Semantic Feedback for Collaborative Perception with Smart Edge Sensors

DISSERTATION
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von
Simon Alexander Bultmann
aus
Göttingen, Deutschland

Bonn, October 2023

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter: Prof. Dr. Sven Behnke
2. Gutachter: Prof. Dr. Hildegard Kühne

Tag der Promotion: 01.07.2024
Erscheinungsjahr: 2024

To Manon, Raphaël, and Antoine.

Abstract

In this thesis, we develop a system for accurate semantic perception of 3D scene geometry, persons, and objects in robotic applications. We consider the limitations of interpreting only a single sensor view with restricted measurement range, field of view, and resolution, and the challenges posed by centralized approaches that rely on high communication bandwidth and computational power.

To address these issues, we propose a network of distributed smart edge sensors equipped with a multi-modal sensor suite and an embedded CNN inference accelerator for on-device image processing. Real-time vision CNN models for person and object detection, semantic segmentation, and pose estimation are deployed on the sensors. The extracted information, such as 2D human keypoints, object poses, and semantic point clouds, is then passed to a central backend where multiple viewpoints are fused into a comprehensive 3D semantic scene model. Since image interpretation is computed locally, only semantic information is sent over the network. The raw images remain on the sensor boards, significantly reducing bandwidth requirements and mitigating privacy concerns for the observed persons.

The concept of *smart edge sensors* is further extended to mobile aerial and ground robots, enabling anticipatory human-aware navigation and active perception in areas not covered by stationary sensors. An outdoor smart edge sensor is presented, based on a UAV platform with on-board multi-modal semantic perception.

We introduce the concept of *semantic feedback*, enabling collaborative perception between sensor nodes and the central backend through bidirectional communication at the semantic level. The incorporation of global context information, such as the fused multi-view human and robot pose estimates, enhances the local semantic models of the smart edge sensors, improving pose estimation and enabling preemptive adjustments to the robot's navigation path, e.g. when a person emerges from an occluded area.

The proposed methods are evaluated using public datasets and real-world experiments in challenging cluttered and dynamic environments. The system demonstrates the ability to generate a real-time semantic scene model that includes semantically annotated 3D geometry, object instances, and poses of multiple persons.

Zusammenfassung

In dieser Arbeit wird ein System zur semantischen Wahrnehmung von 3D-Szenengeometrie, Personen und Objekten in der Robotik entwickelt. Dabei werden Einschränkungen berücksichtigt, die sich aus der Interpretation einer einzelnen Sensoransicht mit begrenztem Sichtfeld, begrenzter Messreichweite und Auflösung ergeben, sowie die Herausforderungen zentralisierter Ansätze, die eine hohe Kommunikationsbandbreite und Rechenleistung erfordern.

Um diese Einschränkungen zu beheben, wird ein Netzwerk verteilter Smart-Edge-Sensoren entwickelt, die mit einer multimodalen Sensorsuite und einem eingebetteten CNN-Inferenzbeschleuniger für die geräteinterne Bildverarbeitung ausgestattet sind. Auf den Sensoren werden Echtzeit-Vision-CNN-Modelle zur Personen- und Objekterkennung, semantischen Segmentierung und Posenschätzung eingesetzt. Die extrahierten Informationen, wie z. B. 2D-Keypoints von Personen, Objektposen und semantische Punktwolken, werden dann an ein zentrales Backend gesendet, wo mehrere Blickwinkel zu einem umfassenden semantischen 3D-Szenenmodell fusioniert werden. Da die Bildinterpretation lokal berechnet wird, werden nur semantische Informationen über das Netzwerk übertragen. Die Rohbilder verbleiben auf den Sensorboards, was die Bandbreitenanforderungen erheblich reduziert und Datenschutzbedenken für beobachtete Personen mindert.

Das Konzept der *Smart-Edge-Sensoren* wird auf mobile Luft- und Bodenroboter ausgeweitet und ermöglicht eine vorausschauende, menschenbewusste Navigation und aktive Wahrnehmung in Bereichen, die von stationären Sensoren nicht erfasst werden. Es wird ein outdoor Smart-Edge-Sensor vorgestellt, der eine UAV-Plattform mit integrierter multimodaler semantischer Wahrnehmung nutzt.

Für die kollaborative Wahrnehmung zwischen Smart-Edge-Sensorknoten und dem zentralen Backend durch bidirektionale Kommunikation auf semantischer Ebene wird das Konzept des *semantischen Feedbacks* entwickelt. Die Einbeziehung globaler Kontextinformation, z. B. der fusionierten Multi-View-Personen- und Roboterposenschätzung, verbessert die lokalen semantischen Modelle der Smart-Edge-Sensoren, indem die Posenschätzung unterstützt und eine vorausschauende Anpassung der Bewegung des Roboters, z. B. wenn eine Person aus einem verdeckten Bereich hervortritt, ermöglicht wird.

Die vorgeschlagenen Methoden werden anhand öffentlicher Datensätze und realer Experimente in anspruchsvollen, unübersichtlichen und dynamischen Umgebungen evaluiert. Das System ist in der Lage, in Echtzeit ein semantisches Szenenmodell zu erstellen, das semantisch annotierte 3D-Geometrie, Objektinstanzen und Posen mehrerer Personen umfasst.

Acknowledgments

I am profoundly grateful to my supervisor, Prof. Dr. Sven Behnke, for his guidance and support throughout my research work. I am deeply appreciative of his insightful feedback and inspiring research ideas that have significantly enriched the depth and quality of this thesis. I would also like to thank Prof. Dr. Hilde Kühne, who kindly agreed to serve as the second reviewer for this dissertation.

Moreover, I would like to thank my colleagues at the Autonomous Intelligent Systems Group, especially my co-authors Jan Quenzel and Dr. Raphael Memmesheimer, for the congenial and productive atmosphere, the mutual support, and the shared discussions. Their diverse perspectives and collective expertise have enriched my research and have been a valuable source of inspiration and motivation. I am also deeply appreciative of the students I have had the privilege of supervising for their contributions to my work. Furthermore, I would like to express my gratitude to Michael Schreiber for his invaluable assistance in building and maintaining the smart edge sensor network. His technical proficiency has ensured the robustness and reliability of the experimental framework of this thesis.

My deepest gratitude goes to my fiancée, Manon, for her unwavering encouragement and boundless love. Her presence has been a source of strength, and her steadfast belief in my abilities has sustained me throughout the challenges of this journey. To my family, I offer my heartfelt thanks for their enduring support.

This work was supported by the German Research Foundation (DFG) under grants No. BE 2556/16-2 and BE 2556/18-2 for the Research Unit FOR 2535 “Anticipating Human Behavior”, the German Federal Ministry of Education and Research (BMBF) under grant No. 13N14859 for the project “Kompetenzzentrum: Aufbau des Deutschen Rettungsrobotik-Zentrums (A-DRZ)”, Google through a faculty research award, and Fraunhofer IAIS.

Contents

List of Figures

List of Tables

Acronyms

2D	two-dimensional
3D	three-dimensional
6D	six-dimensional
AI	artificial intelligence
ADD	average 3D distance of model points
ADD-S	average 3D distance of model points for symmetric objects
AGC	automatic gain correction
AuC	area under the curve
BOP	benchmark for 6D object pose estimation
CAD	computer aided design
CNN	convolutional neural network
COCO	common objects in context
CPU	central processing unit
CSI	camera serial interface
CUDA	compute unified device architecture
DEM	digital elevation map
DLT	direct linear transform
DNN	deep neural network
DoF	degrees of freedom
FoV	field of view
FPS	frames per second
GPU	graphics processing unit
GTSAM	Georgia Tech smoothing and mapping
HOG	histograms of oriented gradients
HSR	human support robot
ICP	iterative closest point
iGPU	integrated graphics processing unit
IPHD	identity-preserved human detection
IoU	intersection over union
JDR	joint detection rate

LiDAR	light detection and ranging
LSTM	long short-term memory
mAP	mean average precision
mIoU	mean intersection over union
MIPI	mobile industry processor interface
ML	machine learning
MLP	multi-layer perceptron
MPJPE	mean per joint position error
MSPD	maximum symmetry-aware projection distance
MSSD	maximum symmetry-aware surface distance
NMS	non-maximum suppression
NTP	network time protocol
NUC	Next Unit of Computing
PAF	part affinity field
PC	personal computer
PCL	point cloud library
PCP	percentage of correct parts
PnP	perspective-n-point
PSM	pictorial structures model
RAM	random-access memory
RANSAC	random sample consensus
RGB	red green blue
R-CNN	region-based CNN
RGB-D	red green blue and depth
RMS	root mean square
RMSE	root mean square error
ROS	robot operation system
SIFT	scale-invariant feature transform
SLAM	simultaneous localization and mapping
SLIC	simple linear iterative clustering
SMPL	skinned multi-person linear
SMPL-X	SMPL-expressive
SoC	system-on-chip
SSD	single shot detector
SVD	singular value decomposition
TOPS	trillion operations per second
TPU	tensor processing unit
TSDF	truncated signed distance field
UAV	unmanned aerial vehicle
UGV	unmanned ground vehicle
USB	universal serial bus
VGA	video graphics array
VSD	visible surface discrepancy
YCB-V	Yale-CMU-Berkeley-video
YOLO	you only look once

1 Introduction

The field of robotics and artificial intelligence (AI) has seen tremendous progress in recent years, leading to the deployment of intelligent robots in a wide range of applications. One of the key challenges in enabling these robots to interact effectively and safely in real-world environments is achieving a comprehensive understanding of the surrounding scene. Semantic scene perception plays a crucial role in empowering robots to perform complex tasks such as safe and predictive collision-free navigation around people, object manipulation, and human-robot interaction. Building and updating a semantic scene model is a challenging task and requires accurate semantic perception of three-dimensional (3D) scene geometry, objects, robots, and people.

Traditionally, semantic scene understanding has been approached using single sensor views, which have inherent limitations such as a restricted field of perception, measurement range, and resolution. They are further prone to frequent occlusion in cluttered real-world scenes. Multiple consecutive viewpoints of a scanning trajectory are aggregated over time to build a comprehensive scene model, but only a single input view is available at a time. SemanticFusion (McCormac et al., 2017) uses the video stream of a single, moving RGB-D camera to build semantic maps of room-scale environments. A simultaneous localization and mapping (SLAM) system (Whelan et al., 2015) is employed to estimate the sensor poses during the scanning trajectory. MID-Fusion (B. Xu et al., 2019) generates an object-level dynamic volumetric map from a single RGB-D camera, estimating geometric, semantic, and motion properties for arbitrary objects in the scene. Kimera (Rosinol et al., 2021) is a modular metric-semantic stereo-inertial-SLAM framework that builds a hierarchical multi-level scene graph to represent an environment at different levels of abstraction such as buildings, rooms, places, and a detailed metric-semantic mesh. The scene model can further be extended to a dynamic scene graph that includes dynamic agents such as humans or robots. It may take about 30 minutes to create a building-scale semantic scene model, as only a single input view is processed at a time.

The constraints of single-view systems can be overcome by employing a collaborative approach using distributed sensor networks or multi-robot systems to enable faster coverage of large scenes. Kimera-multi (Tian et al., 2022) presents a system of multiple ground robots that collaboratively build a globally consistent metric-semantic scene model in a distributed manner, relying only on local peer-to-peer communication. When multiple robots meet, they perform inter-robot place recognition and relative pose estimation to fuse their local map and trajectory estimates. As communication is limited to sparse time points at rendezvous points between agents, the amount of information sharing and collaboration is limited in decentralized systems, and map redundancy between agents can arise. With CCM-SLAM, Schmuck and Chli (2019) propose a centralized approach for collaborative SLAM with multiple unmanned aerial vehicles (UAVs). The agents communicate relevant data, such as keyframes and map points to a central backend server, where observations are fused and a joint map is built. A bidirectional information flow between the agents and the server is implemented that allows the robots to augment their local maps with information from the global, joint map. Nevertheless, each agent keeps local autonomy based on their local map of limited size. The approach is extended to visual-inertial data (Karrer, Schmuck, and Chli, 2018) and to a generic backend that can handle agents with arbitrary local odometry systems (Patel et al., 2023). These systems build sparse feature-based maps but no dense semantic scene models.

Advances in edge computing have spurred interest in processing sensor data at the source, as the centralized fusion of raw data from multiple sensors requires high communication bandwidth and processing power on the central computer, leading to issues when scaling multi-view methods to a larger number of sensor nodes. Embedded inference accelerators, such as Google edge tensor processing units (TPUs) and Nvidia Jetson embedded graphics processing units (GPUs) have enabled deep-learning models, e.g. for object detection or human pose estimation, to process sensor data on mobile edge devices in real time in an energy-efficient manner. Efficient, light-weight convolutional neural network (CNN) architectures (Howard et al., 2019, 2017; Tan and Le, 2019) were developed for inference on embedded devices that can further be optimized using techniques such as quantization (Jacob et al., 2018) or reduced floating point accuracy. A network of smart cameras transmitting only abstract image features to a central processing station was proposed by Naikal, Lajevardi, and Sastry (2014) to build a system for human joint detection and action recognition and J. Zhang et al. (2020) investigate human pose estimation on smartphone SoCs. L. Zhang, Lixing Chen, and J. Xu (2021) propose to partition neural network inference for object detection between a mobile edge sensor and a backend server and to determine the partitioning point in an online manner, depending on the compute capability of the edge device and the network transmission speed. They further build an energy-efficient multi-camera system with onboard two-dimensional (2D) pose estimation for real-time 3D human pose estimation using adaptive camera selection to choose a subset of sensor nodes to process redundant views (L. Zhang and J. Xu, 2023).

Despite these recent advances in semantic scene understanding, collaborative perception, and edge computing, to the best of our knowledge, the comprehensive integration of stationary smart edge sensors and mobile robots coupled with a central backend has not been extensively explored in the context of collaborative 3D semantic scene perception. This thesis aims to bridge the gap by proposing a novel framework that effectively combines the capabilities of distributed stationary and mobile smart edge sensors and a central backend through bidirectional communication at the semantic level. To build an intelligent, interconnected, and efficient system for collaborative 3D semantic scene perception, we develop a network of distributed sensors with onboard compute capabilities for local semantic interpretation of the sensor data. We refer to the developed sensor platforms, which consist of multi-modal sensors (e.g. red green blue (RGB), red green blue and depth (RGB-D), and thermal cameras) and an embedded CNN inference accelerator for local image processing on the device, as *smart edge sensors*. The term emphasizes the ability of the sensor boards to semantically interpret data directly at its source, at the edge of the network. The smart edge sensor platforms used throughout this thesis are depicted in Fig. 1.1.

Powerful image processing CNN models, optimized for the inference accelerators, run in real time on the device in a time- and energy-efficient manner. They provide semantic percepts such as person and object detections, pixel- or point-wise semantic

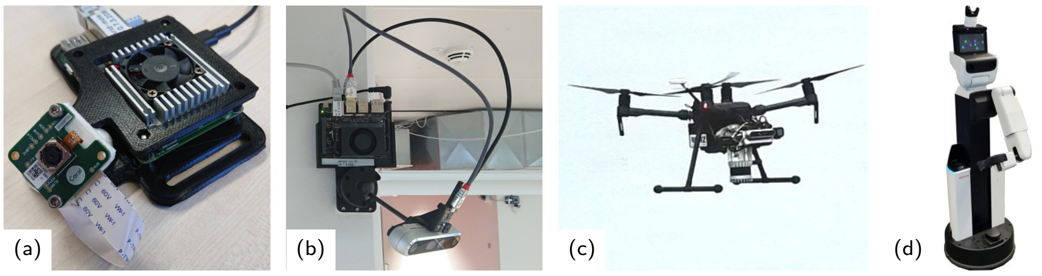

Figure 1.1: Smart edge sensor platforms used throughout this thesis: (a) stationary smart edge sensor with RGB camera and Edge TPU accelerator, (b) stationary smart edge sensor with RGB-D and thermal cameras and Jetson NX embedded GPU, (c) unmanned aerial vehicle (UAV) with 3D LiDAR, RGB-D and thermal cameras, iGPU and Edge TPU accelerators, (d) human support robot (HSR) with 2D LiDAR, RGB-D camera, and Jetson TX2 embedded GPU.

segmentation, and human, object, or robot poses. Since image interpretation is computed locally on the smart edge sensors, only semantic information is sent over the network. The raw images remain on the sensor boards, which significantly reduces the required bandwidth and mitigates privacy issues.

Using static viewpoints of distributed smart edge sensors and changing viewpoints of mobile robots, we develop methods for temporal multi-view fusion of semantic perceptions of individual sensors into an allocentric 3D semantic scene model.

In a first iteration, the scene model comprises dynamic 3D human poses estimated in real time from the observations of the multiple perspectives of the sensor network. Each sensor locally computes person detections and 2D heatmaps of human joint keypoint locations and their uncertainties that are then fused into allocentric 3D skeleton models on a central backend. The movement of multiple persons through the lab-scale capture space is tracked in real time using up to 16 smart edge sensors with RGB cameras. As the multi-view fusion of local joint detections to 3D pose estimates requires the relative camera poses to be known, we develop methods for online marker-free extrinsic camera calibration requiring only a rough, tape-measure-based initialization.

We then extend the smart edge sensor network with additional sensor nodes with enhanced computational capabilities and RGB-D and thermal cameras to build a complete 3D semantic scene model that includes semantically annotated 3D scene geometry as a volumetric map with additional object-level pose and shape information for specific object classes in addition to the dynamic 3D human poses.

The distributed sensors are coupled with a central backend through a feedback loop for bidirectional communication at the semantic level. The concept of *semantic feedback* is a key element of the presented research work. Relevant parts of the allocentric semantic model, which fuses all available sensor perspectives, are reprojected into the local views of individual sensors and sent back to them. This allows the local semantic models on each sensor to be improved and made more robust by incorporating global context information, thereby implementing collaborative information sharing among distributed sensors. For example, occluded or ambiguous joint detections in a local sensor view can be resolved by fusing the local detections with the reprojected allocentric multi-view human pose estimate received as semantic feedback. The high-level data processing architecture of the approaches developed in this thesis is illustrated in Fig. 1.2.

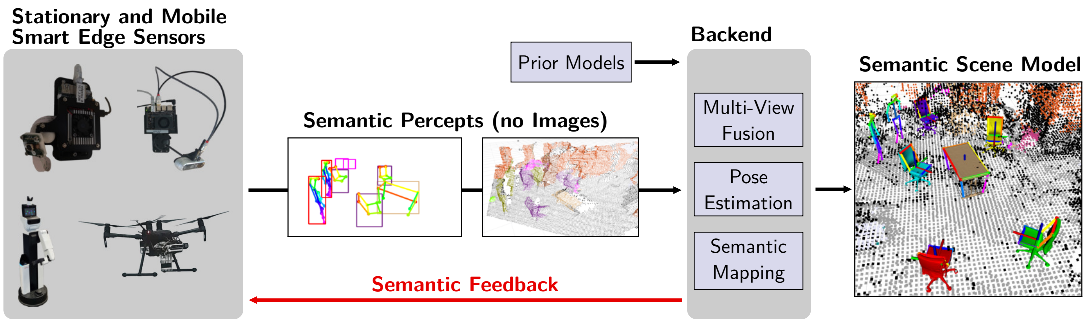

Figure 1.2: Data processing architecture of the approaches developed in this thesis: Stationary and mobile smart edge sensors interpret images locally and send semantic percepts such as person and object detections, pose estimates, and semantic point clouds to a central backend. The backend fuses semantic percepts from multiple smart edge sensors with prior information such as skeleton models or building plans into an allocentric 3D semantic scene model. Parts of the scene model are sent back to individual sensors as semantic feedback to improve local view interpretation.

We extend the concept of *smart edge sensors* with local on-device semantic image processing from static sensor boards to mobile aerial and ground robots. We propose a UAV platform with onboard real-time multi-modal semantic perception as a mobile smart edge sensor operating outdoors. Here we use a single, but moving, sensor node, whereas in the previous scenarios, we worked with multiple, static smart edge sensors. The semantic information from point cloud, RGB, and thermal modalities is fused into a joint image segmentation mask and a semantically labeled 3D point cloud in a Bayesian manner. In addition, we investigate the use of label propagation from an aggregated image-based semantic map to the LiDAR modality to overcome the problems of domain adaptation when generalizing between different sensor types. Consequently, the accuracy of the point cloud semantic segmentation is significantly improved.

Finally, we combine the previous ideas by incorporating a mobile service robot into the network of static smart edge sensors to collaboratively build a more complete semantic scene model. To fuse its semantic observations into the allocentric scene model, the robot pose is estimated from the external smart edge sensors and sent to the robot as semantic localization feedback. The robot extends the coverage and level of detail of the semantic map by actively perceiving the areas not observed by the static sensors. In addition, by incorporating semantic feedback of human pose observations from the external sensors, the robot can anticipate people emerging from behind occlusions and preemptively adjust its navigation path to maintain a safe distance.

The proposed methods are evaluated on various public datasets for single and multi-person multi-view human and object pose estimation. We further present a series of real-world experiments with the sensor network and the mobile service robot in a challenging, highly cluttered, and dynamic indoor lab environment, as well as with the UAV system in outdoor flights in an urban campus area and at a disaster test site. The proposed perception systems provide a complete semantic scene model including semantically annotated 3D geometry, 3D object instances with pose and shape information, and 3D poses of multiple persons estimated in real time and prove to be robust to real-world situations.

1.1 Key Contributions

The key contributions of this thesis are summarized as follows:

Smart Edge Sensor Network and Platforms. We develop two different sensor platforms comprising multi-modal cameras and embedded inference accelerators for local on-device image processing that we refer to as *smart edge sensors*. The first smart edge sensor platform is based on the Google Edge TPU Dev Board[1] and an RGB camera. The second smart edge sensor platform uses an Nvidia Jetson Xavier NX Developer Kit[2] and RGB-D and thermal cameras. With up to 20 instances of these heterogeneous sensor boards, we build a distributed sensor network covering a lab-scale environment of $\sim 240\,\text{m}^2$. We further employ different mobile robots as sensor nodes, i.e. a UAV with RGB-D, thermal, and LiDAR sensors and onboard processing using an Edge TPU or an iGPU as inference accelerator, and a human support robot (HSR) with an RGB-D camera and onboard processing using a Jetson TX2 accelerator.

For real-time processing on board these embedded inference accelerators (Edge TPU, Jetson NX and TX2, iGPU), we adapt efficient CNN architectures for image and point cloud semantic segmentation, person and object detection, and human, robot, and object pose estimation, and retrain or fine-tune them on task-specific datasets.

Collaborative Semantic Perception. We develop methods for temporal multi-view fusion of semantic percepts from individual sensors into an allocentric 3D semantic scene model, using static viewpoints of distributed smart edge sensors and changing viewpoints of mobile robots. The scene model thereby contains dynamic 3D human poses estimated in real time and semantically annotated 3D scene geometry as a volumetric map with additional object-level information. To estimate the poses of the different static and moving sensors contributing to the allocentric scene model, we develop methods for online marker-free extrinsic camera calibration and mobile robot pose estimation using multi-view keypoint detections.

Semantic Feedback. The proposed architecture for collaborative semantic perception divides the computation between smart edge sensors performing image analysis locally for each camera view and a central backend fusing the semantic interpretations of individual views. In this context, we propose *semantic feedback*, a scheme that enables bidirectional communication on a semantic level between distributed sensors and the central backend. Relevant parts of the allocentric semantic scene model are sent back to the individual sensors. This allows them to incorporate global context information, e.g. the fused multi-view human pose estimate or occlusion information for human joints computed via ray-tracing through the estimated 3D scene geometry, into their local semantic models, thus improving local view interpretation. Furthermore, mobile robots can anticipatorily adjust their navigation path to people emerging from behind occlusions through semantic feedback of human pose observations from external sensors.

1 `https://coral.ai/docs/dev-board/datasheet`, accessed: 2023-08-01
2 `https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit`, accessed: 2023-08-01

MULTI-MODALITY FUSION AND CROSS-DOMAIN ADAPTATION USING LABEL-PROPAGATION. Chapter 4 presents an approach for the Bayesian fusion of semantic information from point cloud, RGB, and thermal modalities into a joint image segmentation mask and a semantically labeled 3D point cloud. We further investigate using label propagation from image to LiDAR modalities to overcome domain adaptation issues of LiDAR segmentation when generalizing between different sensor types, thereby significantly improving the accuracy of the point cloud segmentation.

1.2 PUBLICATIONS

Parts of this thesis have been published in peer-reviewed conference proceedings and journals. The most relevant publications covering the main chapters of this thesis are listed below in chronological order:

> Simon Bultmann and Sven Behnke (2021). "Real-time multi-view 3D human pose estimation using semantic feedback to smart edge sensors." In: *Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2021.XVII.040

> Simon Bultmann, Jan Quenzel, and Sven Behnke (2021). "Real-time multi-modal semantic fusion on unmanned aerial vehicles." In: *European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR50962.2021.9568812

> Simon Bultmann and Sven Behnke (2022). "3D semantic scene perception using distributed smart edge sensors." In: *International Conference on Intelligent Autonomous Systems (IAS)*, pp. 313–329. DOI: 10.1007/978-3-031-22216-0_22

> Simon Bultmann, Jan Quenzel, and Sven Behnke (2023). "Real-time multi-modal semantic fusion on unmanned aerial vehicles with label propagation for cross-domain adaptation." In: *Robotics and Autonomous Systems* 159, p. 104286. DOI: 10.1016/j.robot.2022.104286

> Simon Bultmann, Raphael Memmesheimer, and Sven Behnke (2023). "External camera-based mobile robot pose estimation for collaborative perception with smart edge sensors." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8194–8200. DOI: 10.1109/ICRA48891.2023.10160892

The following publications (organized in chronological order) were written in close collaboration with bachelor and master students under my supervision. Sections of these publications are also presented in this thesis.

