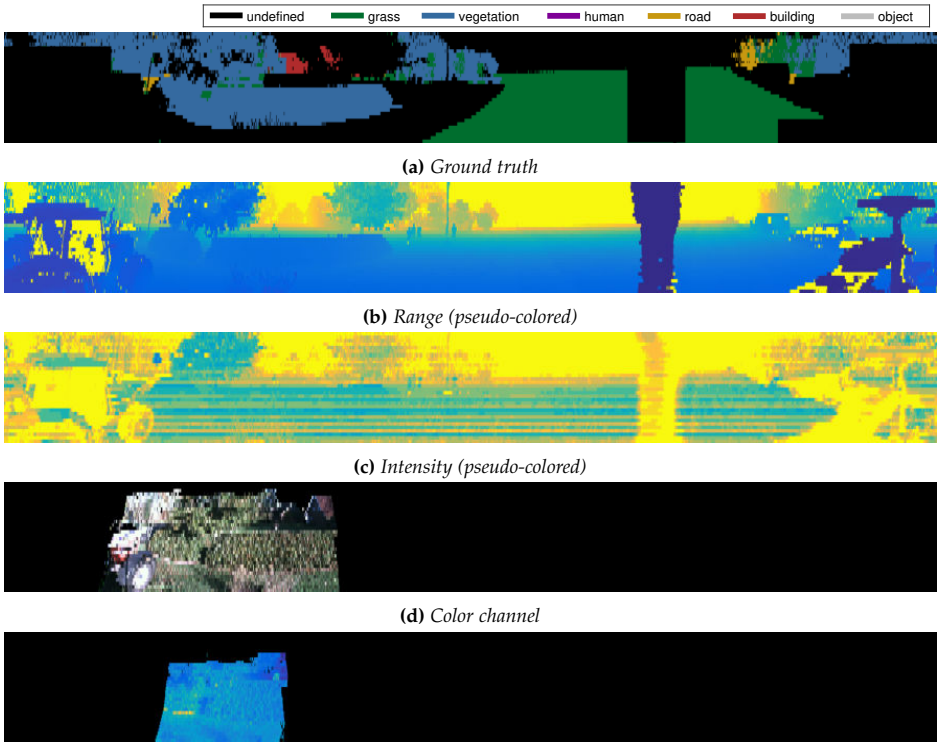
(c) *RGB colors*

(d) *Temperature (pseudo-colored)*

(e) *Ground truth*

(f) *Assigned datasets: training (green), validation (blue), and test (red)*

Figure 2: *Georeferenced points colored by different channels. For technical reasons, only 10% of all points are shown.*



(a) *Ground truth*

(b) *Range (pseudo-colored)*

(c) *Intensity (pseudo-colored)*

(d) *Color channel*

(e) *Temperature channel (pseudo-colored)*

Figure 3: *Channel examples represented in range image format.*

defined in the ground truth range image, the network effectively ignores the regions during training. Figure 3b shows the range channel generated from the lidar range measurements. The channel is pseudo-colored with increasing range from blue to yellow. The range channel is represented as unsigned 16-bit integers scaled linearly in the range interval 0 m to 100 m. Regions with no laser returns (e.g. when pointing towards the sky) are assigned maximum distance. Figure 3c shows the reflectance intensities from the laser measurements. The intensities are represented as unsigned 8-bit integers directly from the lidar. Figure 3d shows the RGB color channels from the stereo camera, with each channel represented as unsigned 8-bit integers. The stereo camera image was converted to range image format by projecting lidar points onto the image utilizing the known static transformation between the sensors and the intrinsics of the camera. Figure 3e shows the temperature channel from the thermal camera, generated using the same procedure as for the stereo camera. The thermal camera provides absolute temperatures as unsigned 16-bit integers with a resolution of 0.04 °C and a temperature range from −273 °C to 2348 °C. For outdoor obstacle detection, however, such extreme temperatures are unlikely to appear. Therefore, the temperature channel is represented as unsigned 16-bit integers scaled linearly in the temperature interval 0 °C to 50 °C.