> Moritz Zappel, Simon Bultmann, and Sven Behnke (2021). "6D object pose estimation using keypoints and part affinity fields." In: *RoboCup International Symposium*, pp. 78–90. DOI: 10.1007/978-3-030-98682-7_7

> Bastian Pätzold, Simon Bultmann, and Sven Behnke (2022). "Online marker-free extrinsic camera calibration using person keypoint detections." In: *44th DAGM German Conference on Pattern Recognition (GCPR)*, pp. 300–316. DOI: 10.1007/978-3-031-16788-1_19

> Julian Hau, Simon Bultmann, and Sven Behnke (2022). "Object-level 3D semantic mapping using a network of smart edge sensors." In: *6th IEEE International Conference on Robotic Computing (IRC)*, pp. 198–206. DOI: 10.1109/IRC55401.2022.00041

The following publications, listed in chronological order, are related to the topics presented in this thesis and were written with my contribution as a co-author during the time the presented research was conducted. They are cited as external literature in this thesis and do not cover significant parts of the chapters:

> Marius Beul, Simon Bultmann, Andre Rochow, Radu Alexandru Rosu, Daniel Schleich, Malte Splietker, and Sven Behnke (2020). "Visually guided balloon popping with an autonomous MAV at MBZIRC 2020." In: *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 34–41. DOI: 10.1109/SSRR50563.2020.9292612

> Marius Beul, Max Schwarz, Jan Quenzel, Malte Splietker, Simon Bultmann, Daniel Schleich, Andre Rochow, Dmytro Pavlichenko, Radu Rosu, Patrick Lowin, Bruno Scheider, Michael Schreiber, Finn Süberkrüb, and Sven Behnke (2022). "Target chase, wall building, and fire fighting: Autonomous UAVs of team NimbRo at MBZIRC 2020." In: *Field Robotics* 2, pp. 807–842. DOI: 10.55417/fr.2022027

1.3 Open Source Software Releases

To facilitate future research on collaborative perception using smart edge sensor networks, implementations for some of the developed methods are publicly available, including:

- The source code of the backend processing for multi-view 3D human pose estimation using semantic feedback, introduced by Bultmann and Behnke (2021)[3].
- The source code for 3D semantic scene perception using distributed smart edge sensors, introduced by Bultmann and Behnke (2022), for both onboard processing on the sensor boards[4] and multi-view fusion, semantic mapping, and semantic feedback on the central backend[5].

1.4 Outline

This thesis comprises six chapters, of which Chapters 2 to 5 present the scientific contributions (cf. Sec. 1.1). The approaches presented usually build on the content of the preceding chapters. The chapters are written to be self-contained so that they can be read individually.

Chapter 2 presents a method for 3D multi-person pose estimation from a multi-camera setup and introduces the network of distributed smart edge sensors that is used and extended throughout the thesis. The distributed sensors are coupled with a central backend through a semantic feedback loop that enables the local semantic models on each sensor to incorporate global context information.

3 https://github.com/AIS-Bonn/SmartEdgeSensor3DHumanPose
4 https://github.com/AIS-Bonn/JetsonTRTPerception
5 https://github.com/AIS-Bonn/SmartEdgeSensor3DScenePerception

In Chapter 3, the smart edge sensor network is extended for 3D semantic scene perception. New sensor nodes with RGB-D perception are added and a volumetric semantic map of the scene is created. Certain furniture objects further are explicitly represented in the scene model via object-centric sub-maps and their movement is tracked through the scene. The semantic feedback for human pose estimation is extended with occlusion information for each joint using the 3D scene geometry from the semantic map and a ray-tracing approach.

Chapter 4 introduces a UAV with onboard real-time multi-modal semantic perception as a mobile smart edge sensor node. Whereas in previous chapters, we worked with multiple, static smart edge sensors, here we employ a single but moving sensor node. Label propagation from an image-based semantic map for the semantic segmentation of individual LiDAR scans allows for sensor-specific adaptation of the LiDAR semantic segmentation CNN with cross-modality and cross-domain supervision. This brings back the idea of semantic feedback: Parts of the allocentric semantic map are reprojected into individual sensor views and used as a reliable source of semantic information.

Chapter 5 combines the previous approaches to the final extent of the proposed system for collaborative semantic scene perception: A mobile service robot is employed together with the network of static smart edge sensors to collaboratively build a more complete semantic map. The robot's pose is estimated from the external smart edge sensors such that its semantic observations can be fused into the allocentric semantic scene model. The robot actively perceives the areas not observed by the static sensors and extends the coverage and level of detail in the semantic map. In addition, the robot can anticipate people emerging from behind occlusions and anticipatorily adjust its navigation path to maintain a safe distance through semantic feedback of human pose observations from the external sensors.

Finally, Chapter 6 concludes the thesis. The achieved scientific results and developed approaches are summarized and discussed, and possible future work directions and applications are suggested.

2 Multi-View 3D Human Pose Estimation

Preface

This chapter is adapted from Bultmann and Behnke (2021), previously published by the RSS Foundation and presented at Robotics: Science and System XVII (RSS 2021), and Pätzold, Bultmann, and Behnke (2022), previously published by Springer and presented at the 44th DAGM German Conference on Pattern Recognition (GCPR 2022).

Statement of Personal Contribution

The author of this thesis substantially contributed to all aspects of the publication (Bultmann and Behnke, 2021), including the literature survey, the conception, formalization, design, and implementation of the proposed methods, the preparation and conduct of experiments for the evaluation of the proposed approach, the analysis and interpretation of the experimental results, the preparation of the manuscript, as well as the revision and final editing of the version to be published.

The author of this thesis substantially contributed to the following aspects of the publication (Pätzold, Bultmann, and Behnke, 2022): the literature survey, the conception, formalization, and design of the proposed methods and experiments, the analysis and interpretation of the experimental results, and the preparation of the manuscript, as well as the revision and final editing of the version to be published. He further provided support for the implementation and evaluation as well as supervision over all other aspects.

The content presented in this chapter, unless otherwise stated, is the contribution of the author of this thesis.

Abstract

In this chapter, we present a novel method for the estimation of three-dimensional (3D) human poses from a multi-camera setup, employing distributed smart edge sensors coupled with a backend through a semantic feedback loop. Two-dimensional (2D) joint detection for each camera view is performed locally on a dedicated embedded inference accelerator. Only the semantic skeleton representation is transmitted over the network and raw images remain on the sensor board. 3D poses are recovered from 2D joints on a central backend, based on triangulation and a body model which incorporates prior knowledge of the human skeleton. A feedback channel from backend to individual sensors is implemented on a semantic level. The allocentric 3D pose is backprojected into the sensor views where it is fused with 2D joint detections. The local semantic model on each sensor can thus be improved by incorporating global context information. The whole pipeline thereby is capable of real-time operation at up to 30 Hz.

We further propose a novel, marker-free online method for the extrinsic calibration of multiple smart edge sensors, relying solely on the 2D human joint detections computed locally on the smart edge sensors and associated to person hypotheses on the central backend. We use these person hypotheses to repeatedly solve optimization problems in the form of factor graphs. Given suitable observations of one or multiple persons traversing the scene, the estimated camera poses converge towards a coherent extrinsic calibration within a few minutes.

We evaluate our method for 3D human pose estimation on three public datasets, where we achieve state-of-the-art results and show the benefits of our feedback architecture. Further evaluation is performed in our own setup for real world multi-person experiments. Using the feedback signal improves the 2D joint detections and in turn the estimated 3D poses. The marker-free extrinsic camera calibration based on human joint detections is evaluated in our real-world setup and we show that the calibration with our method achieves better results in terms of reprojection errors in the targeted application domain of 3D multi-person pose estimation, compared to a reference calibration generated by an offline method using a traditional calibration target.

2.1 Introduction

Accurate perception of humans is a challenging task with many applications in robotics and computer vision. It is a prerequisite e. g. for safe navigation and anticipatory movement of robots in the vicinity of people and can enable human-robot interaction or augmented reality scenarios. Utilizing a multi-camera setup for this task permits to cover a large workspace, increases robustness e. g. against occlusions and enables to lift 2D detections into 3D space. To successfully interpret and fuse the measurements of multiple sensors, they need to be transformed into a common coordinate frame which requires the sensor poses in a common reference frame—their extrinsic calibration. Camera calibration is typically achieved using offline methods that use checkerboard calibration targets. These methods, however, often are cumbersome and lengthy, considering that a new calibration is required each time any camera pose changes. Instead, we propose an online, marker-free approach for extrinsic camera calibration that uses person keypoint detections as calibration targets, allowing efficient calibration updates by just one or multiple people walking around the scene.

In this chapter, we address the task of 3D human pose estimation in allocentric world coordinates from a calibrated multi-camera setup together with the task of obtaining and updating the extrinsic camera calibration. Most state-of-the-art methods for 3D human pose estimation (Long Chen et al., 2020; J. Dong et al., 2019; Pavlakos et al., 2017b; Qiu et al., 2019; Remelli et al., 2020) follow a two-step approach: First, 2D keypoint detections are generated for each available view (cf. Fig. 2.1 bottom). Second, detections from multiple views are fused into a 3D human pose estimate and post-processed using a skeleton model (cf. Fig. 2.1 top-left). Many recent methods focus more on accuracy than efficiency and are difficult to employ in real-world scenarios with real-time constraints.

To enable online human pose estimation in real-world settings, we propose a novel architecture for real-time multi-view 3D human pose estimation using distributed smart edge sensors for the 2D pose estimation part. With the term *smart edge sensor*, we refer to our sensor platform comprising a camera and an embedded inference accelerator for local on-device image processing (cf. Fig. 2.1 top-right). This enables each camera view to

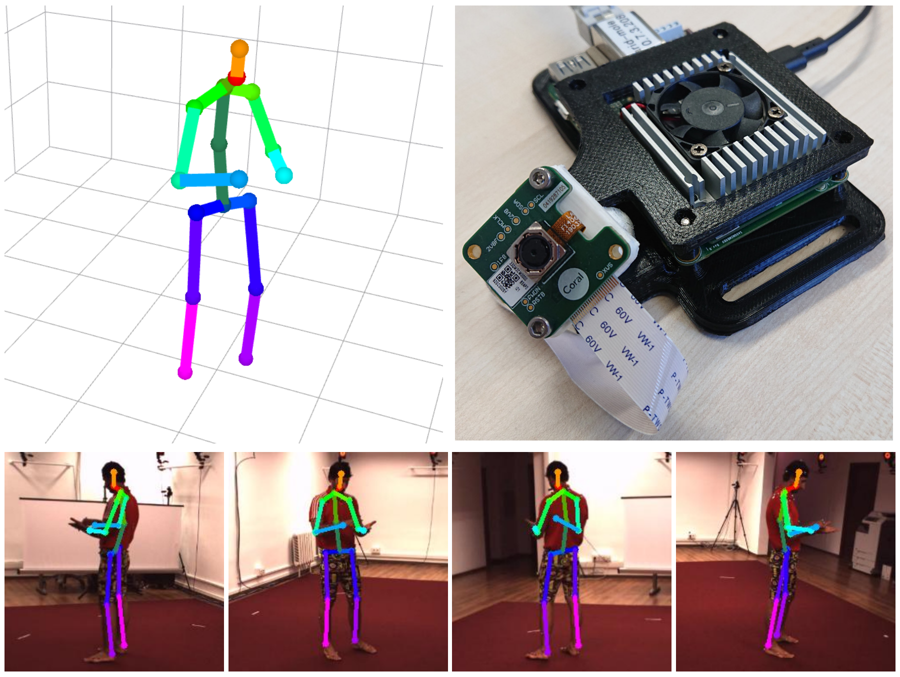

Figure 2.1: Multi-view 3D human pose estimation using smart edge sensors: Sensor board with attached camera (top-right). 2D pose detections from four views of the H3.6M dataset (Ionescu et al., 2014) (bottom). Estimated 3D human skeleton (top-left).

be interpreted locally on the respective sensor board, where 2D human joint positions are inferred from the images. The 2D human poses are streamed over a network to a central backend, where synchronization, data association, triangulation and post-processing are performed to fuse the 2D detections into 3D skeletons. Additionally, we propose a semantic feedback channel from backend to smart edge sensors. The allocentric 3D pose estimate is projected into the respective local views where it is combined with the joint detections. Thus, global context information can be incorporated into the local semantic model of the individual smart edge sensor, improving the pose estimation result.

The use of distributed smart edge sensors has several advantages over the centralized approaches more common in literature. As the images are processed directly on the sensor boards, raw images are not sent to the backend and only the 2D pose information has to be transmitted over the network. This significantly reduces the required communication bandwidth and mitigates privacy issues, as the abstract semantic information contains no personal details. Moreover, using a dedicated inference accelerator for each camera lessens the hardware requirements on the backend side, which, in a centralized architecture, can quickly become the bottleneck. On the other hand, using an embedded sensor platform poses challenges, as the employed vision models need to meet the limitations of the hardware. For this, we propose a lightweight 2D pose estimation model for efficient image processing locally on the smart sensors, at the edge of the network.

An essential prerequisite for multi-view 3D human pose estimation is a precise calibration of the camera network. For multiple reasons, this task is an inherently difficult one to solve (Maye, Furgale, and Siegwart, 2013): First, the calibration parameters change over time, by normal usage, e.g. due to vibration, thermal expansion, or moving parts. Therefore, it is not sufficient to calibrate the parameters only once during the setup of the smart edge sensor network. Instead, calibration must be performed repeatedly

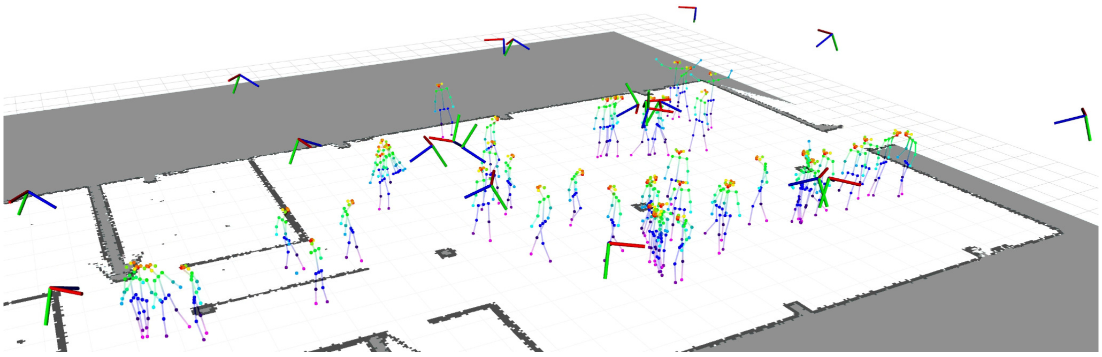

Figure 2.2: Extrinsic camera calibration using person keypoint detections computed locally on smart edge sensors: 3D person hypotheses, initialized from synchronized 2D keypoint observations from multiple views, are accumulated over time on a central backend and a factor graph optimization is solved to obtain the optimal camera poses (shown as coordinate systems with the blues axis being the viewing direction).

throughout its lifetime. Second, the calibration parameters cannot be measured directly with sufficient precision; they must be inferred from the data captured by the considered cameras. Further challenges for inferring calibration parameters from image data involve accounting for noisy measurements and collecting a sufficient amount of data points spread over the entirety of the image planes. Typically, the calibration is performed by actively deploying a calibration target of known correspondences in front of the cameras, e.g. a checkerboard pattern (Z. Zhang, 2000). However, this requires expertise and might be perceived as cumbersome and lengthy when it has to be applied repeatedly for a large multi-camera system.

We develop a novel, marker-free method to obtain and update the extrinsic calibration of the network of static smart edge sensors, where each sensor runs inference for 2D human pose estimation. The person keypoint detections can be used for either 3D human pose estimation, where the camera poses are assumed to be fixed and known, or for extrinsic camera calibration, where the keypoints are the calibration targets for updating and refining the camera poses from a rough initial estimate.

2D keypoints from multiple views are synchronized, filtered, and assigned to 3D person hypotheses on a central backend, where observations are accumulated over time, as illustrated in Fig. 2.2. Factor graph optimizations (Dellaert and Kaess, 2017) are repeatedly solved in an online manner to obtain the optimal camera poses from the observations. The method can handle multiple persons in the scene, as well as arbitrary occlusions, e.g. from tables or pillars. We assume the intrinsic parameters of the cameras to be known and a rough initial estimate of the extrinsic calibration to be available, which can easily be obtained, e.g. from a floor plan or by tape measure.

The proposed calibration method alleviates many of the issues mentioned above: No specific calibration target is required; it suffices for one or several persons to walk through the scene. The method handles data association between multiple observed persons and their unknown dimensions. We propose an efficient online algorithm that optimizes the camera poses on-the-fly, giving direct feedback on success and when enough data has been captured. The calibration procedure can easily be repeated to account for parameter change over time, without expert knowledge. Furthermore, person keypoints can be detected from a significantly larger range of viewing angles (e.g. front, back, or side-view) than the pattern of a classical checkerboard calibration target, which is well detected only from a frontal view. This facilitates the collection of a sufficient amount of corresponding data points visible in multiple cameras that well constrain the factor graph optimization, further reducing the time required for calibration.

In summary, the main contributions presented in this chapter are:

- a new real-time method for multi-view 3D human pose estimation dividing the computation between smart edge sensors performing image analysis locally for each camera view and a backend fusing the semantic interpretations of individual views and using a computationally efficient skeleton model to incorporate prior knowledge,
- a novel, marker-free online method for the extrinsic calibration of multiple smart edge sensors, relying on 2D human joint detections as calibration targets,
- a novel 3D / 2D feedback architecture enabling bidirectional communication on a semantic level between sensors and backend, and
- an extensive evaluation of the proposed approach for 3D human pose estimation on the single-person H3.6M dataset (Ionescu et al., 2014), the multi-person Campus and Shelf datasets (Belagiannis et al., 2014), and in own multi-person experiments in a less-controlled real-world environment, where the proposed method for online marker-free camera calibration is applied.

2.2 Related Work

Human Pose Estimation. Human pose estimation from multi-camera input has been investigated for many years in the computer vision and robotics communities. It refers to the task of detecting anatomical keypoints of persons on images. 2D keypoint detections from multiple, calibrated camera views are fused to obtain 3D human poses.

Early works (Belagiannis et al., 2014, 2015; Burenius, Sullivan, and Carlsson, 2013) use manually designed image features, such as histograms of oriented gradients (HOG) descriptors (Dalal and Triggs, 2005) or pictorial structures (Felzenszwalb and Huttenlocher, 2005; Fischler and Elschlager, 1973), for 2D part detection and combine multiple views using a graph-based body model. With the increasing success of deep-learning methods, more recent approaches (J. Dong et al., 2019; Pavlakos et al., 2017b; Qiu et al., 2019) employ 2D CNNs for human joint detection (Cao et al., 2021; J. Li et al., 2019; Xiao, Wu, and Wei, 2018) and recover the 3D pose using variants of the pictorial structures model (PSM) (Belagiannis et al., 2014; Burenius, Sullivan, and Carlsson, 2013). In these approaches, the body model consists of a graph with 3D joint locations as nodes and pairwise articulation constraints on the edges. While the PSM body model recovers 3D poses accurately, it is computationally very expensive and generally not real-time capable, due to a large volumetric grid used as discrete state space for optimization. In our work, we also employ a graph-based body model but use a fast, iterative optimization scheme (Kaess et al., 2012), achieving real-time operation.

Iskakov et al. (2019) propose a volumetric aggregation of 2D feature maps from multiple views into 3D voxel feature maps for the estimation of 3D body joints. Tu, Chunyu Wang, and Zeng (2020) extend the volumetric approach to multi-person scenarios, making data association decisions in the 3D feature volume, which increases the robustness. The voxel-based approaches provide good performance, but computationally and memory-intensive 3D convolutions make these approaches infeasible for real-time and large-area applications.

Several recent methods with a focus on computational efficiency have been proposed (Long Chen et al., 2020; Remelli et al., 2020). In these approaches, 3D pose estimation is based on direct triangulation of 2D joint detections without the usage of an expensive body model. Long Chen et al. (2020) propose a fast, iterative triangulation scheme but assume 2D pose detections as given. Remelli et al. (2020) consider the whole pipeline including 2D keypoint estimation but use a fully centralized approach while our method employs distributed sensors for 2D pose estimation.

Naikal, Lajevardi, and Sastry (2014) proposed a system for human joint detection and action recognition using a network of smart cameras transmitting only abstract image features to a central processing station. However, at the time, no CNN-based vision models were available for pose estimation on mobile devices, limiting the performance of their framework. Furthermore, their communication channel is purely feed-forward—no feedback for viewpoint fusion is implemented.

Qiu et al. (2019) present an approach for cross-view fusion to improve the estimated 2D poses of individual camera views. 3D poses are recovered using an offline recursive PSM implementation with a processing time of several seconds per frame (Remelli et al., 2020). While Qiu et al. propose to learn attention weights representing epipolar lines between pairs of views from data, Y. He et al. (2020) extend the approach to changing camera configurations by explicitly using extrinsic calibration parameters. This reduces the number of learnable parameters and prevents retraining the attention weights when the multi-camera setup changes.

In our work, we take up the idea of across-sensor viewpoint fusion but propose a different formulation. Qiu et al. (2019) and Y. He et al. (2020) implement the fusion between perspectives on a purely 2D basis, using epipolar constraints. Hence, a 2D joint in one view will be associated with all features on the corresponding epipolar lines of other views, which can be ambiguous. In contrast, we implement a semantic 3D / 2D feedback channel from backend to sensors based on reprojection of the estimated 3D skeleton into the individual camera views. Our work considers a network of smart edge sensors where each sensor node performs 2D human pose estimation, processing the image data locally on the sensor boards in real time and transmitting only the obtained keypoint data. A central backend fuses the data received by the sensors to perform 3D human pose estimation in real time via direct triangulation and a lightweight body model. The semantic feedback loop improves the 2D pose estimation on the sensor boards by incorporating global context information.

After the publication of our original work in 2021, the idea of using smart edge sensor networks for 3D human pose estimation was taken up by other authors: L. Zhang and J. Xu (2023) propose an energy-efficient multi-camera system with onboard 2D keypoint estimation for real-time 3D human pose estimation, adaptively selecting a subset of sensor nodes to process redundant views. They implement backward communication from backend to sensors only on a control level, to choose which sensor board processes a respective detection, but not on the data level to improve the local semantic models via semantic feedback. Recent works on centralized 3D human pose estimation, that do not distribute the computation to smart edge sensors, improve the generalizability of epipolar fusion-based methods to novel camera configurations using a stochastic approach (Bartol, Bojanić, and Petković, 2022), improve the efficiency of voxel-based methods through the decomposition of the 3D feature volume to different orthogonal projections (Ye et al., 2022), and propose the direct regression of 3D poses of multiple persons from multiple views with a transformer-based model (T. Wang et al., 2021).

Lightweight CNN Models for Human Joint Detection. Research interest in computer vision models that run efficiently on mobile and embedded devices has significantly increased in recent years. For human joint detection, OpenPose (Cao et al., 2021) is a groundbreaking work that enables real-time 2D multi-person keypoint detection. MobiPose (J. Zhang et al., 2020) investigates human pose estimation on smartphone SoCs without dedicated inference accelerators, using motion vector-based tracking. Xiao, Wu, and Wei (2018) propose a simple CNN architecture consisting only of a feature extractor and a deconvolutional head but use a standard ResNet backbone (K. He et al., 2016). Popular lightweight backbone architectures include EfficientNet (Tan and Le, 2019) and MobileNets (Howard et al., 2019, 2017). These architectures greatly reduce the number of parameters w.r.t. standard CNN feature extractors like ResNet, e. g. by replacing convolutions with depthwise-separable convolutions. Moreover, tensor processors for inference acceleration, like the Google Edge TPU (Seshadri et al., 2022), can be employed to efficiently run a CNN vision model within a limited size and energy budget. For compatibility with the Edge TPU, weights and activations of the model need to be quantized to 8-bit integer values using a quantization scheme as proposed by Jacob et al. (2018).

In our work, we employ the CNN architecture of Xiao, Wu, and Wei (2018) with a MobileNetV3 backbone (Howard et al., 2019) and apply 8-bit quantization for inference on the Edge TPU.

Camera Calibration. Traditional methods for camera calibration are based on using artificial image features, so-called *fiducials*. Their common idea is to deploy a calibration target with known correspondences in the overlapping field of view (FoV) of the considered cameras. Z. Zhang (2000) utilizes a checkerboard pattern on a planar surface to perform intrinsic camera calibration. The *kalibr* toolkit (Rehder et al., 2016) uses a planar grid of *AprilTags* (Olson, 2011) to perform offline extrinsic and intrinsic calibration of multiple cameras, which allows to fully resolve the target's orientation towards the cameras and is robust against occlusions. We apply this method to obtain a reference calibration for evaluating our work.

Reinke, Camurri, and Semini (2019) propose an offline method to find the relative poses between a set of (two) cameras and the base frame of a quadruped robot. They use a fiducial marker mounted on the endeffector of a limb as the calibration target. The camera poses are resolved using a *factor graph* (Dellaert and Kaess, 2017), modeling kinematic constraints between the marker frame and the base frame together with the visual constraints. We take up the idea of using factor graphs to model calibration constraints in our work.

To cope with the issues of traditional approaches, methods for camera calibration have been proposed that do not extract fiducial features from calibration targets but use naturally occurring features instead. Komorowski (2012) extracts scale-invariant feature transform (SIFT) features (Lowe, 1999) and finds correspondences between multiple views using random sample consensus (RANSAC) (Fischler and Bolles, 1981). They use segmentation to remove dynamic objects and validate their approach on stereo vision datasets. Their method is targeted towards one or a few small-baseline stereo cameras and offline processing of a small batch of images. Bhardwaj et al. (2018) calibrate traffic cameras by extracting vehicle instances via deep neural networks (DNNs) and matching them to a database of popular car models. The extracted features and known dimensions of the car models are then used to formulate a perspective-n-point (PnP) problem (Fischler and Bolles, 1981). They assume a planar ground surface in the vicinity of the cars and process results offline.

A variety of methods considering surveillance scenarios use pedestrians as calibration targets. Lv, T. Zhao, and Nevatia (2006) track head and feet detections of a single pedestrian walking on a planar surface during the leg-crossing phases to perform offline extrinsic calibration of a single camera based on the geometric properties of vanishing points. Following a tracking approach, they resolve the corresponding intrinsic parameters based on Z. Zhang (2000). Hödlmoser and Kampel (2010) use a similar approach, but expand the method to calibrate a camera network from pairwise relative calibrations. The absolute scale of the camera network is resolved by manually specifying the height of the walking person. Jingchen Liu, Collins, and Y. Liu (2011) require a moderately crowded scene to perform online intrinsic and extrinsic calibration of a single camera by assuming strong prior knowledge regarding the height distribution of the observed pedestrians. The approach is based on computing vanishing points using RANSAC. Jingchen Liu, Collins, and Y. Liu (2013) expand their method by introducing a joint calibration for a network of cameras based on the direct linear transform (DLT) (Hartley and Zisserman, 2003). Henning, Laidlow, and Leutenegger (2022) jointly optimize the trajectory of a monocular camera and a human body mesh by formulating an optimization problem in the form of a factor graph. They apply a human motion model to constrain sequential body postures and to resolve scale.

Guan et al. (2016) and Truong et al. (2019) detect head and feet keypoints for each observable pedestrian and perform pairwise triangulation assuming an average height for all visible persons in the image pair. They then compute the calibration offline, using RANSAC, followed by a gradient descent-based refinement scheme. Their resulting calibration is only defined up to an unknown scale factor, which must be resolved manually. The method assumes the center lines between all pedestrians to be parallel, in other words, all persons are assumed to stand upright during the calibration, whereas other poses, e.g. sitting persons, are not supported.

Our method, in contrast, extracts up to 17 keypoints per person (Lin et al., 2014) using CNNs and assumes neither the dimensions or height of the persons, nor their pose or orientation towards the cameras to be known. As for the unknown dimensions of the persons, the scale of our calibration is also ambiguous up to a single scale factor. To address this issue, we force the scale of the initial estimate of the extrinsic calibration to be maintained throughout the calibration procedure.

2.3 PRELIMINARIES

We consider a multi-camera system $\mathcal{S}_N$ of $N \geq 2$ projective cameras $\mathcal{C}_i$, $i \in [0 \,..\, N-1]$ with projection matrices $\boldsymbol{P}_i \in \mathbb{R}^{3\times 4}$. The projection matrices $\boldsymbol{P}_i$ project a 3D point

$\boldsymbol{x} \in \mathbb{R}^3$ defined by its homogeneous coordinates $\tilde{\boldsymbol{x}} = [\boldsymbol{x}^\top, 1]^\top \in \mathbb{R}^4$ onto the image plane of the respective camera $\mathcal{C}_i$ and are given as:

$$\boldsymbol{P}_i = \boldsymbol{K}_i \boldsymbol{C}_i \,, \tag{2.1}$$

with

$$\boldsymbol{K}_i = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \in \mathbb{R}^{3\times 3} \tag{2.2}$$

the intrinsic camera calibration consisting of the focal length parameters f_x, f_y and the optical center (c_x, c_y), and the extrinsic camera calibration given by the pose of the optical center of each camera

$$\boldsymbol{C}_i = \begin{pmatrix} \boldsymbol{R}_i & \boldsymbol{t}_i \end{pmatrix} \in \mathbb{R}^{3\times 4} \,, \tag{2.3}$$

defined by the translation $\boldsymbol{t}_i \in \mathbb{R}^3$ and rotation $\boldsymbol{R}_i \in \mathcal{SO}(3) \subset \mathbb{R}^{3\times 3}$.

In Sec. 2.4 we consider the cameras to be fully calibrated, i.e. the intrinsic and extrinsic calibration parameters are given and, hence, the projection matrices $\boldsymbol{P}_i$ are known, while in Sec. 2.5 we consider only the intrinsic parameters $\boldsymbol{K}_i$ of the cameras to be known and seek to obtain the extrinsic calibration parameters $\boldsymbol{C}_i$ from a rough initial estimate.

Both approaches, for 3D human pose estimation (Sec. 2.4) and for extrinsic camera calibration (Sec. 2.5) use 2D keypoint detections of a fixed set of J human joints as input. If not stated otherwise, we use the set of $J = 17$ joints as defined in the common objects in context (COCO) dataset (Lin et al., 2014): *nose*, *left eye*, *right eye*, *left ear*, *right ear*, *left shoulder*, *right shoulder*, *left elbow*, *right elbow*, *left wrist*, *right wrist*, *left hip*, *right hip*, *left knee*, *right knee*, *left ankle*, *right ankle*. The Human 3.6M dataset (Ionescu et al., 2014), used in Sec. 2.6.2 and for visualization in Fig. 2.1, Fig. 2.3, and Fig. 2.8, employs a slightly different skeleton model, with only one *head* keypoint instead of eyes and ears, and additional *neck*, *belly*, and *pelvis* keypoints. Our 3D body model always comprises the *neck* and *pelvis* joints, for compatibility between different datasets, calculated as mean of both shoulders, or hips, respectively, if not part of the 2D detections.

Each joint keypoint detection $\mathcal{D}_j$, $j \in [0 \mathinner{..} J-1]$ is given as

$$\mathcal{D}_j = \{\boldsymbol{u}_j, c_j, \boldsymbol{\Sigma}_j\} \,, \tag{2.4}$$

where $\boldsymbol{u}_j = [u_j, v_j]^\top$ are the image coordinates, c_j is the confidence value, and $\boldsymbol{\Sigma}_j \in \mathbb{R}^{2\times 2}$ is the covariance matrix of the detection.

Keypoint detections are calculated locally on the smart edge sensors and the 2D pose information is transmitted over a network to a central backend, using the robot operation system (ROS) (Quigley et al., 2009) as middleware for communication. The clocks of sensors and backend are software-synchronized and each 2D pose message includes a timestamp representing the capture time of the corresponding image. Sets of N corresponding 2D pose messages, one for each view, are determined based on the timestamps and further processed by the respective methods as detailed in the following.

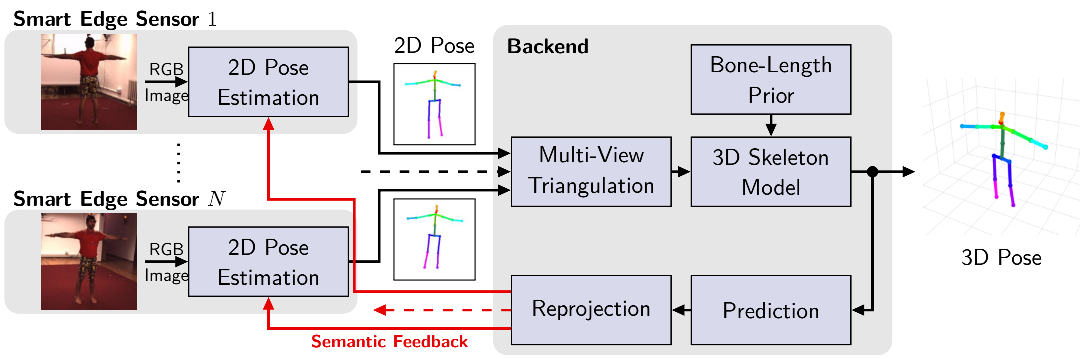

Figure 2.3: Overview of the proposed pipeline for 3D human pose estimation using smart edge sensors and semantic feedback. Images are analyzed locally on the sensor boards. Semantic pose information is transmitted to the backend where multiple views are fused into a 3D skeleton. The 3D pose is reprojected into local views and sent to sensors as semantic feedback.

2.4 Multi-View 3D Human Pose Estimation

Multi-view 3D human pose estimation refers to the task of estimating 3D positions of anatomical joint keypoints of persons observed from multiple viewpoints to create a skeleton-like representation of the human body. It is a prerequisite for many downstream tasks in computer-vision and robotics, such as action recognition, human-robot interaction, and anticipatory robot navigation.

We consider scenarios where N calibrated cameras $\mathcal{C}_i$, $i \in [0 \mathinner{..} N-1]$ with known projection matrices $\boldsymbol{P}_i$ perceive a scene with one or several individuals from multiple viewpoints. Our method is described for the single person case in the following. Extensions to handle multiple persons are described in Sec. 2.4.5. An overview of our proposed approach for multi-view 3D human pose estimation is given in Fig. 2.3.

2D human joint detections $\{\mathcal{D}_{j,i}\}_{j=1}^{J}$, as defined in (2.4), are calculated for each camera view $\mathcal{C}_i$, $i \in [0 \mathinner{..} N-1]$ directly on the respective smart edge sensor board using the vision model described in Sec. 2.4.1. The 2D pose information is then transmitted over a network to a central backend, where sets of N corresponding messages are synchronized based on the detection timestamps, and raw 3D poses are recovered via triangulation as detailed in Sec. 2.4.2.

A skeleton model (cf. Sec. 2.4.3), incorporating prior information on the typical bone-lengths of the human skeleton, is applied and outputs the final estimated 3D pose.

A semantic feedback channel from backend to sensors is implemented as described in Sec. 2.4.4, which enables each individual view to benefit from the fused 3D information. For this, first, a prediction step is performed to compensate for the pipeline delay. Second, the predicted 3D skeleton is reprojected into each camera view and sent to the sensors where it is incorporated into the local 2D pose estimation.

2.4.1 *2D Human Pose Estimation on Smart Edge Sensor*

The smart edge sensor platform employed in this work (cf. Fig. 2.1 top-right) is based on the Google Edge TPU Dev Board[1], equipped with an ARM Cortex-A53 quad-core

1 https://coral.ai/docs/dev-board/datasheet, accessed: 2023-08-01

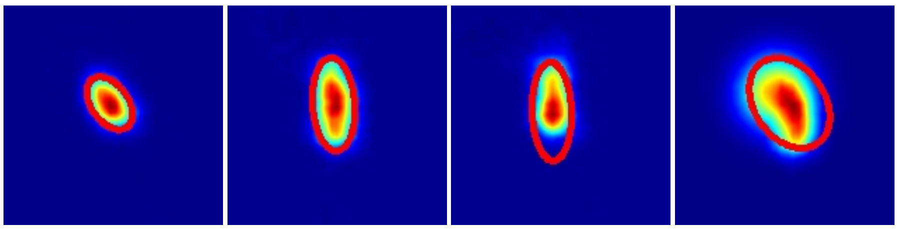

Figure 2.4: Heatmaps and derived covariances (3σ ellipses).

processor, the Edge TPU inference accelerator and 1 GB of shared RAM. A 5 Mpx RGB camera is connected to the board via the MIPI-CSI2 interface.

We adopt the CNN architecture of Xiao, Wu, and Wei (2018) for 2D human pose estimation, consisting of a backbone feature extractor and three transposed convolution layers to extract heatmaps from image features. To achieve real-time performance on the mobile sensor platform, we exchange the ResNet backbone used by Xiao, Wu, and Wei (2018) with the significantly more lightweight MobileNetV3 feature extractor (Howard et al., 2019). Furthermore, for execution on the Edge TPU, the model is quantized for 8-bit integer inference using post-training quantization (Jacob et al., 2018) as implemented in the TensorFlow machine learning (ML) framework (Abadi et al., 2016). In multi-person scenarios, a detector is also run on the sensor boards to provide person crops for the pose estimation network. It is based on the single shot detector (SSD) architecture (W. Liu et al., 2016), also using the MobileNetV3 backbone.

The output heatmaps $\boldsymbol{H}_{\text{det}}$ of the pose estimation model are a multi-channel image with one channel per joint, encoding the confidence of a joint being present at the pixel location. 2D joint locations $\boldsymbol{u}_j = [u_j, v_j]^\intercal$ are inferred as global maxima of the resp. heatmap channel, as single person crops are processed. The value of the heatmap at the joint position gives the corresponding confidence c_j. Only joints with confidence above a minimum threshold are considered as valid detections.

The covariance matrices $\boldsymbol{\Sigma}_j$ are determined as proposed by Pasqualetto Cassinis et al. (2020): Heatmap pixels with values above a threshold contribute to the empirical covariance with their x- and y-locations, weighted by the respective confidence:

$$\boldsymbol{\Sigma}_j = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix} , \tag{2.5}$$

$$\sigma_{xy}^2 = \frac{1}{K} \sum_{k=1}^{K} c_{j,k} \left(x_k - u_j\right) \cdot \left(y_k - v_j\right) , \tag{2.6}$$

where K is the number of contributing pixels and the mean is replaced by the peak $\boldsymbol{u}_j$ to model a distribution about the detected 2D joint location. Some representative examples of heatmaps and extracted covariances are shown in Fig. 2.4. The uncertainty in the heatmaps, including their directionality, is well captured by the covariance ellipses. Note, that the dispersion for asymmetric heatmaps, as in the third example of Fig. 2.4, is overestimated by the proposed procedure. Joint location, confidence and covariance are summarized as a joint detection $\mathcal{D}_j$ (2.4).

2.4.2 *Multi-View Fusion*

The 3D position $\hat{\boldsymbol{x}}_j$ of each joint j is recovered from a set of 2D detections $\{\mathcal{D}_{j,i}\}_{i=1}^{N}$ via triangulation using the direct linear transform (DLT) (Hartley and Zisserman, 2003). The relationship between 2D points $\boldsymbol{u}_i = [u_i, v_i]^\mathsf{T}$ from camera view $i \in \{1, \ldots, N\}$ and 3D point $\hat{\boldsymbol{x}}$ can be written as (omitting the joint index j for readability):

$$\boldsymbol{A}\tilde{\boldsymbol{x}} = \boldsymbol{0}\,, \tag{2.7}$$

with

$$\boldsymbol{A} = \begin{bmatrix} u_1\boldsymbol{p}_{1,3}^\mathsf{T} - \boldsymbol{p}_{1,1}^\mathsf{T} \\ v_1\boldsymbol{p}_{1,3}^\mathsf{T} - \boldsymbol{p}_{1,2}^\mathsf{T} \\ \cdots \\ u_N\boldsymbol{p}_{N,3}^\mathsf{T} - \boldsymbol{p}_{N,1}^\mathsf{T} \\ v_N\boldsymbol{p}_{N,3}^\mathsf{T} - \boldsymbol{p}_{N,2}^\mathsf{T} \end{bmatrix} \in \mathbb{R}^{2N\times 4}\,, \tag{2.8}$$

where $\tilde{\boldsymbol{x}} \in \mathbb{R}^4$ are the homogeneous coordinates of $\hat{\boldsymbol{x}}$ and $\boldsymbol{p}_{i,k}^\mathsf{T}$ denotes the k-th row of projection matrix $\boldsymbol{P}_i \in \mathbb{R}^{3\times 4}$. According to the DLT algorithm (Hartley and Zisserman, 2003), (2.7) is solved by a singular value decomposition (SVD) on $\boldsymbol{A}$, taking the unit singular vector corresponding to the smallest singular value of $\boldsymbol{A}$ as solution for $\tilde{\boldsymbol{x}}$. Finally, $\tilde{\boldsymbol{x}}$ is divided by its fourth coordinate to obtain the 3D vector $\hat{\boldsymbol{x}} = \tilde{\boldsymbol{x}}/(\tilde{\boldsymbol{x}})_4$.

The above formulation (2.7) assumes that all 2D detections make a similar contribution to the triangulation. However, 2D joint positions cannot be estimated reliably on some views, e. g. due to occlusions, which in turn degrades the result. The reliability of a detection is expressed by the heatmap confidence value c_i and can be incorporated into the DLT by multiplying each row of $\boldsymbol{A}$ with the corresponding element of a weight vector $\boldsymbol{w}$, reformulating (2.7) as:

$$(\boldsymbol{w} \circ \boldsymbol{A})\,\tilde{\boldsymbol{x}} = \boldsymbol{0}\,, \tag{2.9}$$

with

$$\boldsymbol{w} = \left(\frac{c_1}{\|\boldsymbol{a}_1^\mathsf{T}\|}, \frac{c_1}{\|\boldsymbol{a}_2^\mathsf{T}\|}, \ldots, \frac{c_N}{\left\|\boldsymbol{a}_{2N-1}^\mathsf{T}\right\|}, \frac{c_N}{\|\boldsymbol{a}_{2N}^\mathsf{T}\|}\right) \tag{2.10}$$

and $\circ$ the Hadamard product, similar to the approach of Long Chen et al. (2020). The confidence values in $\boldsymbol{w}$ are divided by the L^2-norm of the corresponding row of $\boldsymbol{A}$ to compensate for the different image locations of the joints in each view.

To obtain the 3D joint position $\hat{\boldsymbol{x}}_j$ and its covariance $\hat{\boldsymbol{\Sigma}}_j^{\mathrm{3D}}$, deterministic samples are propagated through the triangulation according to the Unscented Transform (Julier and Uhlmann, 2004). Sigma points are generated from the mean vector $\boldsymbol{\mu}_j = [\boldsymbol{u}_{j,1}^\mathsf{T}, \ldots, \boldsymbol{u}_{j,N}^\mathsf{T}]^\mathsf{T}$ and the block-diagonal matrix containing the 2D covariances $\boldsymbol{\Sigma}_{j,i}$ extracted from each heatmap. Each set of samples is triangulated according to (2.9) and $\hat{\boldsymbol{x}}_j$ and $\hat{\boldsymbol{\Sigma}}_j^{\mathrm{3D}}$ are determined as sample mean and covariance of the resulting points using weights given by the Unscented Transform.

2.4.3 *Skeleton Model*

We employ a factor graph model (Kaess et al., 2012) representing the tree structure of the human body, with 3D joint positions $\boldsymbol{x}_j$ as nodes connected by unary and pairwise factors on the edges.

The unary constraints are given by the triangulated joint positions $\hat{\boldsymbol{x}}_j$ and covariances $\hat{\boldsymbol{\Sigma}}_j^{3\text{D}}$ and follow a 3D Gaussian noise model:

$$f\left(\boldsymbol{x}_j\right) \sim \mathcal{N}\left(\boldsymbol{x}_j \middle| \hat{\boldsymbol{x}}_j, \hat{\boldsymbol{\Sigma}}_j^{3\text{D}}\right) . \tag{2.11}$$

The pairwise factors model typical limb lengths of the human body and also follow a Gaussian noise model:

$$g\left(\boldsymbol{x}_j, \boldsymbol{x}_k\right) \sim \mathcal{N}\left(\left\|\boldsymbol{x}_j - \boldsymbol{x}_k\right\| \mid l_{jk}, \sigma_\text{l}\right) , \tag{2.12}$$

with $\|\boldsymbol{x}_j - \boldsymbol{x}_k\|$ the Euclidean distance between joints $\boldsymbol{x}_j$ and $\boldsymbol{x}_k$. l_{jk} and σ_l denote mean and standard deviation of the length of the corresponding limb determined from the statistics of the H3.6M dataset (Ionescu et al., 2014).

The final 3D human poses are obtained by optimizing the factor graph using the Levenberg-Marquardt algorithm and the GTSAM framework (Kaess et al., 2012). The optimization is initialized with the poses from the previous frame, predicted using a linear velocity model.

2.4.4 *Semantic Feedback*

To enable the local semantic models of each sensor to benefit from the globally fused 3D pose, a feedback channel from backend to sensors is implemented in our framework.

First, the motion of the 3D skeleton is predicted using a linear velocity model for each joint to compensate for the pipeline delay Δt. Second, predicted 2D joint positions $\{\hat{\boldsymbol{u}}_{j,i}\}$ and their image-plane covariances $\{\hat{\boldsymbol{\Sigma}}_{j,i}\}$ are determined by reprojecting the predicted 3D pose and its covariance extracted from the factor graph into each sensor view i using the projection matrix $\boldsymbol{P}_i$ and the Unscented Transform (Julier and Uhlmann, 2004).

The reprojected feedback skeleton is sent to the smart edge sensors, where a feedback heatmap $\boldsymbol{H}_\text{fb}$ is rendered to be fused with the detected heatmap $\boldsymbol{H}_\text{det}$ of the current image crop. For each joint $\hat{\boldsymbol{u}}_j$, a 2D Gaussian blob is rendered in the corresponding heatmap channel according to the reprojected covariance matrix $\hat{\boldsymbol{\Sigma}}_j$.

The heatmaps are fused via weighted addition of detection, feedback, and their element-wise multiplication:

$$\boldsymbol{H}_\text{fused} = s\left(\left(1-\alpha-\beta\right)\boldsymbol{H}_\text{det} + \alpha\,\boldsymbol{H}_\text{fb} + \beta\left(\boldsymbol{H}_\text{fb} \circ \boldsymbol{H}_\text{det}\right)\right) , \tag{2.13}$$

with $\alpha + \beta < 1$. The scale s is set as $(1-\alpha-\beta)^{-1}$ to ensure that positive feedback always increases the joint confidence. The feedback gains α and β are important design parameters of our method. A sufficient weight must be accorded to the feedback to improve the raw detections, but too high gains can cause instability. Hence, the feedback gains are learned using a hyper-parameter search (Bergstra, Yamins, and Cox, 2013) optimizing the 3D pose error.

The above formulation models an arbitrary combination of additive and multiplicative feedback and can efficiently be executed on the embedded processor of the sensor board.

Examples of the heatmap fusion are shown in Fig. 2.8. Through the feedback loop, evidence for joint occurrence from detection and feedback is combined in the fused heatmap, improving the accuracy of the joint locations and reducing their uncertainty.

2.4.5 *Multi-Person Pose Estimation*

To handle real-world scenes (cf. Sec. 2.6.3 and 2.6.4), we extend our method to estimate the poses of multiple persons at a time. Person detections are associated across camera views based on the epipolar distance of their joints using the efficient iterative greedy matching proposed by Tanke and Gall (2019). The rest of the pipeline is then run for each person observed in at least two views to compute 3D poses and feedback. A feedback skeleton is associated to its corresponding 2D detection based on the intersection over union (IoU) of their bounding boxes.

2.5 Extrinsic Camera Calibration

While in the previous section, we assumed a multi-camera network with known intrinsic and extrinsic calibration, in this section we introduce our method to obtain and refine the camera poses using person keypoint detections as calibration targets. A precise calibration of the camera network is an important prerequisite for the multi-view human pose estimation approach described in the previous section where errors in calibration quickly lead to a degradation in performance. Furthermore, the extrinsic calibration of the smart edge sensor network must be updated repeatedly throughout its lifetime, when sensors are added or moved, or due to environmental effects such as vibration, thermal expansion, or moving parts.

Our method uses the 2D keypoint detection streams of the multi-camera system $\mathcal{S}_N$ with the projective cameras $\mathcal{C}_i$, $i \in [0 \mathinner{..} N-1]$, to extract and maximize knowledge about the camera poses w.r.t. a reference coordinate frame in real time. Concretely, the task is to find the pose of the optical center $\boldsymbol{C}_i$ (Eq. (2.3)) of each camera $\mathcal{C}_i$. We call this the *extrinsic calibration* of the multi-camera system $\mathcal{S}_N$. Without loss of generality, we set the camera $\mathcal{C}_0$ as the origin of the global reference frame with $\boldsymbol{C}_0 = \boldsymbol{I}_{4\times4}$. In its local coordinate system, the view direction of each camera is the z-axis.

Fig. 2.5 gives an overview of our proposed pipeline. Each camera stream is fed into a person keypoint detector on the connected inference accelerator (cf. Sec. 2.4.1) of the smart edge sensor. The keypoint detections are transmitted to a central backend where they are time-synchronized and processed further.

The *preprocessing* stage removes redundant and noisy detections after which *data association* is performed, where correspondences between person detections from multiple views are established. Corresponding person detections are fused to form a person hypothesis and attached to a queue, which serves to decouple the preprocessing stage from the rest of the pipeline. The *optimization* stage continuously reads from this queue, selects several person hypotheses, and uses them to construct and solve an optimization problem in the form of a factor graph (Dellaert and Kaess, 2017). The *refinement* stage updates the current estimate of the extrinsic calibration by smoothing the intermediate results generated by the optimization and compensates for scaling drift w.r.t. the initialization.

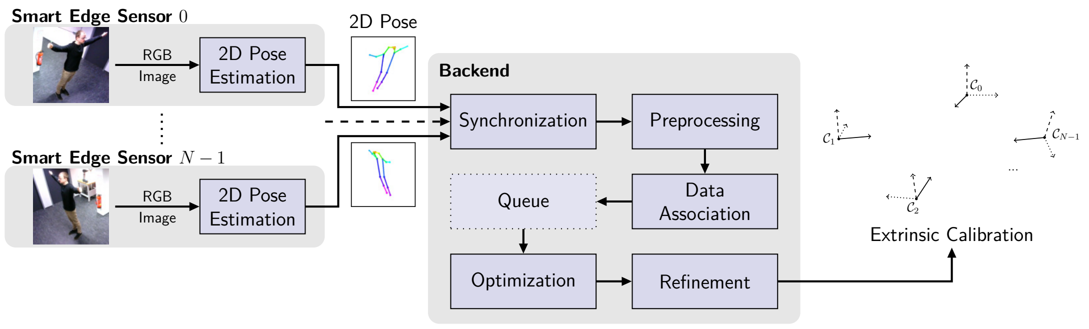

Figure 2.5: Proposed pipeline for extrinsic camera calibration using smart edge sensors and person keypoint detections. Images are analyzed locally on the sensor boards. Keypoint detections are transmitted to the backend where multiple views are fused to construct and solve optimization problems using factor graphs. A queue decouples the preprocessing and optimization stages.

As prerequisites for our method, we assume the intrinsic parameters $\boldsymbol{K}_i$ (2.2) of the cameras to be known, e.g. from factory calibration, and a rough initial estimate of the extrinsic calibration to be available, e.g. by tape measure or from a floor plan. The FoVs of all cameras must overlap such that $\mathcal{S}_N$ forms a connected graph.

2.5.1 *Preprocessing*

The backend receives N streams of keypoint detections $\{\mathcal{D}\}$ (Eq. (2.4)), and synchronizes and preprocesses them so that they can be used for optimization.

First, incoming streams are synchronized into sets of time-corresponding detection messages of size N, subsequently referred to as *framesets*. The preprocessing then rejects false, noisy, redundant, and low-quality detections and passes through only accurate detections suitable for improving the extrinsic calibration. We do this by checking different conditions for each frameset, regarding the number of detections per sensor, the timestamps associated to each sensor, or the confidence value of each detection.

In particular, we reject all framesets where the maximum span or standard deviation of timestamps exceeds a threshold and consider only joint detections with a minimum confidence of $c_j \geq 0.6$. We further require the hip and shoulder detections of each person to be valid, which is necessary for robust data association.

After filtering, we use the distortion coefficients of each camera to undistort the coordinates of all valid detections using the *OpenCV* library (Bradski, 2000).

2.5.2 *Data Association*

In the data association step, we find correspondences between detections from different sensors based on the current estimate of the multi-view geometry of the camera network, which can still be inaccurate. Each keypoint detection $\mathcal{D}_{j,p}$ is associated to a person hypothesis p. First, we back-project each 2D detection $\mathcal{D}$ into 3D, obtaining a ray $\boldsymbol{r}_{\mathcal{D}}$ with undetermined depth originating at the optical center of the respective camera.

Next, we reduce each ray $\boldsymbol{r}_{\mathcal{D}}$ to a line segment $\boldsymbol{l}_{\mathcal{D}}$ by estimating the depth interval $[z_{\text{min}}, z_{\text{max}}]$ of each detection $\mathcal{D}$, as illustrated in Fig. 2.6. For the depth interval estimation, we assume a minimum and maximum torso height and width for the detected

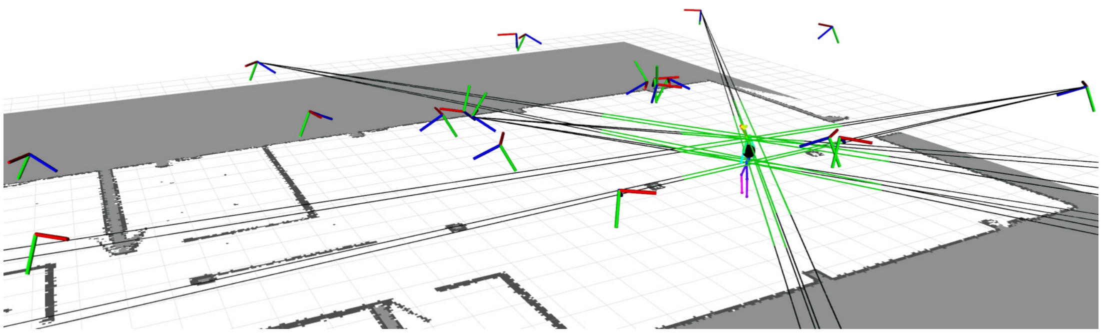

Figure 2.6: Data association: 3D back-projection rays embedded in the global coordinate system for the joint detections of one person (black), the corresponding reduction to line segments after applying depth estimation (green), and the center of mass of the corresponding person hypothesis (black). 3D human poses shown for illustration purposes only.

persons, derived from a specified minimum and maximum person height to be expected during calibration. The four torso keypoints (shoulders and hips) are empirically the most stable and least occluded, and their physical distances are approximately constant for a person, regardless of their pose, due to the human anatomy. The respective keypoint distance on the image plane, however, depends on the persons' orientation towards the cameras. Here, we assume worst-case orientations, leading to larger depth intervals.