III. Network Architecture

For classifying each point in the point clouds, we apply a state-of-the-art CNN for semantic segmentation by Jégou et al. (2017) on the range images. The network is a fully convolutional extension of the DenseNet (Huang et al., 2016) architecture including an upsampling path to match the image input dimensions. Using skip-connections between the encoding and decoding paths, the network utilizes a hierarchy of feature maps to perform accurate and smooth pixel-wise classifications (Long et al., 2015).

Figure 4 illustrates the network architecture. Similar to the U-Net architecture (Ronneberger et al., 2015), it consists of a downsampling path and an upsampling path. In the downsampling path, an initial 3×3 convolution is followed by dense blocks (DBs), concatenations, and transitions down (TDs) that in turn compute new features, increase the number of channels and reduce the spatial dimension. The concatenation links the DB outputs with their inputs, thus continuously forwarding low-level features to deeper layers. This effectively allows for feature reuse, as low-level features required by the final classifier do not need to be recomputed in each layer. In the upsampling path, transitions up (TUs), concatenations, and DBs in turn increase the spatial dimension by upsampling, reduce the number of channels, and combine low- and high-level features. This is followed by a 1×1 convolution and a softmax layer for classification. Unlike the downsampling path, however, the inputs and outputs of DBs are not concatenated. Effectively, this ensures that the number of feature channels (and parameters) are reduced.

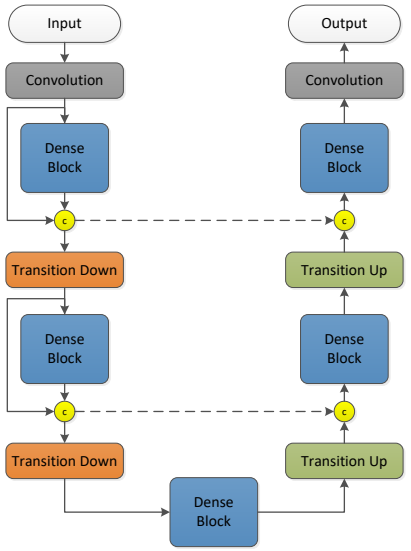


Figure 4: *Network architecture as proposed by Jégou et al. (2017). Concatenations are denoted by* ©.

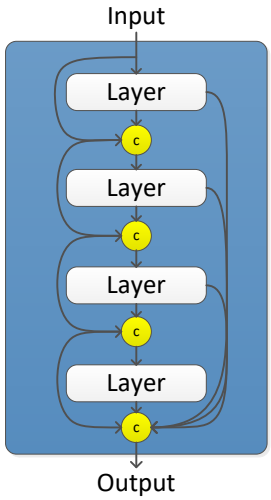


Figure 5: *Dense block consisting of $l = 4$ densely connected composite layers.*

A dense block consists of l densely connected composite layers, each outputting k feature maps. Figure 5 illustrates the concept with $l = 4$. Due to concatenations after each composite layer, the dense block outputs a combined feature map with $l * k$ channels. Each composite layer in a dense block consists of the following four operations: batch normalization (Ioffe and Szegedy, 2015), a rectified linear unit (ReLU) activation function, a 3×3 convolution, and a dropout layer (Srivastava et al., 2014).

A transition down internally consists of batch normalization, a ReLU, 1×1 convolution, dropout, and 2×2 maximum pooling. A transition up, on the other hand, simply consists of a 3×3 transposed convolution with stride 2 (Long et al., 2015).

IV. Training

The range images, as presented in the above section, form 6-channel image inputs to a deep neural network. The channels include range, intensity, red, green, blue, and temperature. Due to the unusual number of channels and their physical meaning, a pre-trained network is not publicly available. The network is therefore trained with randomly initialized weights instead of fine-tuning an existing model.

As explained above and shown in Figure 2f, the dataset is split geographically, allowing the same range image to appear in both the training, validation, and test sets. The pixel-wise labels, however, are masked by the subset, such that no labeled pixels appear in more than a single subset (training, validation, or test). Figure 6 illustrates this for a single frame.