In summary, we do not require persons to always stand upright but support arbitrary poses and orientations towards the cameras instead. Specifying a short person height interval leads to a more constrained search space during data association, accommodating for an inaccurate initial estimate of the extrinsic calibration or a crowded scene. A wider interval, however, yields equal results in common scenarios, while supporting small and tall persons as calibration targets alike. Depth estimation is the only component in the pipeline, where human anatomy is exploited.

To find the correspondences between person detections from multiple views, we deploy an iterative greedy search method similar to the approach of Tanke and Gall (2019), using the distances between the estimated line segments as data association cost. We define the distance of two line segments $\boldsymbol{l}_1$, $\boldsymbol{l}_2$ as the Closest Point-distance described by Wirtz and Paulus (2016):

$$d_{\text{closestpoint}}(\boldsymbol{l}_1, \boldsymbol{l}_2) = \min\left(d_{\perp}\left(\boldsymbol{l}_1, \boldsymbol{l}_2\right), d_{\perp}\left(\boldsymbol{l}_2, \boldsymbol{l}_1\right)\right) . \tag{2.14}$$

To further improve the robustness of the approach, as the extrinsic calibration is not precisely known during the calibration procedure, we iterate over all person detections, sorted in ascending depth order, utilizing the depth estimation $(z_{\text{min}} + z_{\text{max}})/2$ of each person detection. This exploits the fact that near person detections have a relatively short interval $[z_{\text{min}}, z_{\text{max}}]$ and, thus, a more constrained localization in 3D space.

Finally, we compute the *center of mass* for each person hypothesis, which serves to roughly localize them in 3D space by averaging the line segment centers of all assigned torso keypoints. The data association stage outputs a list of person hypotheses $\mathcal{H}_p$ observed by at least two different cameras, which will be used to construct a factor graph optimization problem in the following.

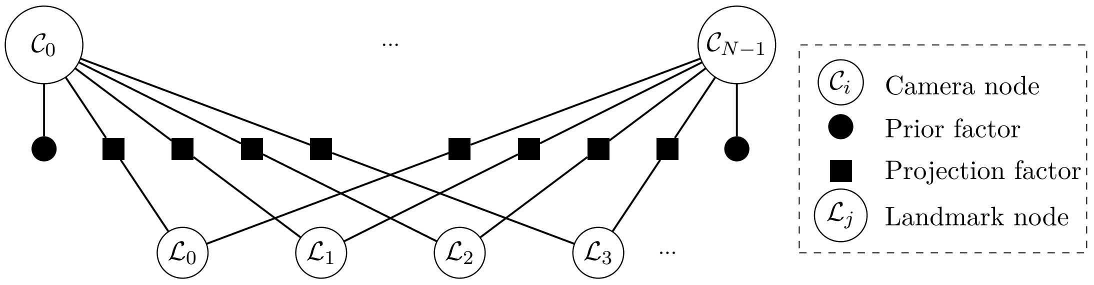

Figure 2.7: Factor graph with camera variable nodes $\mathcal{C}_i$ for the camera poses $\boldsymbol{C}_i$ and landmark variable nodes $\mathcal{L}_j$ for the 3D person joint positions $\boldsymbol{x}_j$. Camera and landmark nodes can be connected via binary projection factors to constrain the reprojection error of a person keypoint detection. Each landmark node must be connected to at least two projection factors to allow triangulation. All camera nodes are connected to a unary prior factor that encodes the initial uncertainty of the camera pose.

2.5.3 *Factor Graph Optimization*

The optimization stage processes the person hypotheses $\mathcal{H}_p$ obtained through data association to extract knowledge about the extrinsic calibration of the utilized multi-camera system. To this end, we construct a factor graph (Kaess et al., 2012) encoding projective constraints, based on a selection of person hypotheses, as well as prior knowledge about the camera poses given by the initial estimate of the extrinsic calibration or the results of previous optimization cycles.

The optimal selection of person hypotheses to be used in an optimization cycle is determined by a selection algorithm from all available hypotheses. The algorithm ensures an optimal spatial and temporal distribution of observations to obtain a well-constrained optimization problem, while maintaining a reasonable degree of entropy between selections over successive optimization cycles.

For this, we generate a random permutation of the indices of all available person hypotheses, biased towards selecting newer hypotheses. Selecting newer hypotheses with higher probability is advantageous because their data association and centers of mass are estimated more reliably as the extrinsic calibration improves over time.

Additionally, we ensure a minimum spacing between all selected person hypotheses, w.r.t. to their center of mass, by only including the next person hypothesis within the permutation if its distance towards all previously selected person hypotheses is above a spacing threshold $s = 0.2\,\text{m}$.

For each optimization cycle t, we construct a factor graph $\mathcal{G}_t$ by using a selection of person hypotheses $\mathcal{H}_t \subset \mathcal{H}_p$. A factor graph is a bipartite graph consisting of *variable nodes* and *factor nodes* (Dellaert and Kaess, 2017). Variable nodes represent the unknown random variables of the optimization problem, i.e. the 3D joint positions $\boldsymbol{x}_j$ of the person hypotheses in $\mathcal{H}_t$ (*landmark nodes* $\mathcal{L}_j$) and the considered camera poses $\boldsymbol{C}_i$ of the multi-camera system $\mathcal{S}_N$ (*camera nodes* $\mathcal{C}_i$).

Factor nodes constrain the variable nodes by encoding the available knowledge about the underlying distribution of the considered random variables. Specifically, the constraints are obtained from the observations contained in $\mathcal{H}_t$ and the resulting camera poses from previous optimization cycles. Each factor node uses a Gaussian noise model that reflects the confidence in the constraint it represents. The constructed factor graph is illustrated in Fig. 2.7.

We equip each camera node $\mathcal{C}_i$ with a unary prior factor that encodes prior knowledge about the camera pose and its uncertainty, and use binary projection factors connecting camera nodes to landmark nodes to encode observation constraints based on person keypoint detections. Projection factors calculate the reprojection error for a 2D detection w.r.t. the corresponding camera pose and landmark position using the known intrinsic parameters $\boldsymbol{K}_i$.

Camera nodes are initialized with the current estimate for the extrinsic calibration (for $t = 0$, we use the initial estimate and for $t > 0$, we reuse the result of the previous time step). Landmark nodes are initialized by triangulation of the 2D observations (cf. Sec. 2.4.2) using the latest camera geometry estimate. Note, that we perform triangulation in every optimization cycle, even when using a person hypothesis that was already utilized in a previous optimization cycle. Hence, the triangulation results are updated based on the current estimate of the extrinsic calibration.

We solve each factor graph $\mathcal{G}_t$ for the most likely camera poses by applying a Levenberg-Marquardt optimization scheme, provided by the GTSAM framework (Kaess et al., 2012). A successful optimization yields a new candidate for the extrinsic calibration of $\mathcal{S}_N$. We forward this candidate to the refinement stage where the current estimate of the extrinsic calibration will be updated based on this candidate. The updated estimate for the extrinsic calibration will then be used for constructing and initializing the factor graph in the next optimization cycle.

2.5.4 *Camera Pose Refinement*

After each successful optimization, we obtain new candidates $\hat{\boldsymbol{C}}_{i,t}$ for the extrinsic calibration of a subset of cameras $\{\mathcal{C}_i\} \subseteq \mathcal{S}_N$ that were constrained by the factor graph $\mathcal{G}_t$. We smooth between the previous state and the new measurement using a Kalman filter to obtain the current estimate of the extrinsic calibration $\boldsymbol{C}_{i,t}$. As each optimization cycle contains only a limited number of observations in the factor graph to enable real-time operation, smoothing prevents overconfidence towards a specific set of observations and improves the convergence behavior. We update the previous estimate with the result of the factor graph optimization, using its marginalized uncertainty as measurement noise. Between optimization cycles, we add a constant process noise to the uncertainty of each predicted camera pose, enabling convergence over longer time horizons. Finally, we prevent scaling drift of the updated extrinsic calibration by applying the scaling factor that minimizes the distance towards the initial estimate of the extrinsic calibration according to Umeyama's method (Umeyama, 1991).

2.6 Evaluation and Experiments

We evaluate the proposed approach for multi-view 3D human pose estimation on three widely-used public datasets: The Human 3.6M dataset (Ionescu et al., 2014) and the multi-person Campus and Shelf datasets (Belagiannis et al., 2014). We further run experiments in real-world multi-person scenarios in our lab, where we also evaluate the proposed method for extrinsic camera calibration.

2.6.1 *Dataset and Metrics*

2.6.1.1 *Human 3.6M*

The Human 3.6M dataset (Ionescu et al., 2014) is a large-scale public dataset for single-person multi-view 3D human pose estimation. It contains 3.6 million frames of eleven different actors, captured by four synchronized cameras together with ground truth 2D and 3D poses.

We measure the 2D pose estimation accuracy as the percentage of correctly detected joints, the joint detection rate (JDR). A joint is correctly detected when its distance towards the corresponding ground-truth annotation is smaller than a threshold. We set the JDR threshold to half the head size as proposed by Qiu et al. (2019).

The 3D pose accuracy is measured by the mean per joint position error (MPJPE) between estimated 3D joints $\boldsymbol{x}_j$ and ground truth 3D joints $\boldsymbol{x}_{j,\text{gt}}$:

$$\text{MPJPE} = \frac{1}{J}\sum_{j=1}^{J}\|\boldsymbol{x}_j - \boldsymbol{x}_{j,\text{gt}}\| \,. \tag{2.15}$$

2.6.1.2 *Campus and Shelf*

The Campus dataset (Belagiannis et al., 2014) consists of three people interacting outdoors, captured by three calibrated cameras. The Shelf dataset (Belagiannis et al., 2014) consists of four people interacting and disassembling a shelf in a small indoor area, captured by five cameras. It is a more complex setting compared to Campus, as frequent occlusions occur between persons and with the shelf. The same evaluation protocol as in previous works (Belagiannis et al., 2014, 2015; Long Chen et al., 2020; J. Dong et al., 2019) is used, employing the 3D percentage of correct parts (PCP) metric (Burenius, Sullivan, and Carlsson, 2013). A body part is considered as correctly estimated if the average of the Euclidean distances of start and end point of the limb with the ground-truth is smaller than half the limb length.

2.6.2 *Evaluation on the H3.6M Dataset*

2.6.2.1 *Implementation Details*

We adopt the network for pose estimation described in Sec. 2.4.1 and use two different training schemes: (i) training solely on H3.6M training data and (ii) pretraining the network on person keypoints from the COCO dataset (Lin et al., 2014) and finetuning on H3.6M. The input resolution is set to 256×256 pixels and we use the joint classes as defined in the H3.6M skeleton model (cf. Sec. 2.3).

As is common practice in literature (Pavlakos et al., 2017b; Qiu et al., 2019; Tome et al., 2018), we use subjects 1, 5, 6, 7, 8 for training and subjects 9 and 11 for testing. Input images are cropped using the provided ground-truth bounding box and evaluation is performed for every 5th frame as subsequent frames are highly similar at the original frame rate of 50 Hz.

All four image streams are processed simultaneously, each on its own smart edge sensor board. The estimated 2D skeletons are transmitted to the backend, where they are triangulated, and the skeleton model is applied. We report evaluation results with

Table 2.1: 2D joint detection rate (JDR) (%) for different feedback modes and training data.

Feedback	Training Data	Hips	Knees	Ankls	Shlds	Elbs	Wrists	**Avg**
w/o fb	H3.6M	99.2	96.1	90.3	93.3	93.3	89.1	95.1
w fb	H3.6M	**99.5**	97.6	96.1	97.2	96.5	**94.8**	97.5
w/o fb	COCO + H3.6M	99.3	97.1	96.9	98.9	96.2	92.8	97.6
w fb	COCO + H3.6M	99.3	**98.0**	**97.8**	**99.0**	**97.1**	**94.8**	**98.2**

Table 2.2: 3D pose error (MPJPE) (*mm*) for different joints, feedback modes, and training data.

Feedback	Training Data	Hips	Knees	Ankls	Shlds	Elbs	Wrists	**Avg**
w/o fb	H3.6M	22.2	29.4	58.6	40.5	43.8	39.8	32.9
w fb	H3.6M	22.1	28.0	47.2	36.7	38.6	33.9	29.8
w/o fb	COCO + H3.6M	**19.2**	25.5	38.0	25.6	30.7	29.4	24.0
w fb	COCO + H3.6M	**19.2**	**24.9**	**36.9**	**25.5**	**29.9**	**28.3**	**23.5**

and without using the proposed feedback channel. The parameters for the heatmap fusion (2.13) are determined as $\alpha = 0.15$ and $\beta = 0.75$.

2.6.2.2 *Quantitative Results*

Tab. 2.1 shows evaluation results for the accuracy of the 2D pose estimation calculated on the smart edge sensors, depending on the employed feedback mode and training data. Our experiments indicate that using the feedback channel (cf. Sec. 2.4.4) significantly improves the JDR accuracy. The improvement is highest for the often-occluded wrist and ankle joints, 5.7 % resp. 5.8 % for the H3.6M-only model. For the better visible joint classes, detection is easier also without feedback and the improvement is smaller.

Pretraining the model on the COCO keypoint dataset generally improves performance, as the model trained on a larger and more varying dataset generalizes better to unknown scenes. For the stronger model, the gain from using the feedback signal is smaller, but still amounts to 2 % for the wrists which are the most difficult joints to detect.

The improved 2D joint detections in turn lead to more accurate 3D poses, as is evident from the results in Tab. 2.2, which shows the 3D pose error. As in the 2D case, the improvement from the feedback channel is more significant for the weaker model and highest for ankles and wrists, around 11 mm resp. 6 mm for the H3.6M-only network.

In Tab. 2.3, the MPJPE 3D pose error is shown per action category and compared to other approaches from literature. 3D pose errors after application of the resp. post-processing step or skeleton model are reported. The recent approaches by Qiu et al. (2019) and Remelli et al. (2020) as well as our method provide significant improvements over the older methods of Pavlakos et al. (2017b) and Tome et al. (2018). Comparing the models trained on H3.6M only, the results of our approach using feedback are better than Qiu et al. (2019) and Remelli et al. (2020) for 10 of the 15 action categories and reduce the average error. The proposed semantic feedback channel is key to this

Table 2.3: Evaluation result on H3.6M dataset: MPJPE 3D pose error (*mm*) per action type. 3D poses after application of the resp. post-processing or skeleton model are reported. $^+$ denotes using additional training data.

Method	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	**Avg**
Pavlakos et al. (2017b)	41.2	49.2	42.8	43.4	55.6	46.9	40.3	63.7	97.6	119.9	52.1	42.7	51.9	41.8	39.4	56.9
Tome et al. (2018)	43.3	49.6	42.0	48.8	51.1	64.3	40.3	43.3	66.0	95.2	50.2	52.2	51.1	43.9	45.3	52.8
Qiu et al. (2019)	28.9	32.5	26.6	28.1	**28.3**	29.3	28.0	36.8	42.0	**30.5**	35.6	30.0	28.3	30.0	30.5	31.2
Remelli et al. (2020)	27.3	32.1	**25.0**	**26.5**	29.3	35.4	28.8	31.6	**36.4**	31.7	31.2	29.9	**26.9**	33.7	30.4	30.2
Ours, w/o fb	27.7	36.5	27.8	27.1	33.9	33.1	29.3	33.6	41.3	42.5	32.8	33.5	33.3	27.8	27.2	32.9
Ours, w fb	**27.1**	**29.9**	27.0	**26.5**	31.3	**28.9**	**27.1**	**29.8**	36.5	36.0	**30.8**	**29.3**	29.7	**27.3**	**26.3**	**29.8**
Qiu et al. (2019)$^+$	24.0	26.7	23.2	24.3	24.8	**22.8**	24.1	28.6	32.1	26.9	31.0	25.6	25.0	28.1	24.4	26.2
Ours$^+$, w/o fb	**22.4**	24.3	22.4	**21.7**	24.6	24.7	22.4	**22.6**	26.8	28.4	25.0	23.1	**24.5**	22.0	21.5	24.0
Ours$^+$, w fb	**22.4**	**24.0**	**22.2**	**21.7**	**24.0**	23.9	**22.1**	**22.6**	**26.0**	**26.8**	**24.5**	**22.8**	24.6	**21.8**	**21.3**	**23.5**

Table 2.4: Average inference time and model size for H3.6M dataset (Values for (Qiu et al., 2019) taken from (Remelli et al., 2020)).

	Qiu et al. (2019)	Remelli et al. (2020)	Ours
Inference Time	8.4 s	0.040 s	**0.024 s**
Model Size	2.1 GB	251 MB	**4 × 12 MB**

improvement over the literature. When using additional training data, our method also achieves state-of-the-art results.

In Tab. 2.4, we compare the inference time per frameset (i. e. for a set of four images) and model size of our approach with the recent approaches (Qiu et al., 2019) and (Remelli et al., 2020). The approach of Qiu et al. (2019) is an offline method with a run time of several seconds. The approach of Remelli et al. (2020) achieves near real-time performance on a powerful desktop GPU. The run time of our method is still about 40 % faster while running on efficient embedded sensor boards and a backend that doesn't require a GPU. Our pose estimation model, optimized for the Edge TPU inference accelerator, requires only 12 MB of memory, significantly less than the models of other approaches.

Results of an ablation study on the impacts of different parts of our proposed pipeline are shown in Tab. 2.5. Using the skeleton model to post-process the raw 3D poses obtained by triangulation significantly improves the average MPJPE. Employing the directional covariances extracted from the heatmaps, instead of modeling uncertainties only by the confidence value, again reduces the error. The semantic feedback further improves the result, where the proposed combination of additive and multiplicative feedback is more efficient than using only a single type. The impact of the feedback signal for each action class can also be observed in Tab. 2.3. It improves the results for all actions for the H3.6M-only trained model, with an average improvement of 3.1 mm. When using the stronger pose estimation model trained on additional data, the average improvement amounts to 0.5 mm. The feedback signal is more important when the raw pose estimates are less accurate but reduces the average 3D pose error in both cases.

Table 2.5: Ablation study on the impact of various components on the MPJPE (*mm*).

skeleton model	heatmap covariance	additive feedback	multiplicative feedback	MPJPE
-	-	-	-	38.08
✓	-	-	-	33.41
✓	✓	-	-	32.94
✓	✓	✓	-	30.07
✓	✓	-	✓	30.10
✓	✓	✓	✓	**29.82**

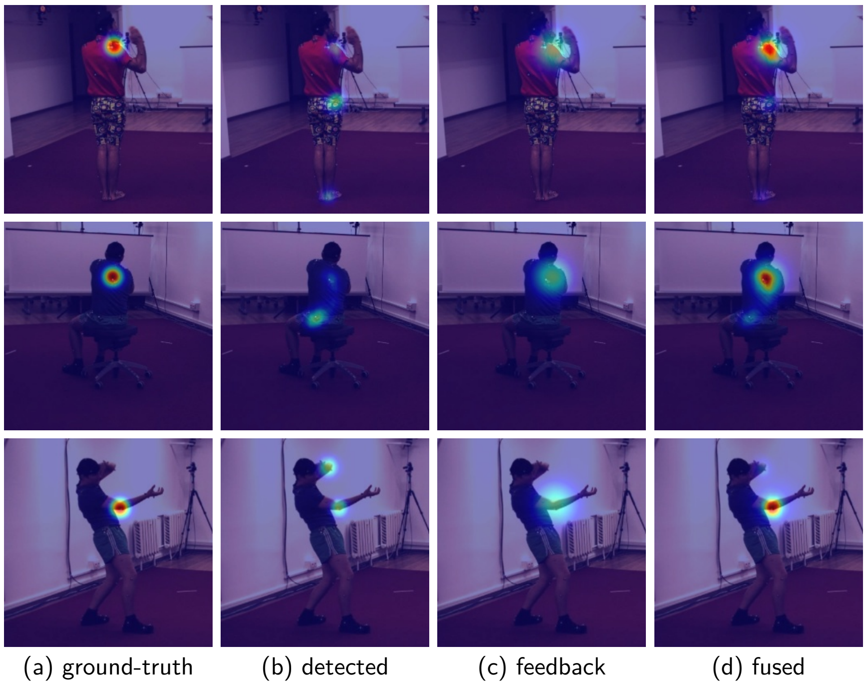

Figure 2.8: Samples of the heatmap fusion approach for left wrist (Rows 1-2) and right elbow (Row 3): Detected heatmap (b) and feedback heatmap (c) are combined into the fused heatmap (d). In difficult situations such as occlusions (Rows 1-2) or left-right inversions (Row 3), feedback results in a heatmap closer to the ground-truth (a).

2.6.2.3 *Qualitative Results*

In addition, we qualitatively show how the proposed feedback loop improves the pose estimation result. Fig. 2.8 shows three example situations where the feedback heatmap helps to recover from incorrect or imprecise 2D joint detections. The images are overlaid with the respective heatmaps for a specific joint. In the first and second row, the left wrist of the actors is occluded by their body and the detected heatmap is very inaccurate. However, from the perspectives of other cameras, the joint is visible, and its 3D position can be estimated. This is reflected in the feedback heatmap which predicts the joint detection close to the ground truth location. The resulting fused heatmap, obtained by combining detection and feedback according to (2.13), permits to accurately estimate the respective joint despite the imprecise local detection. In Row 3 of Fig. 2.8, a similar situation is shown, but for the right elbow, which here cannot be distinguished from the left elbow due to the challenging pose.

Table 2.6: Evaluation result on Campus and Shelf dataset: Percentage of Correct Parts (PCP) (%) and average run time of 2D and 3D pose inference. '-' means offline pre-computation.

	PCP (%)				Inference Time	
Campus	Actor 1	Actor 2	Actor 3	Avg	2D pose	3D pose
Belagiannis et al. (2015)	83.0	73.0	78.0	78.0	-	1 s
J. Dong et al. (2019)	97.6	93.3	98.0	96.3	-	105 ms
Long Chen et al. (2020)	97.1	**94.1**	**98.6**	96.6	-	**1.6 ms**
Ours, w/o fb	98.8	93.4	97.5	96.6	**30 ms**	8.8 ms
Ours, w fb	**99.2**	93.6	98.3	**97.0**	**30 ms**	8.8 ms
Shelf	Actor 1	Actor 2	Actor 3	Avg	2D pose	3D pose
Belagiannis et al. (2015)	75.0	67.0	86.0	76.0	-	1 s
J. Dong et al. (2019)	98.8	94.1	**97.8**	96.9	-	105 ms
Long Chen et al. (2020)	**99.6**	93.2	97.5	96.8	-	**3.1 ms**
Ours, w/o fb	99.4	94.6	96.8	96.9	**40 ms**	20 ms
Ours, w fb	99.3	**95.7**	97.3	**97.4**	**40 ms**	20 ms

2.6.3 *Evaluation on the Campus and Shelf Datasets*

2.6.3.1 *Implementation Details*

To process multi-person scenes, a person detector is employed together with the pose estimation model (cf. Sec. 2.4.1). The detector is trained for 130 epochs on the person class of the COCO dataset (Lin et al., 2014), for input images of 640×480 px and achieves a mean average precision (mAP) of 44.6 %. The pose estimation network is trained for 140 epochs on COCO for person crops of 192×256 px using the joint classes as defined in the COCO skeleton model (cf. Sec. 2.3). It achieves a mAP of 69.6 % in FP32-mode and 68.4 % in INT8-mode on the COCO validation set using ground-truth detections. Note, that the generic detector and pose estimation networks are employed without any fine-tuning on the evaluated datasets.

The three or five image streams of the respective dataset are processed simultaneously on the sensor boards. The entire image is passed to the detector and image crops of the detected persons are analyzed by the pose estimation network. To improve the processing speed, the detector is run only once per second. In between, the crops are determined based on the detections of the previous frame. This is necessary because alternating between models is inefficient on the Edge TPU as parameter caching cannot come into effect in this case (Seshadri et al., 2022).

On the backend, the estimated 2D poses are synchronized based on their timestamps and the framework is run in multi-person mode as detailed in Sec. 2.4.5. The feedback delay amounts to one frame during dataset processing.

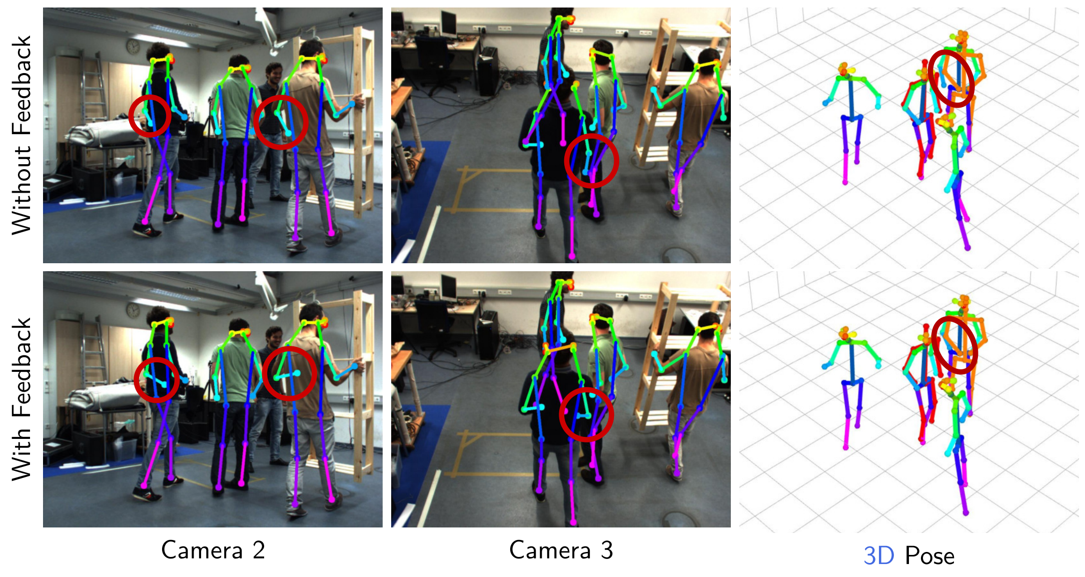

Figure 2.9: Evaluation on Shelf dataset: 2D pose detections and estimated 3D pose without (top) and with feedback (bottom). 3D annotations for Actor 1 (red) and Actor 2 (orange). Highlighted are improvements due to the feedback signal.

2.6.3.2 *Quantitative Results*

We report evaluation results of PCP score and run time on the Campus and Shelf datasets in Tab. 2.6 and compare our method with other approaches: Belagiannis et al. (2015) proposed an early 3D PSM-based method for multi-person pose estimation and exploit temporal consistency in videos. J. Dong et al. (2019) reduce the PSM state-space by using only clusters of 2D joints as 3D proposals instead of a volumetric grid, and exploit appearance information for data association. Long Chen et al. (2020) propose a fast, iterative triangulation scheme performing data association in 3D space.