To handle undefined pixels in the labeled range images, a custom weighted pixel-wise cross-entropy loss function is used:

$$H(p,q) = \sum_{x,y} \sum_{c} -p_c(x,y)\, w_c \log\left(q_c(x,y)\right) \tag{5}$$

Here, $p_c(x,y)$ denotes the ground truth probability for class c at pixel location (x,y), and $q_c(x,y)$ denotes the predicted probability after the softmax layer. The ground truth probability p is 1 for the correct class and 0 for all other classes. w_c is a weight that can make the back-propagated loss depend on the class label. For all undefined pixels, a weight of 0 is used, such that no loss is introduced for these pixels. In practice, the network thus ignores all predictions for these pixels during training. As is clear from Figure 2e, the dataset further suffers from class imbalance. This is handled by weighting class labels by their inverse relative frequencies, such that classes with rare appearances are given larger weights than classes that appear often. For instance, incorrectly classified vegetation gives a larger loss than incorrectly classified ground. Table 1 lists the number of label occurrences for each class along with relative label frequencies, inverse relative frequencies, and $\log_2$ of inverse relative frequencies.

Using the custom loss function, the network was trained over 50 epochs with the Adam optimizer (Kingma and Ba, 2014). The initial learning rate was set to 0.0001, and a weight decay factor of 1×10^{-4} and a dropout rate of 0.2 were used for regularization.

When training the network, square crops of size 256×256 pixels corresponding to the full

(a) *Ground truth*

(b) *Training*

(c) *Validation*

(d) *Test*

Figure 6: *Training, validation, and test splits from a single range image.*

Table 1: *Class weights based on appearances in the training set.*

	Grass	Vegetation	Human	Ground	Building	Object
Occurrences	3.86×10^8	6.40×10^7	7.38×10^4	7.15×10^6	5.37×10^5	1.76×10^5
Frequency	0.8428	0.1399	0.0002	0.0156	0.0012	0.0004
1/Frequency	1	7	6199	64	853	2597
$\log_2(1/\text{Frequency})$	0.25	2.84	12.60	6.00	9.74	11.34

height were used as input. As the range image covers a 360° FOV, the crops wrap-around at the left and right image boundaries. Reducing the input size from the original 2176×256 allowed the network to fit into a GeForce Titan X GPU with 12 GB RAM using a batch size of 8. Random crops were extracted in the horizontal range $[380, 660]$, ensuring non-zero color and temperature channels. Random horizontal flips provided further data augmentation. Table 3 lists all layers in the network, the output size of each layer, and the operations involved.

V. Experiments and Results

From a three lap traversal around the grass field, a total of 9,168 range images were generated. The labels were split pixel-wise into training, validation, and test sets as described above. By only using labeled images with more than 1000 defined pixels, this gave 7,837 frames for training, 4,092 frames for validation, and 3,359 frames for testing.

The network was trained using an implementation in Tensorflow[1]. The training took 21 hours in average to complete 50 epochs on a Gefore Titan X graphics card.

Different data formats were evaluated for the 6 range image channels. However, it was found that a simple conversion to floats and normalization to the range $[0, 1]$ gave the best performance. The three different class weighting strategies were further evaluated, with $\log_2$ of the inverse relative class frequencies providing the best results.

Table 2 lists the intersection over union (IoU) for each class as well as the mean over all classes as more range image channels are added. Clearly the network did not learn to distinguish all classes, as the IoU for *human* and *building* were close to 0. However, the network accomplished an IoU of 0.985, 0.905, and 0.545 for *grass*, *vegetation*, and *road* using the range and intensity channels. Adding more channels did generally not improve results. In fact, adding color and thermal channels decreased performance for *grass*, *vegetation*, and *road*, whereas *human*, *building*, and *object* were increased marginally.

Figure 7 shows an example of a predicted range image along with its ground truth labels. Figure 8 shows the same example as point clouds.

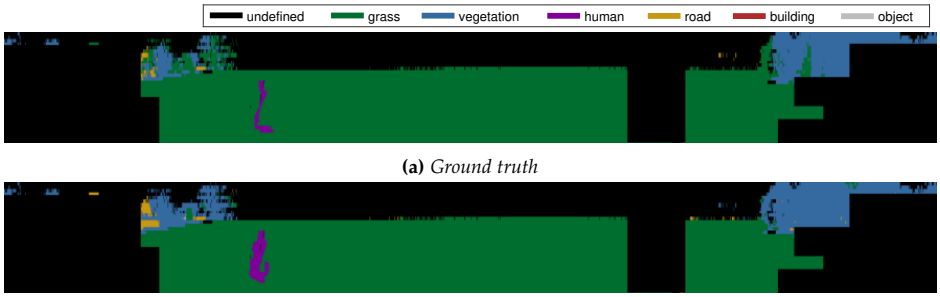
[1] https://github.com/titu1994/Fully-Connected-DenseNets-Semantic-Segmentation

VI. Discussion

The proposed method for multi-modal semantic segmentation using range images has provided preliminary results on a public agricultural obstacle detection dataset. The results showed that the network was able to distinguish classes with high occurrences such as *grass* and *vegetation*, whereas rarely occurring classes such as *human* and *building* experienced remarkably low classification scores.

Two possible explanations for the questionable classification performance are class imbalance and label errors. As shown in Table 1 above, the dataset suffered from severe class imbalance, which may have made it impossible for the network to efficiently distinguish rare classes. In addition, systematic label errors due to misalignment in the annotation process affected rare classes more than classes with high occurrences. Figure 8b illustrates this phenomenon with an adult mannequin doll, where only a small part of it was labeled correctly. The *grass* class, on the other hand, was only affected by label misalignment near the boundaries of the field. The ratio between falsely and correctly annotated points was thus smaller for *grass* and *vegetation* than for e.g. *human*.

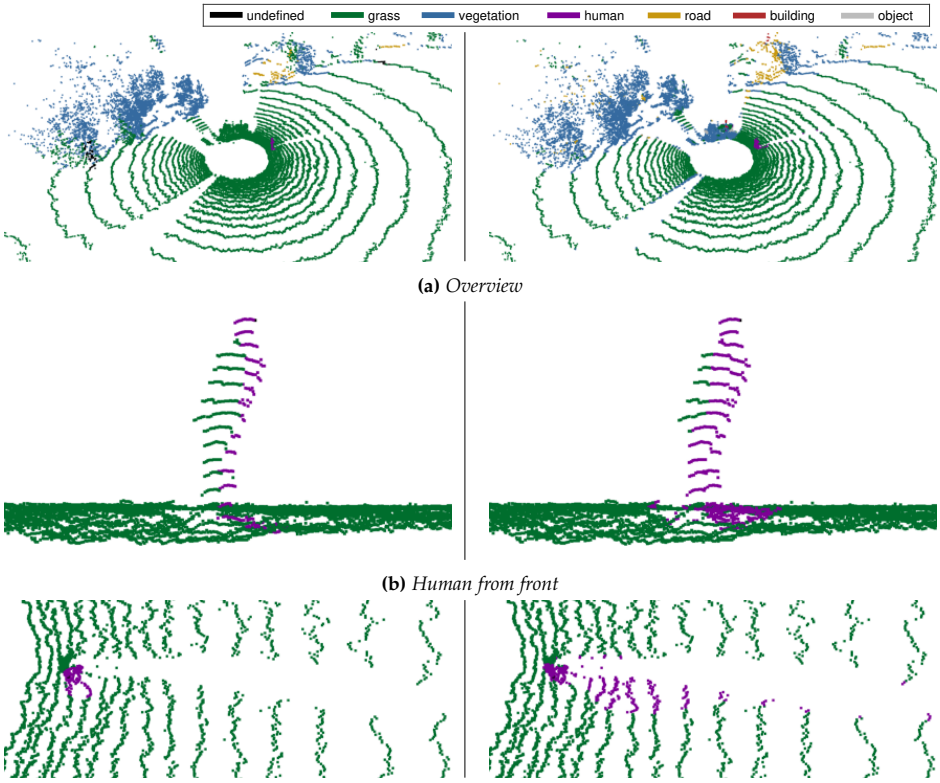
As seen in Figure 8b, the network was, to some degree, able to mitigate the problem of label misalignment for the *human* class. The custom loss function thus introduced a larger error when a true *human* point was misclassified than when a true *grass* point was misclassified. In practice, this allowed the network to "misclassify" true *grass* points, that in fact should have been annotated as *human* in the first place.