In terms of PCP score, our method largely outperforms the older method (Belagiannis et al., 2015) and is on par with the recent approaches (Long Chen et al., 2020; J. Dong et al., 2019). The overall result is improved by our method using feedback for both Campus and Shelf dataset in comparison to the literature. The improvement is most significant for Actor 2 of the Shelf dataset, whose arms are often severely occluded, which can be resolved by the semantic feedback signal.

In terms of processing speed, our method does not reach the high frame rates of Long Chen et al. (2020) but achieves significant improvements over J. Dong et al. (2019) and Belagiannis et al. (2015). Furthermore, 2D poses are estimated online, at run times of 30-40 ms per frame, while other methods use offline pre-computed keypoint detections. Our method is the only approach in the comparison providing a fully online multi-person pose estimation.

2.6.3.3 *Qualitative Results*

Fig. 2.9 shows an exemplary scene of the Shelf dataset. The proposed semantic feedback improves the estimation of occluded wrist joints in 2D and 3D. Annotations for evaluation are only provided for two of the four actors in this scene.

Table 2.7: Evaluation in own experiments with up to 16 cameras and 6 persons: Reprojection error (px) per joint class between detected 2D poses and fused 3D poses.

Feedback	Cams	Pers	Hips	Knees	Ankls	Shlds	Elbs	Wrists	**Avg**
w/o fb	4	1-4	5.4	4.6	5.0	2.8	4.0	5.2	4.2
w fb	4	1-4	4.4	**3.5**	**3.4**	**2.3**	**3.2**	**3.7**	**3.3**
w/o fb	16	6	5.4	5.2	6.4	3.9	5.1	6.4	5.1
w fb	16	6	**4.3**	3.8	4.7	3.4	4.0	4.9	4.1

2.6.4 *Experiments in Multi-Person Scenes*

We further evaluate the proposed framework in online experiments in multi-person scenarios in our lab, a large room with an area of ~240 m^2 and a height of 3.2 m. As the room is partly a robotics workshop and partly a desk-based workspace, it is densely filled with different objects and furniture, which can cause false detections and occlusion, providing a challenging real-world setting.

2.6.4.1 *Implementation Details*

16 sensor boards are mounted under the ceiling of the lab in a roughly 12×16 m area. The cameras face downwards towards the center and run at 30 Hz and VGA resolution. We conduct experiments with a subset of 4 cameras, similar to the setting of the H3.6M dataset, as well as with all 16 sensors to demonstrate the scalability of our method to large-scale camera systems. The same detection and pose estimation models as for the Campus and Shelf dataset are employed in the experiments and the pipeline runs in multi-person mode (cf. Sec. 2.4.5).

2.6.4.2 *Quantitative Results*

To analyze the consistency of the online pose estimation, we evaluate the error between detected 2D poses and fused 3D poses reprojected into the camera views in Tab. 2.7. The reprojection error decreases for all joints when using semantic feedback, indicating that the locally estimated 2D poses are more consistent with the globally fused 3D poses through the proposed feedback architecture. The error is slightly higher with 16 than with 4 sensors, probably due to the more difficult camera calibration and synchronization in the large-scale setup.

2.6.4.3 *Qualitative Results*

Exemplary real-world scenes from the experiments conducted in our lab are shown in Fig. 2.10 for the experiment with four cameras and in Fig. 2.11 for the large-scale experiment with 16 cameras. The camera views contain complex scenes with cluttered background and up to six persons.

Estimated 3D poses are reprojected onto the corresponding images to provide a visual evaluation. The reprojected skeletons closely fit the persons in the images, indicating that 3D and 2D poses are reliably estimated. Even large occlusions by objects (i.e. the

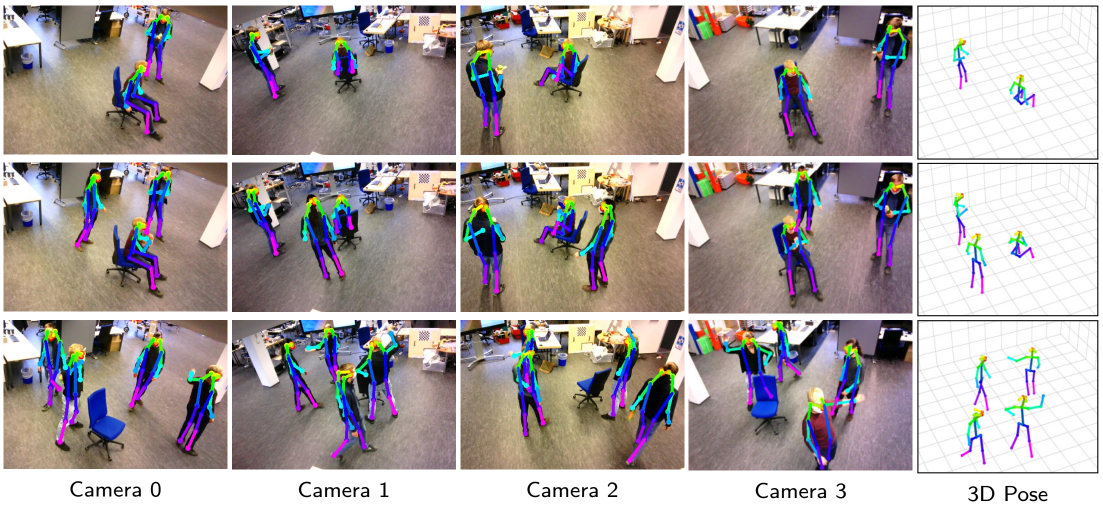

Figure 2.10: Evaluation in multi-person scenarios with four cameras: Estimated 3D poses reprojected into the corresponding camera images for three different scenes with different numbers of persons.

chair, Fig. 2.10, Camera 1, Rows 1, 2) or by other people (Fig. 2.10, Camera 2, Row 3) can be resolved through the multi-view architecture and the proposed semantic feedback. Persons are reliably detected also at large distances to the cameras (Fig. 2.11).

The human poses are estimated online, in real time, and could directly be used, e. g. for human-robot interaction or augmented reality scenarios. Note, that the camera images are not transmitted during operation of our framework but are only shown for visualization. A video of the experiments is available on our website[2].

2.6.4.4 *Run Time Analysis*

The average processing time per image crop on the sensor boards consists of 4.5 ms for pose estimation on the TPU and 6 ms on the ARM-CPU for pre- and post-processing and sums to 10.5 ms per detected person. Once per second, the person detector requires additional 20 ms on the TPU. Up to three persons can thus be tracked at the full camera frame rate of 30 Hz, six persons still at 15 Hz.

The backend processing on a desktop PC with an Intel i9-9900K CPU takes 10.7 ms in average per frameset for the 4-camera setup and 60.8 ms during the experiments with 16 cameras and six persons. Especially the computational load of multi-view triangulation grows with larger number of cameras.

Camera images and semantic feedback are processed asynchronously on the sensors during the online experiments, the frequencies of the feedback and forward parts of the pipeline do not need to match. The most recent feedback message not older than a threshold is used for a camera image. The average pipeline delay Δt including processing on sensors and backend as well as network and synchronization delays sums to 89 ms in the 4-camera setup and to 200 ms with 16 cameras. This delay does not limit the feed-forward pose inference frequency due to the asynchronous parallel processing. The latency is compensated by the prediction step in the feedback channel (cf. Sec. 2.4.4).

2 https://www.ais.uni-bonn.de/videos/RSS_2021_Bultmann

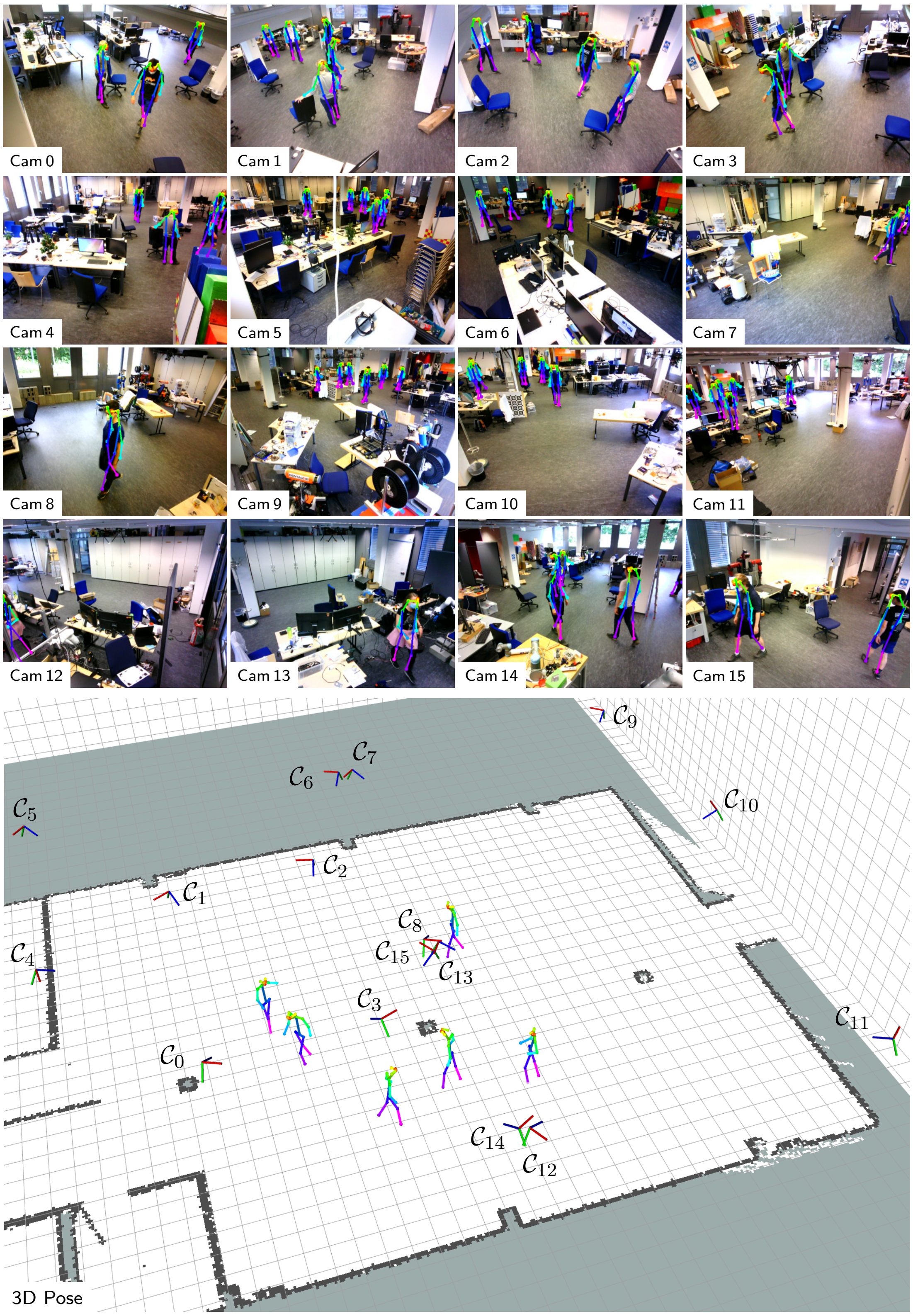

Figure 2.11: Evaluation in multi-person scenarios with 16 cameras: Estimated 3D poses reprojected into the corresponding camera image for the experiment with six persons.

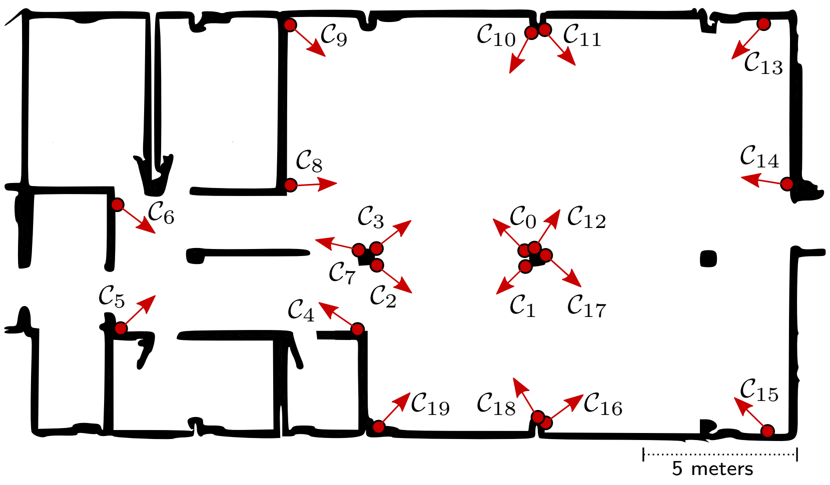

Figure 2.12: Sketched floor plan with camera poses for evaluation of extrinsic camera calibration.

2.6.4.5 *Network Bandwidth and Power Consumption*

The required network bandwidth when processing a 30 Hz video stream only amounts to 15 kB/s per detected person, as only semantic skeletons are transmitted between sensors and backend. This is an over 99 % reduction compared to 27 MB/s when transmitting the raw VGA images. The power consumption of a sensor board was measured as approx. 7 W when running inference on the 30 Hz multi-person video stream.

2.6.5 *Evaluation of Camera Calibration*

To evaluate the proposed method for extrinsic camera calibration, we extend the camera network with four new sensor nodes and change the positions of several sensors: The indices of the cameras from Sec. 2.6.4 are offset by 4, and the new cameras $\mathcal{C}_0$–$\mathcal{C}_3$ are mounted to the columns in the middle of the room. Cameras $\mathcal{C}_4$–$\mathcal{C}_7$ are moved to additionally cover the entry hall of the lab (left part in Fig. 2.12) and $\mathcal{C}_{19}$ is moved to the bottom left corner. The resulting updated positions and viewing directions of the $N = 20$ cameras are illustrated in Fig. 2.12. All cameras are mounted at ~2.6 m height.

For evaluation, we apply our pipeline to recordings of one and two persons (1.96 m resp. 1.70 m tall) crossing the room and generating detections in all cameras over a duration of ~180 s. We repeat the experiment 10 times with different initializations and compare our results towards a marker-based reference calibration obtained using the kalibr toolkit (Rehder et al., 2016) and an AprilTag target.

We apply an initial error of a magnitude of 0.25 m and 10° in a random direction w.r.t. the reference calibration for all cameras $\mathcal{C}_i$ for $i > 0$ and use default parameters provided in the linked repository[3]. We empirically verified that errors of this order of magnitude are easily attainable via manual initialization utilizing a floor plan, height measurements, and RGB images from all cameras.

Tab. 2.8 shows the statistics of the final position and orientation error distributions towards the reference calibration averaged over all repetitions of the experiments with one or two persons present in the scene, respectively. The position error is obtained by rigid alignment of the calibration result towards the reference according to Umeyama's

3 `https://github.com/AIS-Bonn/ExtrCamCalib_PersonKeypoints`

Table 2.8: Statistics of the position and orientation error towards the reference calibration averaged over 10 repetitions of the experiment.

	1 Person				**2 Persons**			
Error	**Avg.**	**Std.**	**Min.**	**Max.**	**Avg.**	**Std.**	**Min.**	**Max.**
Position	0.053 m	0.030 m	0.011 m	0.119 m	0.052 m	0.030 m	0.017 m	0.122 m
Orientation	0.390°	0.184°	0.120°	0.891°	0.436°	0.177°	0.154°	0.818°

Table 2.9: Position error towards the reference calibration for different initial errors.

Initial Error	**0.10 m, 5°**	**0.25 m, 10°**	**0.35 m, 15°**	**0.50 m, 20°**	**0.75 m, 20°**
Final Error	0.050 m	0.052 m	0.067 m	0.118 m	0.214 m
Std.	0.003 m	0.030 m	0.011 m	0.073 m	0.164 m

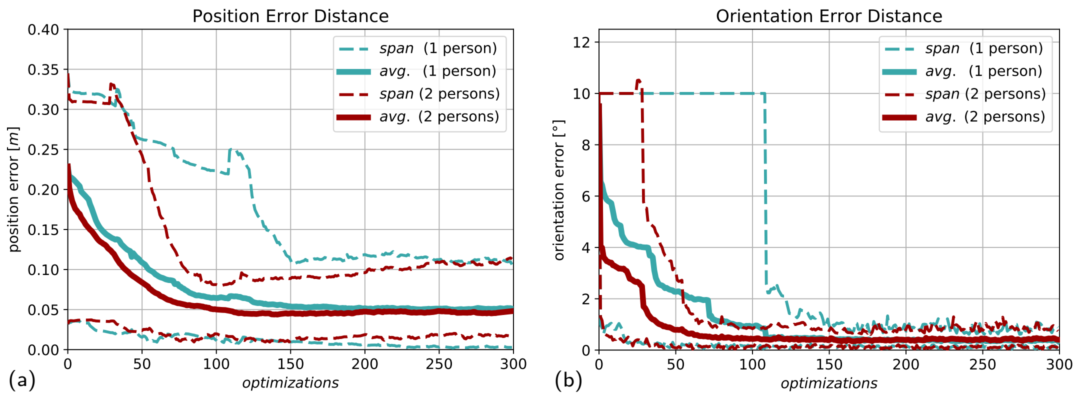

Figure 2.13: Evolution of mean and min–max span of (a) position and (b) orientation error towards the reference. Convergence is faster when observing multiple persons.

method (Umeyama, 1991) without rescaling. The orientation error is computed as the angle between two orientations via the shortest arc (Huynh, 2009).

We do not observe a significant difference in the final result between calibrating with one or two persons. However, convergence is faster in the two-person case, as all cameras provide detections earlier in the procedure.

Fig. 2.13 shows the evolution of the error over time for one exemplary repetition of the resp. experiment with one or two persons. The majority of the convergence takes place in the first ∼50 optimization cycles or ∼35 s and after ∼100 optimizations, the camera poses and errors remain stable in the two-person experiment. With only a single person, convergence is slower. Observations from all cameras are obtained after ∼110 optimization cycles and it takes ∼150 iterations for the poses to remain stable.

Tab. 2.9 shows the final position error for different initialization errors with two persons. Convergence remains stable for initial errors up to 35 cm and 15° but becomes less reliable for larger errors. In particular, the likelihood of the camera poses being stuck in a local minimum consistent with queued person hypotheses containing false data association increases with larger initialization errors, as the accuracy of the data association relies on the geometry of the provided initialization.

Table 2.10: Comparison of the average reprojection error of our method and the reference calibration for keypoint- and AprilTag-based evaluations, averaged over 10 repetitions.

Calibration	Keypoints	AprilGrid
Reference	4.57 px	**1.95 px**
Our Method	**4.01 px**	5.00 px

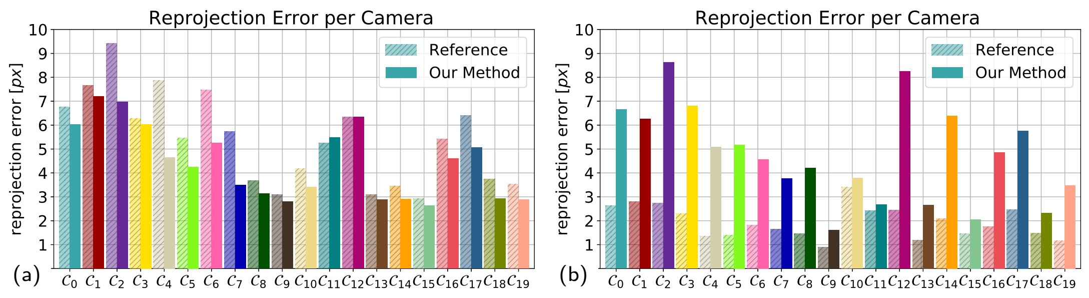

Figure 2.14: Comparison of the reprojection error per camera between our method and the reference calibration using (a) keypoint- and (b) marker-based evaluation pipelines.

Additionally, we compare reprojection errors, measured using two different evaluation pipelines, using the calibration obtained from our experiments with two persons. The first evaluation pipeline processes keypoint detections for 3D human pose estimation (cf. 2.4) and matches the data domain from which our calibration was obtained. We use a distinct recording for the evaluation, unseen during the calibration.

The second pipeline uses a sequence of multi-view images of the AprilGrid used to obtain the reference calibration (Rehder et al., 2016), thus matching its data domain. In general, the keypoint-based evaluation is biased towards our keypoint-based calibration method, while the AprilTag evaluation is biased towards the reference calibration.

Fig. 2.14 shows the reprojection error per camera for both evaluation pipelines and Tab. 2.10 reports the averaged reprojection error. For the keypoint-based evaluation, our calibration method achieves lower reprojection errors for all but two cameras. For the marker-based evaluation, our calibration method achieves similar reprojection errors as for the keypoint-based pipeline, while the reprojection errors of the reference calibration are significantly lower.

Our flexible, marker-free method achieves lower reprojection errors for the envisaged application of 3D multi-person pose estimation and still achieves a coherent result when evaluating with a traditional calibration target. The difference in accuracy for the second evaluation is mainly due to our method being marker-free using features from persons of unknown dimensions for calibration, while the reference method knows the exact scale of the calibration (and evaluation) target. Also, the noise in the joint detections may be larger than for the tag detections.

The averaged reprojection error per joint group for the keypoint-based evaluation is shown in Tab. 2.11. Our method achieves lower reprojection errors in all categories. The reprojection error is larger for faster-moving joints like ankles and wrists, while it is smaller for more stable joints. This can be explained by limitations in the synchronization within framesets.

Table 2.11: Comparison of the reprojection errors per joint group between our method and the reference calibration averaged over 10 repetitions of the experiment.

Calibration	Head	Hips	Knees	Ankles	Shlds	Elbows	Wrists
Reference	4.02 px	5.10 px	4.83 px	5.88 px	3.61 px	4.29 px	5.14 px
Our Method	**3.55 px**	**4.28 px**	**4.20 px**	**5.27 px**	**3.21 px**	**3.75 px**	**4.55 px**

It is worth noting that the measured reprojection error does not exclusively originate from the provided extrinsic calibration, but also from other factors, e.g. the intrinsic camera calibration, or the approach for detection, data association, and triangulation.

2.7 Discussion

In this chapter, we presented a novel method for real-time 3D human pose estimation using a network of smart edge sensors, together with a marker-free online method for the extrinsic calibration of the camera network.

Our main idea is to process each camera view locally, on-board the embedded sensor boards, and transmit only semantic person keypoint detections to a central backend where they are fused into 3D skeleton models or used to update the camera poses. The communication only on a semantic level drastically reduces the required network bandwidth and mitigates privacy issues, as the abstract semantic information contains no personal details.

The smart edge sensor network can be operated either in 3D human pose estimation mode, where the camera poses are assumed to be fixed and known, and the goal is to estimate and track the poses of all persons present in the scene, or in calibration mode, where the person keypoint detections are used as calibration targets to update and refine the camera poses from a rough initial estimate. With this approach, the extrinsic camera calibration can efficiently be updated before every new set of pose estimation experiments, to compensate for environmental effects such as vibration, thermal expansion, or moving parts, or when sensors are added or moved.

For 3D human pose estimation, sets of corresponding keypoint detection messages are synchronized based on the detection timestamps, associated across camera views to person hypotheses based on the epipolar distance of their joints, and raw 3D poses are recovered via triangulation. The 3D poses are further refined using a skeleton model that incorporates prior information on the typical bone lengths of the human skeleton.

We implement a 3D / 2D semantic feedback channel that lets the local semantic models of individual sensors incorporate fused multi-view information, enabling bidirectional communication between sensors and backend.

The pipeline is able to track up to three persons at 30 Hz and up to six persons at 15 Hz, achieving real-time performance. It is evaluated on the H3.6M, Campus, and Shelf datasets where it achieves state-of-the-art results, as well as on own data in challenging real-world scenarios with up to 16 cameras and six persons.

For camera calibration, factor graph optimization problems are repeatedly solved to estimate the camera poses constrained by the synchronized sets of person keypoint observations. The camera poses obtained from the optimization of one factor graph are used to construct the next factor graph, allowing the accumulation of knowledge and the convergence of all cameras to an accurate pose. Finally, the convergence behavior is improved by a refinement scheme based on a Kalman filter.

Our calibration method is designed to be robust against false or sparse sets of detections and occlusions, and is free of many typical assumptions of similar methods: It does not require a specific calibration target, can cope with and exploit the detections of multiple persons simultaneously, and handles arbitrary person poses.

We evaluate the calibration method in a series of experiments and compare our results to a reference calibration obtained by an offline method based on traditional calibration targets. We show that our calibration results are more accurate than the reference calibration by reliably achieving lower reprojection errors in the 3D multi-person pose estimation pipeline used as application scenario. Not only does our method provide a quick and easy-to-use calibration utility, but it also achieves state-of-the-art accuracy.

While we show in the evaluation that our methods perform well in various challenging real-world scenarios, there also are limitations: The embedded inference accelerator of the smart edge sensor boards is very space- and energy-efficient, but has severely limited computational capabilities. This restricts the use of higher resolution images or more powerful network architectures that could improve the keypoint detection accuracy, especially for farther away persons. Also, the processing speed and synchronization accuracy will drop when many persons are present in the scene.

The 3D human pose estimation could use a more elaborate motion model in the prediction step when generating the semantic feedback on the central backend to compensate better for the pipeline delay and improve the feedback signal, especially for fast motions. Furthermore, data association in multi-person scenes currently only relies on geometric cues and could be improved by using visual re-id descriptors, especially during the calibration procedure where it needs to be robust to inaccurate initial estimates of the extrinsic calibration.

The calibration method inherently cannot resolve the scale of the extrinsic calibration, as the dimensions of the persons used as calibration targets are unknown. The scale of the initial estimate of the extrinsic calibration is maintained throughout the calibration procedure but can be erroneous in case the initialization is biased or inaccurate. The calibration method could be extended by including intrinsic camera parameters in the optimization which are currently assumed to be known. Finally, the approach could be improved by also using additional environment features for calibration.

In follow-up research work, we extend the semantic scene model to also include objects and 3D scene geometry (Chapter 3) and include mobile robots carrying a smart edge sensor in the network for collaborative perception to add further viewpoints (Chapter 5). The estimated human pose information then enables safe human-robot interaction and anticipatory robot behavior in a workspace shared with people.

3 3D Semantic Scene Perception

Preface

This chapter is adapted from Bultmann and Behnke (2022), previously published by Springer and presented at the 17th International Conference on Intelligent Autonomous Systems (IAS 2022); Zappel, Bultmann, and Behnke (2021), previously published by Springer and presented at the 24th RoboCup International Symposium; and Hau, Bultmann, and Behnke, 2022, previously published by IEEE and presented at the 6th IEEE International Conference on Robotic Computing (IRC 2022)

Statement of Personal Contribution

The author of this thesis substantially contributed to all aspects of the publication (Bultmann and Behnke, 2022), including the literature survey, the conception, formalization, design, and implementation of the proposed methods, the preparation and conduct of experiments for the evaluation of the proposed approach, the analysis and interpretation of the experimental results, the preparation of the manuscript, as well as the revision and final editing of the version to be published.