(a) *Ground truth*

(b) *Prediction*

Figure 7: *Example of range image prediction along with ground truth labels.*



(a) *Overview*

(b) *Human from front*

(c) *Human from above*

Figure 8: *Three views on a single frame from the test set. The left column shows ground truth annotations, whereas the right column shows predictions using a network trained on range and intensity channels, only.*

Table 2: *Class-wise classification results as more range image channels are added.*

	IoU							accuracy
	grass	vegetation	human	road	building	object	mean	
Range	0.979	0.888	0.000	0.246	0.000	0.014	0.355	0.975
Range, intensity	**0.985**	**0.905**	0.020	**0.545**	0.000	0.103	**0.426**	**0.981**
Range, intensity, color	0.977	0.878	0.000	0.202	0.000	0.009	0.344	0.972
Range, intensity, thermal	0.980	0.878	**0.023**	0.426	**0.005**	0.074	0.398	0.976
Range, intensity, color, thermal	0.976	0.879	0.000	0.140	**0.005**	**0.113**	0.352	0.972

The addition of more range image channels generally decreased classification performance rather than increasing it. This may be caused by multiple problems. As discussed above, only the part of the range image covering all modalities was used for training and testing. Although this prevented most zero-valued color and temperature pixels, some were still left as shown in Figure 3e. If the network did not learn to ignore zero-valued colors and temperatures, these could thus degrade sensor fusion performance. Moreover, an inaccurate calibration between the lidar and the two cameras would result in misaligned range image channels. From visual inspections, however, the modalities seemed to be aligned fairly well, although not perfectly. Another reason may be differences in object appearances between the training and test sets. If the network learned to recognize e.g. humans by the colors of their clothes in the training set, a worse performance would be seen on the test set if the colors changed. Since only four mannequin dolls were used as humans in the dataset, the issue of overfitting could arise. And since the mannequin dolls were not preheated to human-like temperatures, the thermal channel did not succeed in distinguishing them from other objects.

In future work, a new test with recordings from another field will be generated with manual point-wise annotations. This will allow for an accurate and realistic evaluation of how well points are predicted. It will further provide relevant results of how well the trained network generalizes to unseen environments with potential differences in grass height, vegetation geometry, and obstacle appearances.

VII. Conclusion

In this paper, an approach for multi-modal semantic segmentation was proposed using 2D range images. A rotating lidar was fused with a color camera and a thermal camera by projecting 3D lidar points onto the two image planes. A fully convolutional neural network was used to infer pixel-wise class labels of 6 classes. The network was trained on a publicly available agricultural obstacle detection dataset.

Preliminary results showed that the per-class classification performance was particularly affected by class imbalance, although different measures were taken to mitigate the problem. The results further showed that fusion with color and thermal images did not improve results. Possible causes for this include inter-modality misalignment, time synchronization issues, and localization inaccuracies for both the recording platform and the ground truth annotations. Therefore, future work will focus on sensor calibration and synchronization, as well as ground truth data refinement.

References

Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR*.

De Deuge, M., Quadros, A., Hung, C., and Douillard, B. (2013). Unsupervised feature learning for classification of outdoor 3d scans. In *Australasian Conference on Robotics and Automation*, volume 2, page 1.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li,