The author of this thesis substantially contributed to the following aspects of the publication (Zappel, Bultmann, and Behnke, 2021): the literature survey, the conception, formalization, and design of the proposed methods and experiments, the analysis and interpretation of the experimental results, and the preparation of the manuscript, as well as the revision and final editing of the version to be published. He further provided support for the implementation and evaluation as well as supervision over all other aspects.

The author of this thesis substantially contributed to the following aspects of the publication (Hau, Bultmann, and Behnke, 2022): the literature survey, the conception, formalization, and design of the proposed methods and experiments, the training of the CNN models for pose estimation, the analysis and interpretation of the experimental results, and the preparation of the manuscript, as well as the revision and final editing of the version to be published. He further provided support for the implementation and evaluation as well as supervision over all other aspects.

The content presented in this chapter, unless otherwise stated, is the contribution of the author of this thesis.

Abstract

In this chapter, we extend the network of distributed smart edge sensors to a system for 3D semantic scene perception. New sensor nodes are added to the camera network, based on a more powerful embedded CNN inference accelerator and RGB-D and thermal cameras. Efficient vision CNN models for object detection, semantic segmentation, and

human and object pose estimation run on-device in real time. As the image interpretation is computed locally, only semantic information is sent over the network. The raw images remain on the sensor boards, significantly reducing the required bandwidth, and mitigating privacy risks for the observed persons.

2D human keypoint estimations, augmented with the RGB-D depth estimate, as well as semantically annotated point clouds are streamed from the sensors to a central backend, where multiple viewpoints are fused into an allocentric 3D semantic scene model. Additionally, pose and shape information for detected object instances is computed on the sensor boards and fused on the backend to include object-level information into the semantic map. Using the 3D scene model and a ray-tracing approach, the semantic feedback for human pose estimation (cf. Chapter 2) is extended with occlusion information for each joint so that unreliable, locally occluded joint detections can be improved by the more reliable global context.

Human poses are represented via 3D skeleton models, as in Chapter 2, and the 3D scene geometry via an allocentric volumetric semantic map. Objects are represented in the map via their 3D mesh model or as an object-centric volumetric sub-map that can model arbitrary object geometry when no detailed 3D model is available. We propose a keypoint-based approach for object pose estimation via PnP from 2D-3D correspondences between detected and model keypoints. The object poses are further refined via ICP alignment of the 3D object model with the observed point cloud segments. Object instances are tracked over time to be robust against temporary occlusions.

We evaluate the object pose estimation approach on the YCB-V dataset (Xiang et al., 2018) and the proposed system for 3D semantic scene perception on the Behave dataset (Bhatnagar et al., 2022) for multi-view object pose estimation and in real-world experiments with the sensor network in a challenging lab environment where multiple persons and different furniture objects are tracked through the scene online, in real time even under high occlusions.

The proposed perception system provides a complete scene view containing semantically annotated 3D geometry, 3D object instances with pose and shape information, and 3D poses of multiple persons, estimated in real time.

3.1 Introduction

Autonomous robots that interact with their environment require a detailed semantic scene model to enable applications such as safe and anticipatory robot movement in the vicinity of people, object manipulation, or human-robot interaction. Creating and updating a semantic scene model is a challenging task and requires accurate semantic perception of 3D scene geometry, persons, and objects.

In this work, we propose a system for 3D semantic scene perception consisting of a network of distributed smart edge sensors. The environment is captured with various sensors from different view points and a semantic scene model is created from the interpreted measurements. Our system provides a complete scene view containing semantically annotated 3D geometry, 3D object instances with pose and shape information, and estimates 3D poses of multiple persons in real time.

We build upon our previous work on real-time 3D human pose estimation using semantic feedback to smart edge sensors presented in Chapter 2. While this existing pipeline is able to track poses of multiple persons in real time, it lacks modeling of other

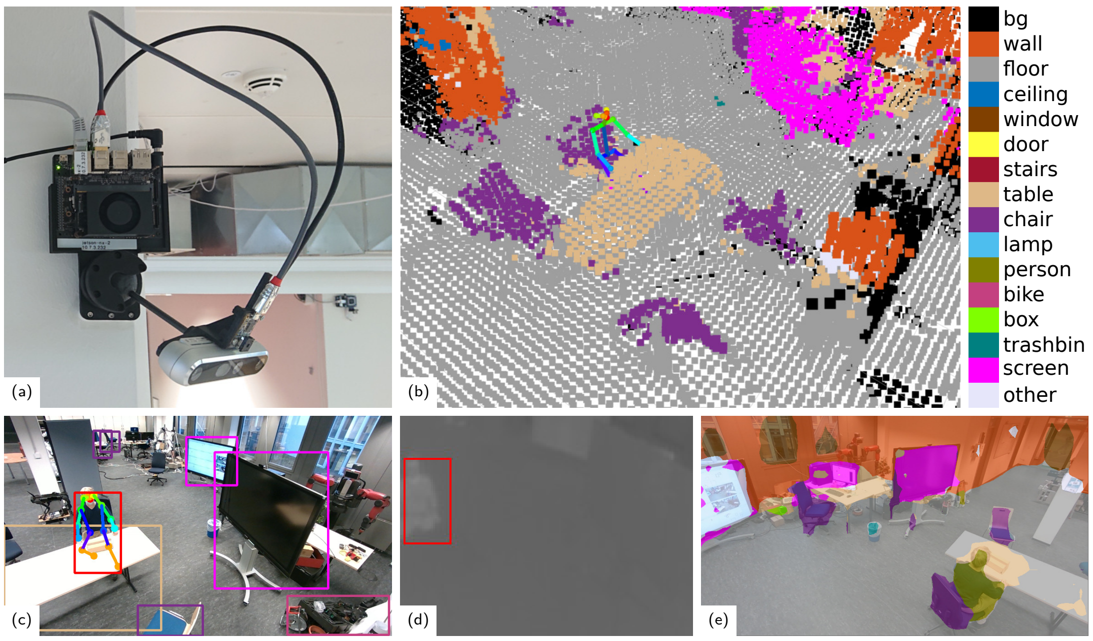

Figure 3.1: Semantic perception with distributed smart edge sensors: (a) developed sensor node, (b) volumetric 3D semantic scene model with 3D human skeleton, (c) RGB and (d) thermal detections, (e) semantic segmentation. Person detections in red and skeleton keypoints colored by joint index. Occluded joints are marked in orange. CNN inference runs online on distributed sensors and semantic information is aggregated into an allocentric 3D scene model on the backend including 3D geometry (e.g., furniture, walls, floor) and 3D human pose.

aspects of the scene, i.e. 3D geometry, object detections, and surface categorization. Semantically annotated 3D geometry, however, is required to explain and predict interactions between persons and objects in the scene, and to handle occlusions. Temporal aggregation and fusion of semantic point clouds from multiple sensor perspectives further leads to a consistent and persistent 3D semantic scene model with the field of perception not limited by the measurement range or occlusions of a single sensor.

To enable perception of these additional characteristics of the scene, the sensor network is extended with updated smart edge sensors with higher compute capabilities and greater flexibility w.r.t. the employed vision CNNs, as shown in Fig. 3.1. This enables to run object detection and semantic image segmentation together with human and object pose estimation on the sensors in real time. RGB-D cameras estimate 3D scene geometry and thermal cameras increase the person detection performance in low-light conditions. Semantic information from detections and image segmentation is fused into the point cloud computed from the depth image. 2D human joint detections are augmented with the depth measured at the keypoint location. Additionally, pose and shape information for object detections of the considered furniture classes (*chair* and *table*) are estimated on the sensor boards. Semantic point cloud, object instance hypotheses, and human poses are communicated to a central backend where they are fused into an allocentric 3D metric-semantic scene model. Only the semantic information is sent over the network; the raw images remain on the sensor boards, significantly reducing the required bandwidth, and mitigating privacy issues for the observed persons.

The semantic point clouds from multiple viewpoints are aggregated into an allocentric map of 3D scene geometry and semantic classes on the backend. The map is further

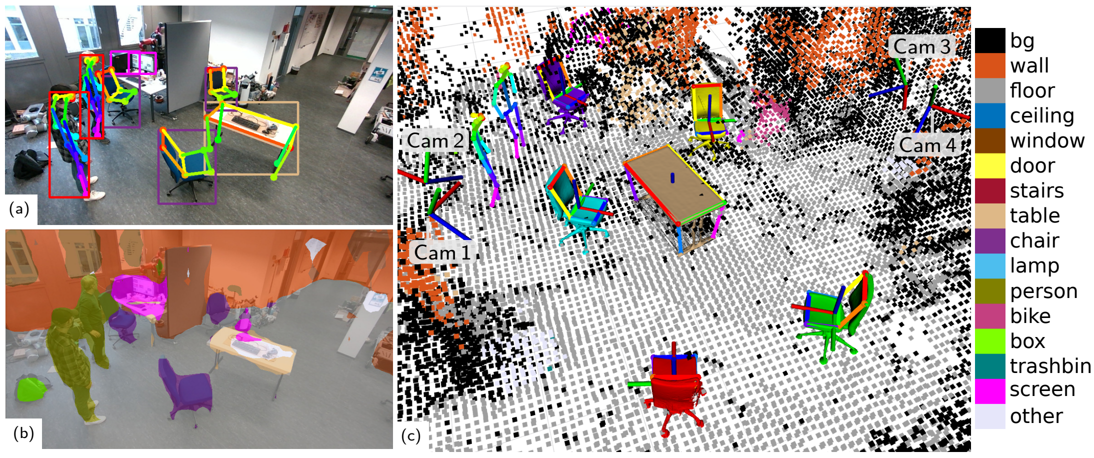

Figure 3.2: Object-level 3D semantic mapping: (a) object detection and keypoint estimation for the *chair* and *table* class; (b) semantic segmentation from an exemplary smart edge sensor view (Cam 2); (c) 3D scene view with five chairs and a table, represented by their resp. 3D mesh and colored by instance ID, and human skeleton models.

updated via ray-tracing to account for moving objects. 3D human poses are estimated in real time via multi-view triangulation. The allocentric 3D human poses are projected into the local camera views and sent back to the sensors as semantic feedback (cf. Chapter 2), where they are fused with the local detections. The 3D scene geometry enables to compute occlusion information for each joint in the respective camera view. This information is included into the semantic feedback from backend to sensors, improving the local scene model of each sensor by incorporating global context information. Unreliable, occluded joint detections can be discarded, and the local model is completed by the more reliable semantic feedback reprojected from the global, fused 3D model.

To include object-level information into the semantic map, object instance hypotheses from the different viewpoints are fused and tracked through the scene model. We represent objects in the map via their 3D mesh model, if available, or via an object-centric volumetric sub-map that can model arbitrary object geometry.

For object pose estimation, we employ a keypoint-based approach using CNNs for keypoint detection trained only on synthetic data obtained through randomized scene generation (Boltres et al., 2022; Schwarz and Behnke, 2020). Object poses are recovered from keypoint detections in each camera view, using known correspondences between the predicted 2D keypoints from the image and the keypoints defined on the 3D object models via a variant of the perspective-n-point (PnP) algorithm (Lepetit, Moreno-Noguer, and Fua, 2008). The PnP object pose estimates are then refined via iterative closest point (ICP) alignment of the 3D object model with the observed point cloud segments. The pose estimates from multiple viewpoints are fused on the backend via weighted interpolation.

Fig. 3.2 illustrates the object-level semantic mapping, showing object detections, keypoint estimation for the *chair* and *table* class, and semantic segmentation from an exemplary smart edge sensor view together with the fused 3D semantic scene model with five chairs and a table represented by their respective 3D mesh tracked in the allocentric semantic map.

In a first set of experiments, the approach for 6 DoF object pose estimation from a single camera view is evaluated on the challenging Yale-CMU-Berkeley-video (YCB-V) dataset (Xiang et al., 2018). Different ways to define keypoints on the object models and to scale the keypoint estimation CNNs to handle multiple object classes are compared.

We then evaluate our method for multi-view object pose estimation and object-level mapping on the public Behave dataset (Bhatnagar et al., 2022), containing various scenes with human-object interactions, and the full system for 3D semantic scene perception in real-world experiments with the sensor network in a challenging, highly cluttered and dynamic lab environment with multi-person scenes.

In summary, the main contributions presented in this chapter are:

- the development of a smart edge sensor platform based on the Nvidia Jetson Xavier NX development kit and an RGB-D and thermal camera, running efficient vision CNN models for object detection and semantic segmentation together with human and object pose estimation on-device in real time,
- temporal multi-view fusion of semantic point clouds from individual sensors into an allocentric semantic map of 3D scene geometry,
- a novel multi-view object-level 3D semantic mapping approach, adding object instance information to the semantic scene model,
- a keypoint-based object pose estimation approach trained solely on synthetic data,
- and the integration of multiple instances of the proposed novel sensor nodes into the network of distributed smart edge sensors for real-time multi-view 3D human pose estimation using semantic feedback, complementing the feedback from backend to sensors with occlusion information for human joints in the respective camera views, computed via ray-tracing through the estimated 3D scene geometry.

3.2 Related Work

Semantic Mapping. Many approaches exist in the literature to create three-dimensional (3D) maps to be used for localization and navigation of mobile robots. A common approach are occupancy grid maps. Octomap (Hornung et al., 2013), a widely-used 3D occupancy mapping framework, uses an efficient octree-based data structure to save occupancy probabilities of the environment divided into discrete volume elements (*voxels*). A second popular map representation are truncated signed distance fields (TSDFs). A TSDF map saves the distance to the closest surface in each voxel. Voxblox (Oleynikova et al., 2017) is a commonly-used framework for building TSDF-based maps.

The above works represent only geometry and don't contain any semantics. Semantic information about the environment, however, is a prerequisite for many high-level robotic tasks. For this, semantic mapping systems build an allocentric model of 3D scene geometry with semantic class information. To extend geometric 3D maps with semantic information, Stückler, Biresev, and Behnke (2012) fuse probabilistic object segmentations from multiple RGB-D camera perspectives into a voxel-based 3D map using a Bayesian framework. SemanticFusion (McCormac et al., 2017) builds semantic maps from RGB-D camera input using surface elements (Surfels), where a Gaussian approximates the local point distribution. Pixel-wise class probabilities are obtained from the color image via semantic segmentation and fused into the map using a Bayesian approach assuming independence of individual measurements. Literature on allocentric semantic mapping is discussed in more depth in Chapter 4 while we focus on object-level mapping in the following.

The scene understanding can further be improved by including object-level information into the semantic map through explicit modeling of object instances. MaskFusion (Rünz, Buffier, and Agapito, 2018) is an RGB-D SLAM system that can reconstruct and track multiple objects in a scene without knowing prior models of the objects. The constructed map uses Surfels to represent surfaces and Mask-RCNN (K. He et al., 2017) to obtain a semantic instance segmentation of the RGB images. MID-Fusion (B. Xu et al., 2019) uses an octree-based TSDF-map to implement RGB-D SLAM. On top of the scene geometry, RGB-color, semantic classes, and a foreground probability are represented in the map. Voxblox++ (Grinvald et al., 2019) uses both geometric and semantic instance segmentation to build a semantic map with object-level information for static scenes. With TSDF++, Grinvald et al. (2021) propose to create TSDF sub-volumes for object instances. The object sub-volumes are included in an allocentric volumetric map that can reference multiple objects at each location. Thus, temporally occluded objects or surfaces remain in the map and do not need to be reconstructed anew when they become visible again, and dynamic scenes can be represented. Dengler et al. (2021) proposed an object-centric 2D/3D map representation for real-time service robotics applications, using RGB-D data as input. A geometric segmentation of small objects in the point cloud is obtained via Euclidean clustering.

While the above works use a single, moving camera, we propose to fuse percepts from multiple static viewpoints to create a voxel-based semantic map of static scene geometry that includes object-centric sub-maps or mesh models for tracked object instances.

Lightweight Vision CNNs for Embedded Hardware. Convolutional neural networks (CNNs) set the state-of-the-art for image processing and computer vision. However, on systems with limited computational resources, such as mobile embedded sensor platforms, lightweight, efficient models must be employed to achieve real-time performance. A popular approach is to replace classical backbone networks such as ResNets (K. He et al., 2016) with MobileNet (Howard et al., 2019; Sandler et al., 2018) or EfficientNet (Tan and Le, 2019) architectures, as the main computational load of CNN inference often lies in the backbone feature extractor. These architectures decrease the number of parameters and the computational cost significantly, e. g. by replacing standard convolutions with depthwise-separable convolutions.

For object detection on embedded devices, single-stage architectures such as SSD (W. Liu et al., 2016) or YOLO (Redmon et al., 2016), which use predefined anchors instead of additional region proposal networks, have proven to be efficient. In our work, we employ the recently proposed MobileDets (Xiong et al., 2021), that are optimized for embedded inference accelerators using the SSD architecture with a MobileNet v3 backbone.

The DeepLab v3+ architecture (L.-C. Chen et al., 2018) for semantic segmentation leverages elements of MobileNets, such as depthwise-separable convolutions, for efficiency on embedded hardware, and shows state-of-the-art performance on large, general datasets. We employ a DeepLab v3+ model with a MobileNet v3 backbone in our work.

For human pose estimation, OpenPose (Cao et al., 2021) set a new standard by detecting body parts of multiple persons in an image and associating them to individuals via part affinity fields (PAFs). This keypoint-based approach can also be generalized to object pose estimation by defining distinct points on the respective object model that are then detected in the camera image. The assignment of detected keypoints to person or object instances is commonly implemented in two different manners: bottom-up or top-down. The bottom-up approaches scale well with the number of persons or objects present in the image but can wrongly associate keypoints of different instances and have difficulties handling detections of small scales. Top-down approaches, on the other hand, first detect instances and then estimate body or object keypoints for each single-instance crop. These approaches achieve higher accuracy and better scale invariance, as the detections are interpolated to a fixed input resolution before pose inference. However, they have the risk of early commitment due to errors in person or object detection and scale linearly with a higher number of detected instances.

Xiao, Wu, and Wei (2018) propose an efficient CNN architecture for 2D human pose estimation consisting of a backbone feature extractor and deconvolutional layers. We adopt this architecture for human and object keypoint estimation, replacing the ResNet backbone with MobileNet v3 for better efficiency on the embedded hardware.

6 DoF Object Pose Estimation. The task of 6 degrees of freedom (DoF) object pose estimation consists of detecting known objects and estimating their orientation and translation in 3D space from a single RGB image. 6 DoF object pose estimation methods from the literature can roughly be divided into two classes.

Direct methods infer pose parameters directly from the image (Kendall, Grimes, and Cipolla, 2015; Xiang et al., 2018). Xiang et al. (2018) define a CNN architecture, called *PoseCNN*, that segments objects from 2D images, predicts their depth, and regresses the 6 DoF pose parameters. Capellen, Schwarz, and Behnke (2020) extend this approach to a fully convolutional network for dense prediction of pose parameters not depending on prior object segmentation. Amini, Periyasamy, and Behnke (2021) propose a transformer-based architecture for multi-object direct 6 DoF pose regression. However, direct pose regression is a difficult task—especially for the rotation parameters as the 3D rotation space is highly nonlinear.

In contrast, keypoint-based approaches adopt a two-step pipeline: First, 2D keypoints are predicted for each object instance in the image and the 6 DoF pose is computed in a second step from 2D-3D correspondences with a variant of the PnP-Algorithm (Lepetit, Moreno-Noguer, and Fua, 2008). Approaches mainly differ in how the keypoints are defined on the object model and how they are inferred from the image. In BB8 (Rad and Lepetit, 2017), a segmentation mask is computed for each object and the keypoints are then inferred as the eight corners of the 3D object bounding box. The coordinates of the keypoints are directly regressed by the network. The corners of the 3D bounding box, however, often are not located on the object surface and are thus difficult to infer from local object image features. Pavlakos et al. (2017a) define keypoints on the object surface and infer them as maxima of pixel-wise heatmaps. With PVNet, Peng et al. (2019) also define keypoints on the object surface but infer them in a dense manner: Each pixel in the object segmentation mask predicts vectors that point to every keypoint. The keypoint locations are then computed through RANSAC-based voting, choosing locations where the predicted directions intersect. This permits to also represent keypoints that are occluded or outside of the image. Amini, Periyasamy, and Behnke (2022) develop a transformer-based model for object keypoint regression and propose to use interpolated bounding box keypoints, i.e. the corners of the 3D object bounding box and additional interpolated points on the bounding-box edges. They use learnable rotation and translation estimators instead of the analytical PnP algorithm to infer the 6 DoF object pose from the keypoints.

Some approaches apply additional refinement after the initial pose estimation to further improve performance. Cosypose (Labbé et al., 2020), e.g., implements an additional network that refines a given pose with the input image as additional input. When depth information is available, e.g., in the RGB-D variant of PoseCNN (Xiang et al., 2018), ICP can also be used for pose refinement. Periyasamy, Denninger, and Behnke (2022) propose a probabilistic approach to improve the pose estimation of symmetric objects.

In this work, we adopt a keypoint-based approach using keypoints on the object surface. We employ a top-down approach for object keypoint detection on embedded smart edge sensors using the efficient CNN architecture introduced in the previous paragraph. The 6 DoF object poses are recovered using the PnP-RANSAC Algorithm (Fischler and Bolles, 1981; Lepetit, Moreno-Noguer, and Fua, 2008) to calculate the object's translation and rotation in the camera frame from 2D–3D correspondences of keypoints on rigid objects. The PnP pose estimates are further refined via ICP alignment with the observed point cloud segments and pose estimates from multiple sensor views are fused via weighted interpolation.

3D Human Pose Estimation. Literature on 3D human pose estimation was discussed in depth in Chapter 2, where we proposed a pipeline for real-time 3D human pose estimation using multiple calibrated smart edge sensors that perform 2D pose estimation on-device. Semantic pose information is transmitted to a central backend where multiple views are fused into a 3D skeleton via triangulation and an efficient, factor graph-based skeleton model. The fused allocentric 3D joint positions, after motion prediction to compensate for the pipeline delay, are reprojected into local views and sent back to the sensors as semantic feedback, where they are fused with the detected keypoint heatmaps. This enables the sensors to incorporate global context information into their local scene view interpretation. The pipeline delay is estimated as the difference between the timestamps of the current detection and the latest received feedback message on a sensor and updated using a moving average filter.

We build upon this work and extend the sensor network with new smart edge sensor nodes with significantly increased computational power and RGB-D cameras that enable the perception of 3D geometry. In addition to human pose estimation, object detection and pose estimation as well as semantic image segmentation are computed on the sensor boards and fused via 3D projection into a semantic point cloud. Semantic point clouds from multiple sensor views are fused into a sparse voxel hash-map with per-voxel full semantic class probabilities on the backend. Pose and shape information of object instances is included into the map, representing objects through their posed 3D mesh model or an object-centric volumetric sub-map and robustly tracking their movement over time. The semantic map is used to obtain occlusion information for person keypoints in the local camera views, which is added to the semantic feedback to increase the robustness of the human pose estimation pipeline.

Also related to our work are recent studies on tracking human-object interactions from multi-view RGB-D data (Bhatnagar et al., 2022; Huang et al., 2022). Bhatnagar et al. (2022) propose to jointly track humans, objects, and their interactions in real-world environments using multiple RGB-D cameras as input. Humans are represented via the

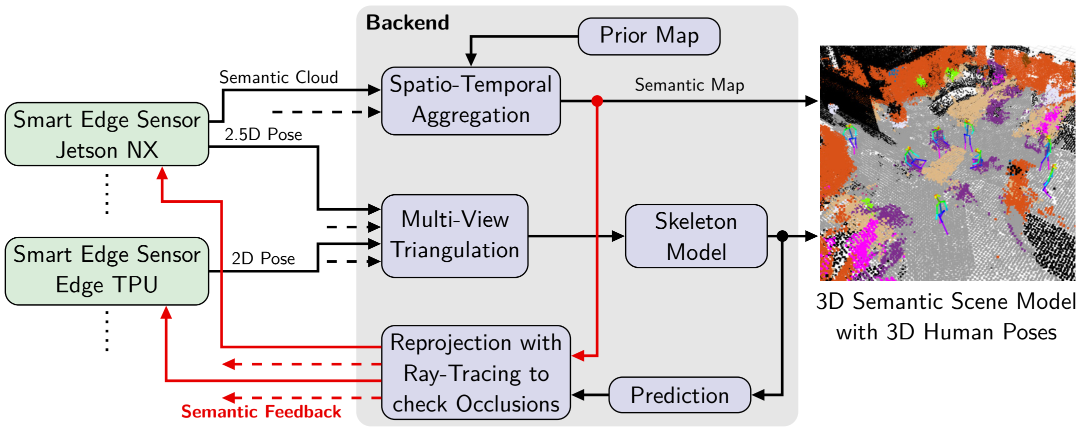
Backend
Prior Map
Semantic Cloud
Spatio-Temporal Aggregation
Semantic Map
Smart Edge Sensor Jetson NX
2.5D Pose
Multi-View Triangulation
Skeleton Model
Smart Edge Sensor Edge TPU
2D Pose
3D Semantic Scene Model with 3D Human Poses
Reprojection with Ray-Tracing to check Occlusions
Prediction
Semantic Feedback

Figure 3.3: Overview of the multi-sensor pipeline for 3D semantic mapping and human pose estimation: The Jetson NX smart edge sensors extend the sensor network of nodes with lower compute capabilities (cf. Chapter 2). Semantic point clouds from multiple sensor views are aggregated into an allocentric 3D semantic map and 3D human poses are estimated in real time. The map is used to check reprojected joints for occlusion in the resp. sensor view via ray-tracing and this information is added to the semantic feedback sent to the smart edge sensors.

parametric SMPL 3D body mesh model (Loper et al., 2015), objects via their pre-scanned 3D mesh model, and interactions as surface contacts. Huang et al. (2022) additionally provide detailed hand and finger poses, using the SMPL-X body model (Pavlakos et al., 2019), and focus on realistic hand-object contacts. These methods implement a centralized, offline multi-view fusion, they do not run inference in real time on distributed sensors. Their focus is on a single human interacting with a single object, but not on building a complete semantic scene model.

3.3 Method

Figure 3.3 illustrates the proposed multi-sensor pipeline for 3D semantic scene perception and human pose estimation combining two types of smart edge sensors. The proposed Jetson NX sensors are integrated into the sensor network from Chapter 2, consisting of nodes based on the Google Edge TPU with lower compute capabilities and RGB image-only 2D human pose estimation, without local depth estimation. We consider a calibrated camera network, with known projection matrices from sensor to world coordinates, where the sensors are software-synchronized via the network time protocol (NTP). Semantic mapping is only performed with the newly added Jetson NX sensor nodes, while data from both sensor types is combined for 3D human pose estimation.

An overview of the proposed approach for semantic perception onboard each Jetson NX smart edge sensor is given in Fig. 3.4. We detail individual components of the data processing on each sensor board, as well as the fusion of multiple sensor views for 3D semantic mapping and 3D human pose estimation in the following. The semantic mapping and point cloud branches in Figs. 3.3 and 3.4 illustrate the allocentric volumetric semantic mapping, while object pose estimation and object-level mapping are detailed in Sec. 3.3.4 and Sec. 3.3.6.

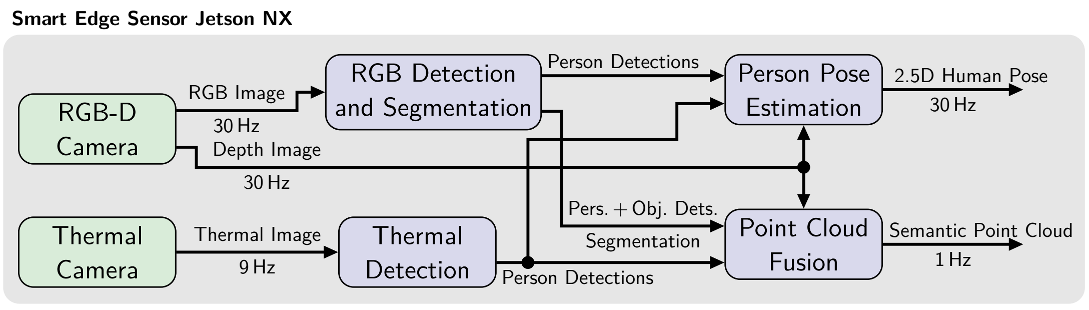

Figure 3.4: Smart edge sensor semantic perception system overview. Human poses are estimated in real time, while the semantic point cloud of the static or slowly moving scene geometry is output at a lower frequency to save compute resources.

3.3.1 *Smart Edge Sensor Hardware*

We developed smart edge sensors based on the Nvidia Jetson Xavier NX developer kit[1] (cf. Fig. 3.1 (a)), equipped with a 6-core ARM processor, 384 CUDA cores, and 8 GB of RAM. The Jetson NX embedded system achieves a CNN inference performance of 21 trillion operations per second (TOPS), a significant increase compared to the 4 TOPS of the sensor platform employed in Chapter 2. For visual perception, we connect an Intel RealSense D455 RGB-D camera and a FLIR Lepton 3.5 thermal camera to the Jetson NX board.

3.3.2 *Single-View Embedded Semantic Perception*

PERSON AND OBJECT DETECTION. We employ the recent MobileDet architecture (Xiong et al., 2021) for person and object detection. The RGB detector is trained on the COCO dataset (Lin et al., 2014) using the *person* and 12 indoor object classes (e.g., *chair*, *table*, *screen*), with an input resolution of 848×480 px. The same network architecture is used for the thermal detector, taking one-channel 8-bit thermal images at the camera resolution of 160×120 px as input. The thermal detector is trained only for the *person* class on the ChaLearn IPHD dataset (Clapés et al., 2020).

PERSON AND OBJECT KEYPOINT ESTIMATION. We adopt a top-down approach for person and object keypoint estimation on the smart edge sensors, where crops of single persons or objects are analyzed by the keypoint estimation CNN. These crops are extracted from the input RGB image using bounding boxes obtained through the detection network introduced above. The CNN architecture of Xiao, Wu, and Wei (2018) is the basis of our keypoint estimation, but we exchange the ResNet backbone with the significantly more lightweight MobileNet v3 feature extractor (Howard et al., 2019). This architecture proved efficient on embedded hardware in prior work (Chapter 2). We train keypoint estimation networks for human pose estimation using the person keypoint annotations of the COCO dataset (Lin et al., 2014) and for object pose estimation for chairs and tables using synthetic training data (see Sec. 3.3.4).

Person detections from RGB and thermal images are forwarded to the person keypoint estimation CNN. The RGB-D depth thereby is used to project detections from the

1 https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit, accessed: 2023-08-01

thermal camera to the color image. Redundant detections of the same person in both modalities are filtered via non-maximum suppression (NMS). Each person crop is then resized to the fixed 192×256 input resolution of the keypoint estimation CNN and inference is run for all crops together in batched mode. Batch processing gives a significant improvement in the scaling of inference time with the number of persons compared to previous work, where the embedded hardware only supported processing a single crop at a time (cf. Sec. 2.6.4.4 and Sec. 3.4.3.4).

The pose estimation model outputs multi-channel images, called *heatmaps*, encoding the confidence of a joint being present at the pixel location. As single-person crops are processed, 2D joint locations are determined as global maxima of the respective heatmap channel. The RGB-D range image is used to augment the 2D keypoints to a 2.5D pose representation. For each joint, the median depth of a 5×5 px region around the joint location is obtained from the depth image. The local depth estimate enables the projection of keypoints into three-dimensional space but often suffers from noise and occlusions, as is further analyzed in Sec. 3.3.7. The 2.5D pose estimate for each detected person is sent to a central backend, where multiple sensor views are fused into a coherent 3D pose representation. The person pose estimation pipeline runs with the highest real-time priority on the sensor boards, to enable tracking of dynamic human motions. To save computational resources, the detector is run only once per second and the crops are updated based on the keypoint estimations between detector runs.

Similarly, for object keypoint estimation, chair and table detections from the RGB image are forwarded to the respective object keypoint estimation CNN. Object keypoints, together with corresponding point cloud segments obtained from the depth data, are used for object pose estimation as detailed in Sec. 3.3.4. Object pose estimates from multiple views are sent to a central backend, where multiple viewpoints are fused and the object movement is tracked through the allocentric map. To limit computational resource usage as the motion of the considered furniture object classes is significantly less dynamic than human motions, object keypoint estimation is only executed once per second, together with the detector runs.

SEMANTIC SEGMENTATION. We adopt the DeepLab v3+ (L.-C. Chen et al., 2018) architecture with MobileNet v3 (Howard et al., 2019) backbone for semantic segmentation. We train the model on the indoor scenes of the ADE20K dataset (Zhou et al., 2019) and reduce the labels to the 16 classes most relevant for the intended indoor application scenarios (cf. Fig. 3.1). The input image size is set to 849×481 px, fitting the 16:9 aspect ratio of our camera. The semantic segmentation is run only once per second, similar to the person and object detector, to save computational resources.

3.3.3 *Semantic Point Cloud Fusion*

We obtain a geometric point cloud by projecting the RGB-D range image into 3D. The point cloud is uniformly subsampled using a voxel-grid filter with 5 cm resolution to reduce the amount of data, economizing computational resources, and later network bandwidth for transmission to the backend. Sparse outlier measurements are removed by a statistical outlier filter, as implemented in the point cloud library (PCL) (Rusu and Cousins, 2011). A point is deleted when the distance to its neighbors is outside an interval defined by the mean and standard deviation of the entire point cloud.

Semantic information from RGB and thermal detections, as well as RGB semantic segmentation, is fused into the point cloud using a projection-based approach. For this, the points are projected into the segmentation mask inferred from the RGB image. The semantic class scores $\boldsymbol{c}_{\text{segm}} \in \mathbb{R}^C$ are obtained from semantic segmentation via bilinear interpolation at the projected point location. A normalized probability distribution over the employed $C = 16$ classes is then approximated by applying the soft-max operation:

$$p_i = \sigma\left(c_i\right) = \frac{\exp c_i}{\sum_{j=1}^{C} \exp c_j}\,, \tag{3.1}$$

obtaining $\boldsymbol{p}_{\text{segm}} \in \mathbb{R}^C$, with $p_i \in [0,1]$ and $\sum_i p_i = 1$.

If a projected point falls inside a detection bounding box in either thermal or color images, we further fuse the detector result with the semantic segmentation. We reconstruct the detection probability distribution $\boldsymbol{p}_{\text{det}}$ from the score for the detected class following the maximum entropy principle: The probability of the detected class p_{det} is given by the detector score and the remaining probability mass $1 - p_{\text{det}}$ is equally distributed over the remaining $C - 1$ classes. Both estimates are fused following the Bayesian update rule (McCormac et al., 2017), assuming independence of segmentation and detection:

$$\boldsymbol{p}_{\text{fused}} = \frac{\boldsymbol{p}_{\text{segm}} \circ \boldsymbol{p}_{\text{det}}}{\sum_{i=1}^{C} p_{i,\text{segm}}\, p_{i,\text{det}}}\,, \tag{3.2}$$

with $\circ$ the coefficient-wise product. For better numerical stability, we use a logarithmic implementation of the Bayesian fusion (Bultmann, Quenzel, and Behnke, 2023).

As the detection bounding boxes are axis-aligned, border-effects have to be considered for non-rectangular or non-axis-aligned objects before detection fusion. Inclusion of all points projected into the bounding box in the fusion would falsely label points on the ground and in the background as the detected class. To alleviate this issue, the ground plane is removed and the remaining points are clustered in 3D Euclidean space by a distance threshold. Only the clustered points are included into detection fusion. Further details on the clustering approach are given in Sec. 4.3.3.

The output semantic point cloud includes the class probability vector and the argmax class color per point (cf. Fig. 3.7 (a)), and is sent to the central backend over the network. Additionally, points of the *chair* and *table* semantic classes are extracted and geometrically clustered into object instance segments used for object pose estimation together with the object keypoint detections. The semantic point cloud is computed at a reduced update frequency of 1 Hz on the sensors, as it targets the static or slowly moving scene geometry. Thus, computational resources are kept free for the real-time estimation of dynamic human motions in the pose estimation pipeline.

3.3.4 *Keypoint-based Object Pose Estimation*

For keypoint-based pose estimation, keypoint locations need to be defined at distinct points of the object model. We perform keypoint estimation for the *chair* and *table* object classes in this work and aim to represent different types of chairs and tables with the defined keypoints. For this, we define $L_{\text{chair}} = 6$ keypoints on chairs: four keypoints on the corners of the seating and two keypoints at the top of the backrest of a chair. These keypoints can be consistently defined for most types of chairs and

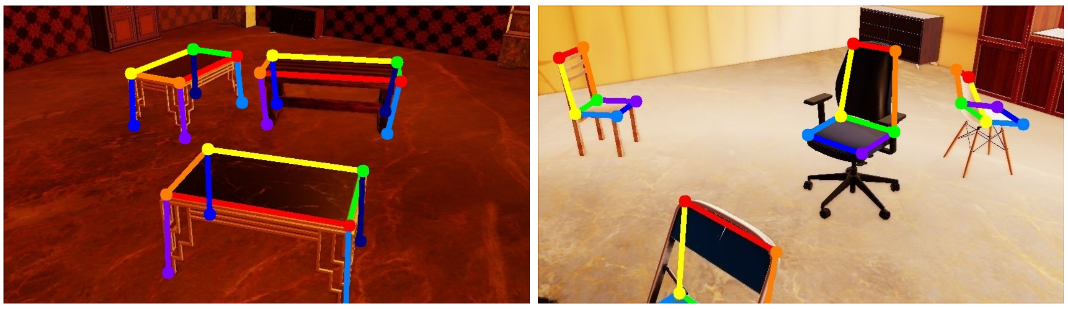
Figure 3.5: Frames from two synthetic training scenes with corresponding ground truth keypoint annotations for the *table* or *chair* class, respectively. Background textures and object models are randomly selected.

have less variance in appearance and geometry than, e.g., armrests or legs of chairs. Similarly, we define $L_{\text{table}} = 8$ keypoints on tables: four keypoints on the corners of the tabletop and four on the points of the table legs (cf. Fig. 3.5).

To avoid costly manual annotation of training data and to facilitate generalization to different object classes, we only use synthetic training images for the keypoint estimation. We employ the *sl-cutscenes* framework (Boltres et al., 2022), an extension of the *stillleben* framework (Schwarz and Behnke, 2020), for randomized photorealistic indoor scene generation with physically interacting objects. We create a dataset of ~13k training and ~2.5k validation images per object class where between three and six randomly selected chairs or tables move around a room with randomly selected textures and background objects. Fig. 3.5 shows samples of the generated training images.

The detected object keypoints are used in a second step to recover the object's translation and rotation in the camera coordinates via the PnP-RANSAC algorithm. For this, we assume a 3D model of the specific type of chair or table visible in the respective scene to be available and exploit the known correspondences between 2D image keypoint and keypoints defined on the 3D object model. A 3D object skeleton in camera coordinates is then obtained by transforming the object model keypoints with the estimated PnP pose. As we consider chairs and tables standing or moving on the ground plane, the PnP pose estimate is further projected to the ground plane (xy-plane in allocentric coordinates), to obtain a stable and plausible object pose, compensating for noise or outliers in the keypoint detections.

To further improve the estimated object pose, the keypoint-based PnP pose estimate is refined via ICP alignment with the point cloud segments of the observed objects obtained from the semantic point cloud (cf. Fig. 3.9). For this, 3D object skeletons are assigned to point cloud segments in a data association step. For each frame, we obtain a set of 3D keypoint skeletons $\mathcal{K}$ and a set of point cloud segments $\mathcal{S}$ of the corresponding semantic class. For each segment $s_j \in \mathcal{S}$, we find the corresponding object skeleton $k_i \in \mathcal{K}$ with the minimum average distance between 3D object keypoints x_{l,k_i} and their nearest neighbor in the respective point cloud segment y_{l,s_j}:

$$d_{\text{kps-segm}}\left(k_i, s_j\right) = \frac{1}{L}\sum_{l=1}^{L}\left\|x_{l,k_i} - y_{l,s_j}\right\| , \tag{3.3}$$

$$k_{\min} = \underset{\forall k_i \in \mathcal{K}}{\arg\min}\left(d_{\text{kps-segm}}\left(k_i, s_j\right)\right) . \tag{3.4}$$

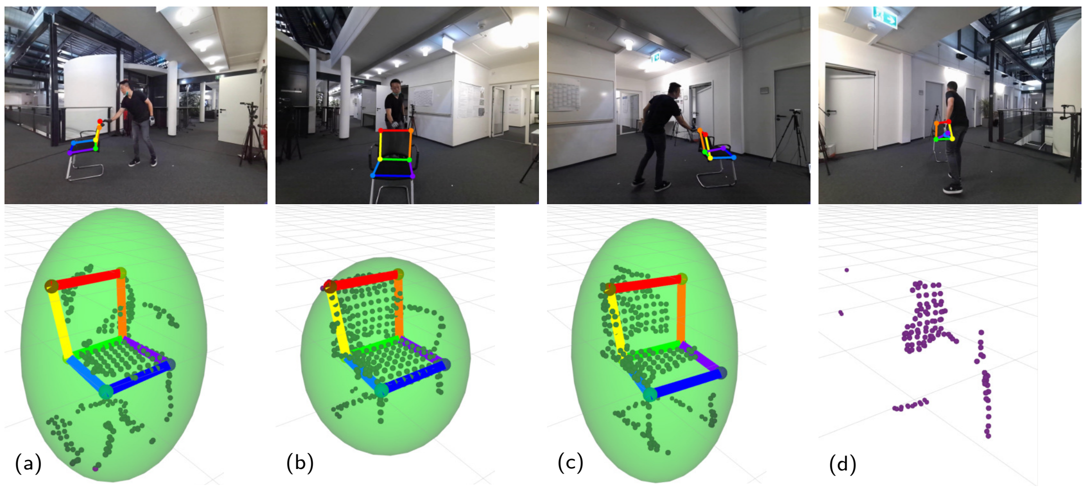

Figure 3.6: 2D keypoint detections and corresponding 3D poses calculated with PnP-RANSAC and ICP refinement. Top-row: keypoint detections in four different perspectives of a scene from the Behave dataset (Bhatnagar et al., 2022). Bottom-row: object pose estimation and associated point cloud segments. For (a)–(c), the method was able to estimate a pose for the object, while no valid pose estimate and data association could be obtained for perspective (d) due to the high occlusion.

Data association is performed in a greedy manner, starting with the largest point cloud segment, and associations are valid only when the obtained average keypoint to nearest point cloud neighbor distance $d_{\text{kps-segm}}(k_{\text{min}}, s_j)$ is below a threshold τ_{dist}. Segments and keypoint detections without a valid data association are discarded.

The object model point cloud, sampled from the 3D object mesh model and initialized with the PnP pose, is then aligned with the associated observed point cloud segment via ICP, resulting in the final object pose estimate.

The semantic object information, comprising the refined pose estimate, the data association distance $d_{\text{kps-segm}}(k_{\text{min}}, s_j)$, and the statistical distribution of the points in the cluster, is then streamed to a central backend, where object observations from multiple sensor perspectives are synchronized and fused. If the sub-map representation is chosen on the backend (cf. Sec. 3.3.6), the associated point cloud segments are additionally transmitted to the backend, significantly increasing the used network bandwidth (cf. Sec. 3.4.3.4). Fig. 3.6 shows object keypoint detections and resulting pose estimates with associated point cloud segments for an exemplary scene of the Behave dataset (Bhatnagar et al., 2022) with the *chair* object class.

3.3.5 *Allocentric Semantic Mapping*

Semantic point clouds from multiple calibrated camera perspectives are fused into an allocentric semantic map on the central backend, as illustrated in Fig. 3.7. For this, the 3D space is uniformly subdivided into cubic volume elements (voxels). We employ sparse voxel hashing (Quenzel and Behnke, 2021) as a memory-efficient data structure.

For indoor environments, prior information on building structure is often easily available, e.g., via floor plans or 3D models. To incorporate this prior information, we initialize the scene model with a prior map of the empty building (Fig. 3.7 (b)). Here, the prior map was obtained from aggregated laser scans of the empty rooms, but it

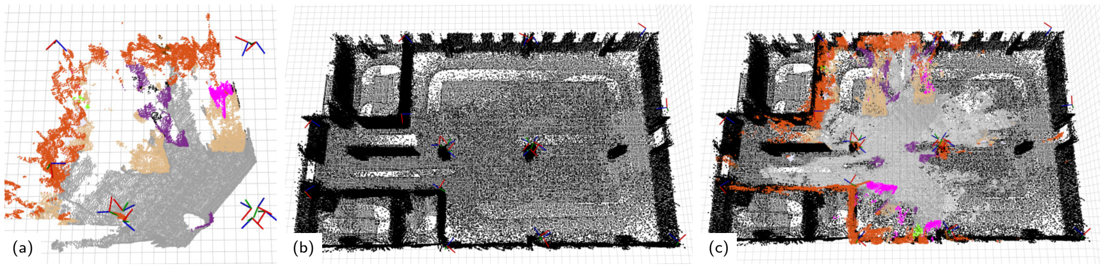

Figure 3.7: 3D semantic mapping: (a) semantic point cloud of a single sensor, (b) prior map, (c) fused semantic map. The smart edge sensors send semantic point clouds of their resp. perspectives to the backend. Here, the fused map is initialized with a prior map and updated with the observations including semantic classes.

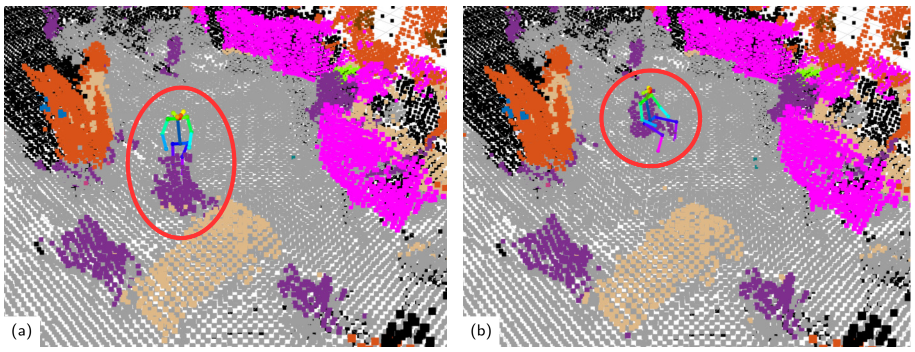

Figure 3.8: Map update: 3D semantic map and 3D person skeleton (a) before and (b) after moving a chair (highlighted with red circle). The semantic map is updated via ray-tracing to account for moving objects.

could also be replaced, e.g., by a floor plan with a fixed wall height or an architectural computer aided design (CAD) model of the building.

To include semantic information and current observations into the map, we transform the semantic point clouds from individual sensors (Fig. 3.7 (a)) into allocentric coordinates using the known camera calibration and bin the points into voxels of 10 cm side length. The semantic probabilities of all points falling into a voxel are fused probabilistically, using Bayes' rule (McCormac et al., 2017), assuming independence between observations $P(l_i|X_k)$ for the semantic point cloud X_k with label l_i for class i:

$$P(l_i|X_{1:k}) = \frac{P(l_i|X_{1:k-1})\,P(l_i|X_k)}{\sum_i P(l_i|X_{1:k-1})\,P(l_i|X_k)}\,. \tag{3.5}$$

We again use the implementation of Bayesian fusion in logarithmic form proposed by Bultmann, Quenzel, and Behnke (2023) for better numerical stability. Points labeled as *person* are not included in the semantic map, as dynamic human segments are tracked at a higher update rate via the 3D skeleton representation (cf. Sec. 3.3.7).

The fused semantic map (Fig. 3.7 (c)) contains 3D geometry and semantic classes of the areas observed by the smart edge sensors and is completed by the prior information for currently unobserved areas.

To account for moving objects, we adopt a simple ray-tracing approach to update occupancy information of the voxels (Schleich et al., 2021), as illustrated in Fig. 3.8. Starting from the sensor pose towards the measured voxels, we ray-trace using a 3D implementation of Bresenham's algorithm (Amanatides and Woo, 1987). All voxels

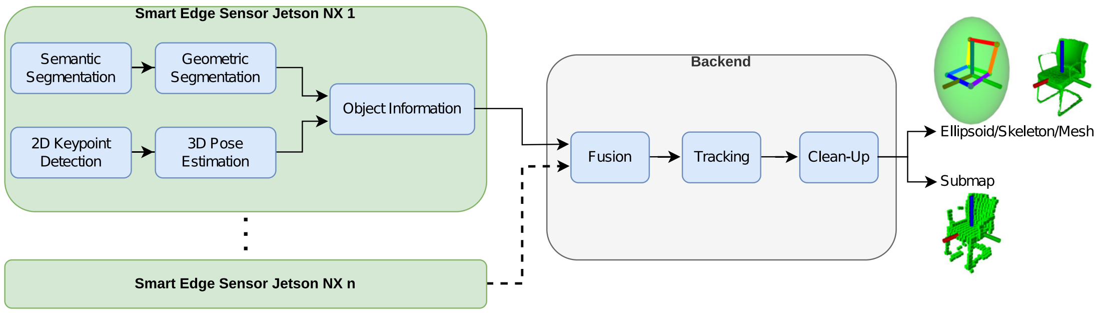

Figure 3.9: Overview of the object-level mapping pipeline: Smart edge sensors generate semantically and geometrically segmented point clouds. Simultaneously, 3D object poses are estimated via PnP using 2D keypoint detections. With the results, we calculate information about each observed instance. On the backend, observations from multiple views are fused and objects are tracked over time.

between the start and endpoint of the ray are updated as being free space, while the measured voxels are updated as being occupied. The semantic class probability is reset when a voxel state transitions from occupied to free.

The ray-tracing update, however, reacts gradually to object movement and does not implement any instance-level object representation. Hence, object trajectories cannot be reconstructed. To improve the scene understanding, instances of different furniture object classes are explicitly represented in the map through their 3D model or an object-centric sub-map and tracked through the static scene, as detailed in the following.

3.3.6 *Object-level Semantic Mapping*

Fig. 3.9 gives an overview of our proposed pipeline for multi-view object-level semantic mapping. In each sensor view, point clouds for the considered object classes are extracted via semantic segmentation of the RGB image and projection of the depth data (Sec. 3.3.3). The object point clouds are then geometrically segmented via the Euclidean cluster algorithm and a statistical outlier filter (Rusu and Cousins, 2011) to obtain a point cloud segment per detected object instance.

Simultaneously, object keypoint detection is performed on the RGB images and 6 DoF object poses are recovered via the PnP-RANSAC Algorithm using 2D–3D correspondences between detected image keypoints and 3D object model keypoints (Sec. 3.3.4). We assume 3D models of the considered object classes to be available as prior information on the sensor boards and the backend. The PnP pose estimate is projected to the ground plane, as the considered objects are standing or moving on the ground plane with only 3 degrees of freedom.

The keypoint-based pose estimates are then associated to point cloud segments via the closest distance between object keypoints and their nearest neighbor in the respective point cloud segment. The pose estimate is further refined via ICP alignment of the object model, initialized with the PnP pose estimate, with the corresponding point cloud segment (Sec. 3.3.4). Various semantic object properties are calculated from the associated point cloud segments and keypoint detections.

The semantic object information is streamed to a central backend, where the observations from multiple smart edge sensors are fused. We discern two different cases,

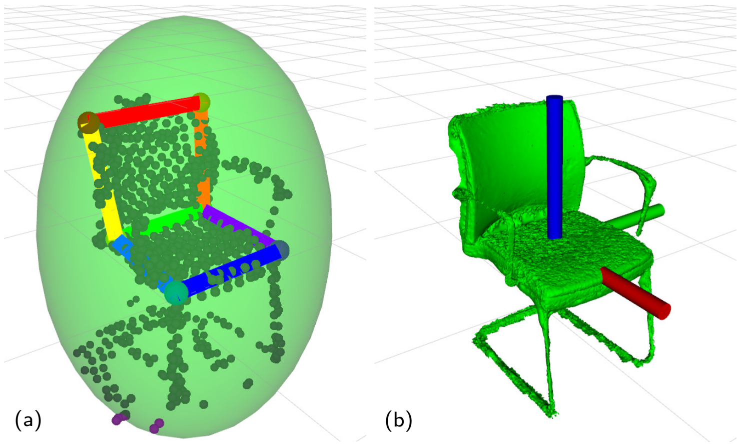

Figure 3.10: Fusion of keypoint poses and point cloud variance: (a) fused keypoint skeleton and the point cloud variance from smart edge sensor observations (cf. Fig. 3.6); (b) object mesh transformed with the fused pose estimate. Merged point cloud segments are shown as a reference in (a).

depending on the available network bandwidth and the used object representation: The transmitted information always comprises the pose estimate, the mean distance between object keypoints and nearest neighbors in the associated point cloud segment, and the statistics of the point distribution, visualized as a covariance ellipsoid anchored at the object model origin. Objects are represented in this case by a 3D keypoint skeleton with an associated point distribution ellipsoid or by their 3D mesh model.

If a 3D mesh model is not available, objects are represented by an object-centric volumetric sub-map, which can model arbitrary geometric shapes. For this, the point cloud segments are additionally transmitted to the backend, requiring higher network bandwidth (cf. Sec. 3.4.3.4).

A tracking module enables to robustly follow object trajectories through the scene and a clean-up step removes object hypotheses that moved out of view or were falsely initialized from noisy measurements.