Layer	Output size	Operations
Input	$256 \times 256 \times 6$	
Convolution (CV1)	$256 \times 256 \times 48$	3×3 conv
Dense Block (DB1)	$256 \times 256 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Concatenation (C1)	$256 \times 256 \times 112$	CV1 + DB1
Transition Down (TD1)	$128 \times 128 \times 112$	$\text{BN} + \text{ReLU} + 1 \times 1\,\text{conv} + 2 \times 2\,\text{pool}$
Dense Block (DB2)	$128 \times 128 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Concatenation (C2)	$128 \times 128 \times 176$	TD1 + DB2
Transition Down (TD2)	$64 \times 64 \times 176$	$\text{BN} + \text{ReLU} + 1 \times 1\,\text{conv} + 2 \times 2\,\text{pool}$
Dense Block (DB3)	$64 \times 64 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Concatenation (C3)	$64 \times 64 \times 240$	TD2 + DB3
Transition Down (TD3)	$32 \times 32 \times 240$	$\text{BN} + \text{ReLU} + 1 \times 1\,\text{conv} + 2 \times 2\,\text{pool}$
Dense Block (DB4)	$32 \times 32 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Concatenation (C4)	$32 \times 32 \times 304$	TD3 + DB4
Transition Down (TD4)	$16 \times 16 \times 304$	$\text{BN} + \text{ReLU} + 1 \times 1\,\text{conv} + 2 \times 2\,\text{pool}$
Dense Block (DB5)	$16 \times 16 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Concatenation (C5)	$16 \times 16 \times 368$	TD4 + DB5
Transition Down (TD5)	$8 \times 8 \times 368$	$\text{BN} + \text{ReLU} + 1 \times 1\,\text{conv} + 2 \times 2\,\text{pool}$
Dense Block (DB6)	$8 \times 8 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Transition Up (TU1)	$16 \times 16 \times 64$	3×3 transposed conv, stride2
Concatenation (C6)	$16 \times 16 \times 432$	C5 + TU1
Dense Block (DB7)	$16 \times 16 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Transition Up (TU2)	$32 \times 32 \times 64$	3×3 transposed conv, stride2
Concatenation (C7)	$32 \times 32 \times 368$	C4 + TU2
Dense Block (DB8)	$32 \times 32 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Transition Up (TU3)	$64 \times 64 \times 64$	3×3 transposed conv, stride2
Concatenation (C8)	$64 \times 64 \times 304$	C3 + TU3
Dense Block (DB9)	$64 \times 64 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Transition Up (TU4)	$128 \times 128 \times 64$	3×3 transposed conv, stride2
Concatenation (C9)	$128 \times 128 \times 240$	C2 + TU4
Dense Block (DB10)	$128 \times 128 \times 64$	$[\text{BN} + \text{ReLU} + 3 \times 3\,\text{conv}] \times 4$
Transition Up (TU5)	$256 \times 256 \times 64$	3×3 transposed conv, stride2
Concatenation (C10)	$256 \times 256 \times 176$	C1 + TU5
Convolution	$256 \times 256 \times 7$	1×1 conv
Softmax		

Table 3: *Network architecture for processing 6-channel range images. Some operations are abbreviated: batch normalization (BN), rectified linear unit (ReLU), convolution (conv), and max pooling (pool).*

K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*.

Hackel, T., Savinov, N., Ladicky, L., Wegner, J. D., Schindler, K., and Pollefeys, M. (2017). SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume IV-1-W1, pages 91–98.

Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L. (2016). Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456.

Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., and Bengio, Y. (2017). The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1175–1183. IEEE.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kragh, M. F., Christiansen, P., Laursen, M. S., Larsen, M., Steen, K. A., Green, O., Karstoft, H., and JÃ¿rgensen, R. N. (2017). Fieldsafe: Dataset for obstacle detection in agriculture. *Sensors*, 17(11).

Li, B., Zhang, T., and Xia, T. (2016). Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440.

Maturana, D. and Scherer, S. (2015). Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE.

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., and Yuille, A. (2014). The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4.

Riegler, G., Ulusoy, A. O., and Geiger, A. (2017). Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 3.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

Shah, S., Dey, D., Lovett, C., and Kapoor, A. (2017). Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Wu, B., Wan, A., Yue, X., and Keutzer, K. (2017). Squeezeseg: Convolutional neural nets with

recurrent crf for real-time road-object segmentation from 3d lidar point cloud. *arXiv preprint arXiv:1710.07368*.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920.