The central backend receives semantic object pose and shape information from multiple smart edge sensor views. The data streams are software-synchronized according to their timestamps. Fused object pose estimates are obtained by i) transforming the object pose estimates of individual cameras to allocentric coordinates, using the known extrinsic camera calibration, and ii) weighted interpolation between the sensor views. The interpolation weights are inversely proportional to the data association distance (3.3), giving the highest confidence to perspectives where the keypoint-based pose estimate is most consistent with the point cloud segments. Spherical linear interpolation of quaternions is used for the orientations. The point segment distribution variance parameters are averaged using the same interpolation weights. Observations from at least one sensor view are required for a valid object instance. The fusion of multiple perspectives increases the robustness and accuracy of the pose and shape estimation. Fig. 3.10 shows the fused pose estimate using the individual poses from the three valid perspectives of Fig. 3.6.

If the sub-map representation is used for objects in the allocentric map, an object-centric sub-volume is maintained for each object instance, using a sparse voxel grid data structure, similar to the allocentric semantic map (cf. Sec. 3.3.5). Its size and

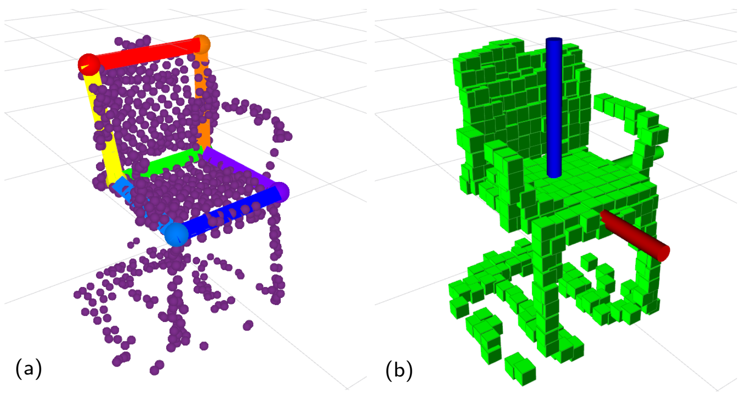

Figure 3.11: Sub-map update with fused point cloud data: The point measurements of the merged point cluster (a) are integrated into a volumetric sub-map (b). Simultaneously, the sub-map is updated with the estimated object pose.

resolution are chosen according to the represented object, independent of the resolution of the allocentric map. We use a voxel edge length of 5 cm in our experiments. To initialize and update the object sub-map, the point cloud segments associated to the detected object instances are additionally transmitted to the backend. In a first step, point cloud segments from individual views are transformed to allocentric coordinates, using the camera extrinsics, and concatenated to form a merged object point cluster. The merged object point cluster is then transformed into local object coordinates using the fused object pose estimate and integrated into the object-centric sub-map. Each point measurement increments the occupancy count of its corresponding voxel. Once the occupancy is above a threshold τ_{occ}, the voxel is considered occupied. The updated local sub-map is displayed at the estimated object pose in the allocentric scene model. Fig. 3.11 illustrates the sub-map update with the fused point cloud cluster from the three valid perspectives of Fig. 3.6. Fig. 3.12 shows 2D keypoint detections and fused 3D pose estimate for a sample scene of the Behave dataset with the *table* object class for both mesh and sub-map object representations.

Object instances are tracked over time on the backend via data association using a constant velocity model. The position of known objects is predicted using a moving average velocity, computed over a fixed time window of past positions, and the passed time Δt since the last synchronized frame-set was received from the sensors. For each observed object instance, the nearest neighbor from the tracked object hypotheses is used, using their predicted position. Data associations are valid only if the distance between observation and corresponding track is below a threshold τ_{track}. For observations with no valid association, new tracking hypotheses are initialized. In a clean-up step, object hypotheses that have not been observed for a longer time are removed.

3.3.7 *3D Human Pose Estimation with Occlusion Feedback*

The 3D joint positions of detected persons are recovered from a set of 2D keypoint detections from multiple viewpoints via triangulation, and the result is refined using a factor graph skeleton model, as introduced in Chapter 2. Furthermore, our framework implements a semantic feedback channel from backend to sensors that enables the local semantic models of each sensor to incorporate globally-fused 3D pose information.

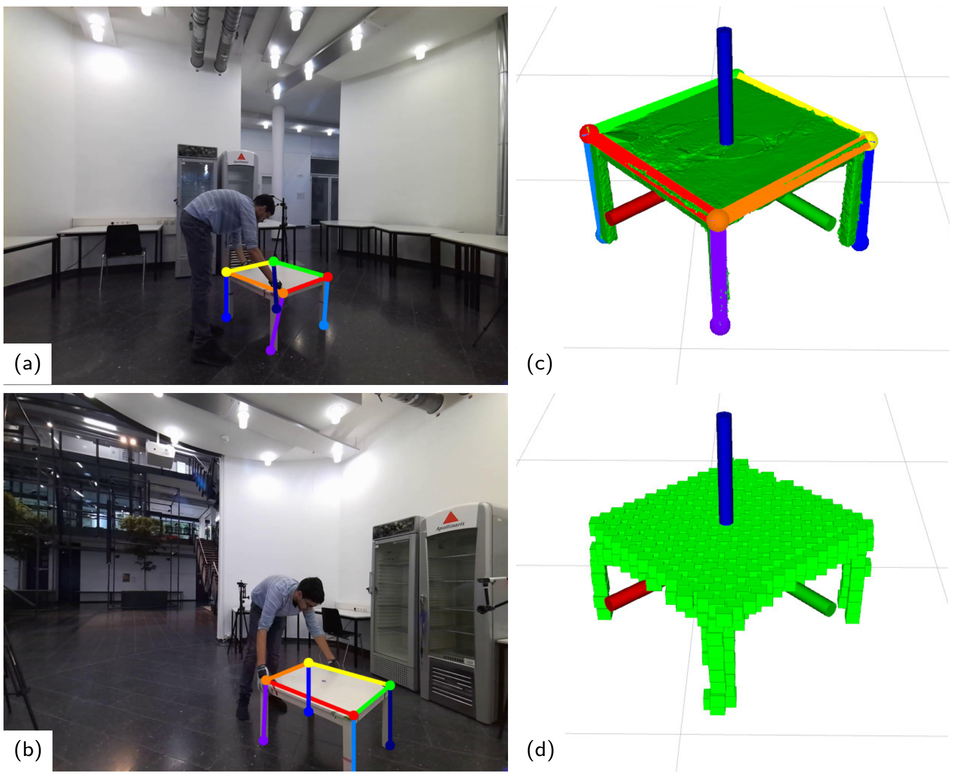

Figure 3.12: 2D keypoint detections for the *table* class from two perspectives of the Behave dataset (a, b); fused 3D pose represented through (c) mesh or (d) sub-map.

Using the allocentric 3D semantic map (cf. Fig. 3.3), we add occlusion information for each human joint to the semantic feedback. We employ ray-tracing to check each joint for occlusion in the respective local sensor view. For this, we traverse the ray from the respective camera pose to the 3D joint through the 3D map using Bresenham's 3D line-search (Amanatides and Woo, 1987). When the ray hits a minimum number of $k = 2$ occupied voxels, the joint is marked as occluded in the respective local view.

The benefits of the occlusion information for the local sensor model are illustrated in Fig. 3.13. Without occlusion feedback, heavy occlusion causes the pose estimation to collapse to the visible side only (Fig. 3.13 (b)), which cannot be recovered by the feedback on the heatmap level (Sec. 2.4.4). With occlusion information (Fig. 3.13 (a)), unreliable, occluded joint detections can be discarded, and the local model is completed by the more reliable semantic feedback. Completely occluded persons can also be added back into the local model (Fig. 3.13 (d)), making the sensor aware of persons that are going to re-appear in the future. Furthermore, the known occluded joints are excluded from multi-view triangulation in the next forward pass, as no new information can be gained from the respective sensor view. In Sec. 3.4.3.2, we show that the added occlusion information improves the overall consistency in terms of reprojection error.

We further investigate the reliability of the local depth estimate of skeleton joints from the Jetson NX smart edge sensors. The local depth enables estimating 3D joint positions from a single camera only, without dependence on other sensors. However, the RGB-D depth suffers from significant noise at larger distances and the depth measurement for a joint often is obstructed by occlusions or self-occlusions, as illustrated in Fig. 3.14. The local depth estimate results in a good approximation of the 3D skeleton for a front view,

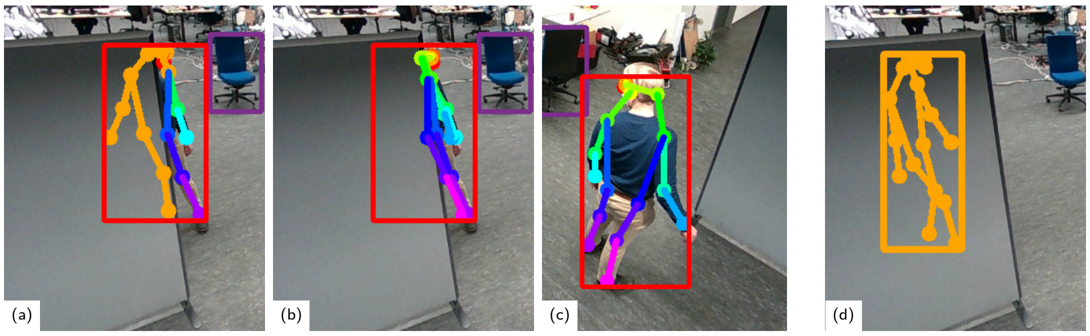

Figure 3.13: Occlusion information in semantic feedback: Local 2D pose estimation (a) with and (b) without occlusion information via semantic feedback, (c) reference view without occlusion, (d) fully occluded person. Person detections in red and skeleton keypoints colored by joint index. Occluded joints are marked in orange. Heavy occlusion causes the pose estimation to collapse to the visible side only. With occlusion information, unreliable, occluded joint detections can be discarded, and the local model is completed by the more reliable semantic feedback.

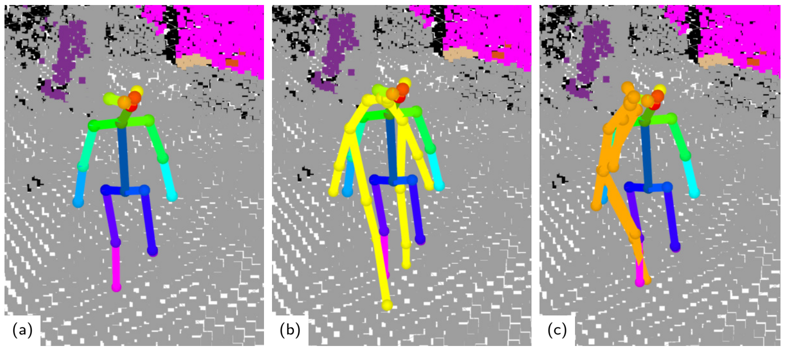

Figure 3.14: Comparison of multi-view triangulation and local depth for estimating person keypoints in 3D: (a) multi-view triangulated 3D skeleton, (b) local depth from front view, and (c) local depth from side view. The local depth estimate results in a good approximation of the 3D skeleton for a front view, but is inaccurate in case of self-occlusion, e.g. from a side view. Multi-view triangulation is more robust.

but is inaccurate in case of self-occlusion, e.g. from a side view. Multi-view triangulation is more robust to these issues but requires synchronization with other sensors.

The local depth estimate, however, can still be used as an indication to constrain the data association between cameras for multi-view triangulation. Person detections from different camera views are associated based on the epipolar distance of their joints using the efficient iterative greedy matching proposed by Tanke and Gall (2019). Keypoint detections from one image are projected as epipolar lines into the other cameras, where the distance from corresponding joint detections to the epipolar line is used as data-association cost. When a depth estimate is available, including an uncertainty interval computed from the keypoint confidence and the distribution of local depth readings, the matching can be restricted to a line segment. This helps to resolve ambiguous situations, where keypoints from multiple persons have a low distance to the epipolar line but are located at different positions along the line. Keypoints located on the line segment close to the projected depth estimate will receive lower data association cost while correspondences outside the projected depth interval will be discarded.

3.4 Evaluation and Experiments

In a first set of experiments, we evaluate our keypoint-based approach for 6 DoF object pose estimation from a single camera view on the YCB-V dataset (Xiang et al., 2018) (Sec. 3.4.1). Second, our method for multi-view object pose estimation and object-level mapping is evaluated on the public Behave dataset (Bhatnagar et al., 2022), using different scenes with human-object interactions (Sec. 3.4.2). Lastly, we evaluate the full system for 3D semantic scene perception in real-world experiments with the sensor network in a challenging, highly cluttered and dynamic lab environment with multi-person scenes (Sec. 3.4.3).

3.4.1 *Object Pose Estimation on YCB-Video Dataset*

In this study, some general design choices for the keypoint-based object pose estimation method are evaluated and justified. We compare different ways to define keypoints on object models and to scale keypoint estimation CNNs to handle multiple object classes.

The YCB-V dataset (Xiang et al., 2018) comprises over 130k images at VGA resolution of 21 different object classes and is widely used for robot manipulation and object pose estimation tasks. For each object, a textured mesh is included as 3D model with the origin of the object coordinate frame defined at its center. All objects are household objects relevant for real-world robot experiments in domestic service scenarios (cf. Fig. 3.15). The images contain multiple objects in realistic settings with changing lighting conditions, significant image noise and cluttered backgrounds.

3.4.1.1 *Implementation Details*

As we target the YCB-V dataset (Xiang et al., 2018) for generic object pose estimation in this set of experiments, without focus on real-time inference on the embedded hardware, we employ a more generic and powerful CNN architecture compared to the network used on the sensor boards (cf. Sec. 3.3.2), less optimized for computational efficiency, to achieve results comparable to the literature.

We base the following evaluations on the OpenPose framework (Cao et al., 2021). OpenPose is a keypoint-based bottom-up approach for human pose estimation in images. Together with heatmaps of keypoints, the CNN computes vector fields, called part affinity fields (PAFs), connecting the keypoints of an object instance. This allows to directly predict keypoints on the input image without prior segmentation or detection required. Local maxima in the heatmaps are assembled into instances via the PAFs.

The network architecture is adjusted to the YCB-V dataset as well as the used keypoints and PAFs. For an input image of size $H \times W$ and C object classes, the output shape is $\frac{H}{8} \times \frac{W}{8} \times (C \cdot 8 + 1)$ for the heatmaps and $\frac{H}{8} \times \frac{W}{8} \times (C \cdot 12 \cdot 2)$ for the PAFs. Heatmaps consist of eight keypoint channels per object class and the background, while PAFs consists of x and y channels for the 12 keypoint connections per class. Training labels are generated from the object model keypoints projected into the images of the video sequences from the dataset, using the annotated poses. A Gaussian blob is rendered at the keypoint position in the respective heatmap channel and the respective PAF channels represent the unit vector in the direction of the connection, within a fixed width along the connecting line, as proposed by Cao et al. (2021).

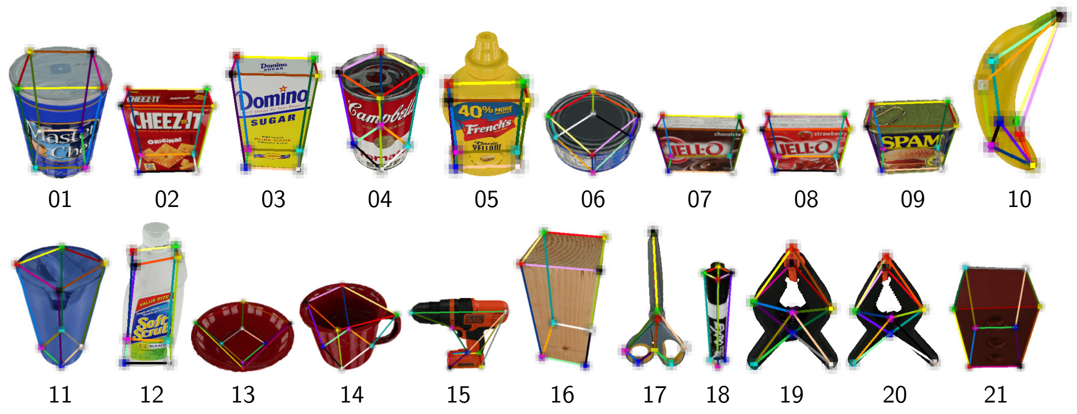

Figure 3.15: Objects of the YCB-V dataset with heatmaps of the manually defined keypoints and their interconnections.

To enable the network to estimate the poses for multiple object classes, the layer width of the intermediate stages needs to be scaled accordingly to the output layers. A network for all 21 object classes would require huge amounts of GPU memory. Because of this, we train a separate model for each object class. We verify in the ablation studies that the performance of these 1-object models is superior to models trained for multiple object classes.

The input image is processed by the network and heatmaps and PAFs are estimated. The local maxima of each heatmap are candidates for the respective keypoint. These keypoint candidates are grouped into object instances using the PAFs, as in the OpenPose framework (Cao et al., 2021). This step is repeated for every object class. The PnP and RANSAC algorithms are used to calculate the 6 DoF poses of the found object instances with four or more valid keypoints[2] using correspondences between detected 2D image keypoints and 3D model keypoints, as detailed in Sec. 3.3.4. In this study, no ICP refinement or multi-view fusion is employed.

All experiments run on a workstation PC with an RTX 2080 GPU, i7-8700K CPU, and 32 GB of RAM. Pose estimation takes 50 ms in average per object and image, thereof 8 ms for pre-processing, 8 ms for inference, and 34 ms for post-processing and PnP. The network requires 1.24 GB of GPU memory during inference.

3.4.1.2 *Selection of Object Model Keypoints*

The choice of the object model keypoints is an important design parameter of any keypoint-based method for object pose estimation. They need to be well localized on the object geometry and texture, to facilitate their CNN-based detection, and should be spread out on the object surface such that a stable and well-defined solution of the PnP-problem can be found. We compare two different ways to define keypoints and PAFs on the 3D object models in this study: They are chosen manually or automatically. Eight keypoints and twelve PAFs are defined per object class.

The manually defined keypoints are located on easy-to-find spots of the object geometry and texture and represent the object contour. If applicable, the keypoints are placed to form a cuboid. This set of keypoints is shown in Fig. 3.15. The automatically defined set of keypoints is chosen with the farthest-point-algorithm, inspired by PVNet (Peng

2 The PnP algorithm requires at least four correspondences for a unique solution.

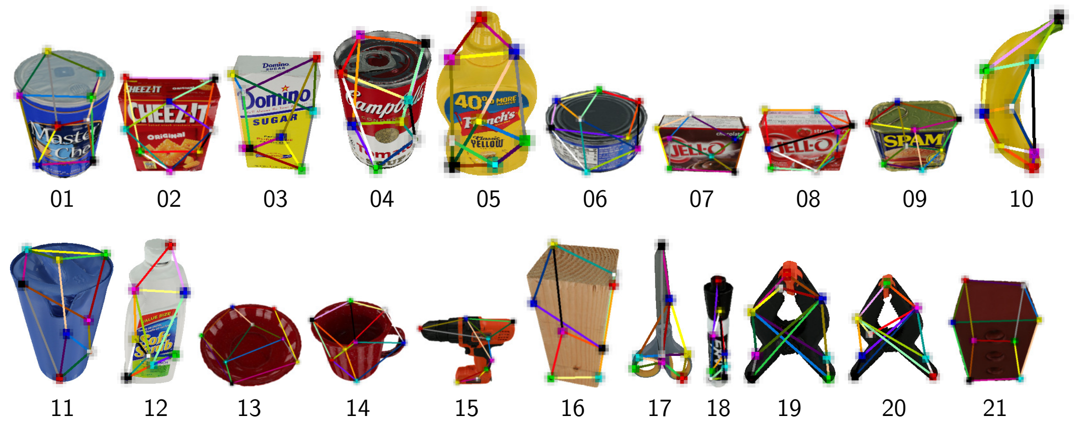

Figure 3.16: Objects of the YCB-V dataset with heatmaps of the automatically defined keypoints and their interconnections.

et al., 2019): Starting with the object center, points on the object surface which are farthest from the already chosen points are added to the keypoint set. Eight points on the object surface are retained—the center point is not part of the final keypoint set. The set of automatically chosen keypoints is shown in Fig. 3.16. The PAFs for both sets of keypoints are defined by hand. The objective is to choose connections that run along distinctive features and to form one upper and one lower polygon that are connected with vertical PAFs. The PAFs run along the keypoint connections displayed in Figs. 3.15 and 3.16 and have a fixed width. The automatically picked keypoints are less intuitively placed and harder to find than the manually picked ones. The reason for this is, that the automatically picked keypoints are often located on edges instead of corners and on surfaces instead of edges. Furthermore, the texture is ignored in the automatic selection although it is important to localize keypoints on images. The inferior performance of the automatically chosen keypoints is confirmed by the evaluation results in the ablation studies. Therefore, the manually chosen keypoints are used for the comparison of our results with other methods from the literature. Consequently, keypoints on the objects tracked in the real-world object-level mapping with the sensor network are also manually defined (cf. Sec. 3.3.4).

The YCB-V dataset contains several symmetric objects: 13, 16, 19, 20, and 21. The bowl (Obj. 13) is rotationally symmetric while the other objects possess discrete symmetry transformations. For each symmetric object, some poses are not distinguishable from each other. Hence, keypoints of the symmetric objects cannot be learned by the network if the symmetries are ignored. A simple elimination of symmetric poses during training is implemented in this work. All symmetry-equivalent poses are mapped to the same pose which ensures same relative position of keypoints on the image plane.

3.4.1.3 *Metrics*

We employ two standard metrics for evaluation: average 3D distance of model points (ADD) (Hinterstoisser et al., 2012) and 2D projection error (Brachmann et al., 2016).

Both metrics employ the meshes of the object models to calculate the pose error. The ADD metric is defined as:

$$\epsilon_{\text{ADD}} = \frac{1}{|V|} \sum_{\boldsymbol{v} \in V} \| (\boldsymbol{R}\boldsymbol{v} + \boldsymbol{t}) - (\tilde{\boldsymbol{R}}\boldsymbol{v} + \tilde{\boldsymbol{t}}) \| , \tag{3.6}$$

with $\tilde{\boldsymbol{R}}$ and $\tilde{\boldsymbol{t}}$ being the estimated rotation and translation, $\boldsymbol{R}$ and $\boldsymbol{t}$ defining the ground-truth pose and V the set of vertices of the object model mesh. For symmetric objects, the point-to-point correspondences can be ambiguous and the metric is adapted to compute the average distance using the closest point from the mesh (Xiang et al., 2018):

$$\epsilon_{\text{ADD-S}} = \frac{1}{|V|} \sum_{\boldsymbol{v}_1 \in V} \min_{\boldsymbol{v}_2 \in V} \| (\boldsymbol{R}\boldsymbol{v}_1 + \boldsymbol{t}) - (\tilde{\boldsymbol{R}}\boldsymbol{v}_2 + \tilde{\boldsymbol{t}}) \| . \tag{3.7}$$

The 2D projection metric computes the average 2D pixel distances between corresponding points projected onto the image plane of the evaluated view:

$$\epsilon_{\text{2DProj}} = \frac{1}{|V|} \sum_{\boldsymbol{v} \in V} \| \text{proj}(\boldsymbol{R}\boldsymbol{v} + \boldsymbol{t}) - \text{proj}(\tilde{\boldsymbol{R}}\boldsymbol{v} + \tilde{\boldsymbol{t}}) \| . \tag{3.8}$$

The evaluation scores are given in terms of the area under the curve (AuC). For this, the threshold for the respective distance metric is varied and the pose accuracy is computed for each threshold value. The maximum thresholds are set to 10 cm for ADD(-S) and 40 px for the 2D projection metric.

3.4.1.4 *Quantitative Results*

In Tab. 3.1, we give detailed evaluation results of the AuC scores for all 21 objects of the YCB-V dataset and compare them to the results of PoseCNN (Xiang et al., 2018). The proposed approach outperforms PoseCNN in terms of ADD for most of the non-symmetric objects and on average over all objects. The improvement is most significant for the box-shaped Objects 2, 3, 7, and 8 as well as for Objects 11, 15, and 17 which have a more complex shape (cf. Fig. 3.15). PoseCNN, on the other hand, achieves better results for the symmetric objects in terms of ADD-S. The proposed approach provides less accurate results for these objects, where several keypoint configurations can result in visually equivalent poses. This makes the keypoint estimation harder to learn and cannot be fully compensated by the symmetry handling during training (cf. Sec. 3.4.1.2). Also, keypoints are difficult to infer for objects with little prominent geometric features (e.g., edges or corners) such as Objects 10 and 18. The 2D projection metric scores per object are not reported by the authors of PoseCNN.

In Tab. 3.2, we further compare the overall results of our method with the recent benchmark for 6D object pose estimation (BOP) challenge 2020 (Hodaň et al., 2020, 2018). The BOP challenge defines slightly different evaluation metrics: The maximum symmetry-aware surface distance (MSSD) and maximum symmetry-aware projection distance (MSPD) metrics are similar to the ADD and 2D projection metrics, but give the maximum value instead of the average error and deal with symmetries. The visible surface discrepancy (VSD) metric is the percentage of pixels that are visible in the estimated and ground truth pose and are close in the image space. The formal definitions are given by Hodaň et al. (2020). We achieve the third-best result. In comparison to

Table 3.1: Area under accuracy Curve (AuC) for pose estimation of YCB-V objects. * denotes symmetric objects. Best results are marked bold.

Object	Ours			PoseCNN	
	ADD	ADD-S	2D Proj.	ADD	ADD-S
1	49.9	80.7	54.8	**50.9**	**84.0**
2	**80.5**	**88.4**	84.3	51.7	76.9
3	**85.5**	**92.4**	88.8	68.6	84.3
4	**68.5**	**81.4**	84.8	66.0	80.9
5	**87.0**	**93.3**	89.8	79.9	90.2
6	**79.3**	**89.7**	81.7	70.4	87.9
7	**81.8**	**89.5**	88.7	62.9	79.0
8	**89.4**	**94.0**	92.9	75.2	87.1
9	**59.6**	70.0	69.0	**59.6**	**78.5**
10	36.5	58.3	55.0	**72.3**	**85.9**
11	**78.1**	**86.9**	78.0	52.5	76.8
12	**56.7**	67.1	66.2	50.5	**71.9**
*13	**12.2**	23.5	4.1	6.5	**69.7**
14	54.0	76.9	75.2	**57.7**	**78.0**
15	**82.8**	**91.0**	88.2	55.1	72.8
*16	16.7	29.6	29.5	**31.8**	**65.8**
17	**46.0**	**64.1**	76.7	35.8	56.2
18	9.8	11.9	20.8	**58.0**	**71.4**
*19	20.0	47.4	8.9	**25.0**	**49.9**
*20	14.1	45.5	3.5	**15.8**	**47.0**
*21	12.1	29.7	2.3	**40.4**	**87.8**
average	**59.0**	72.7	65.0	53.7	**75.9**

Table 3.2: Results on YCB-V of the five best methods using only RGB images for the BOP challenge 2020 (Hodaň et al., 2020) in comparison with our work.

Name	AR	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	training data
CosyPose (Labbé et al., 2020)	0.821	0.772	0.842	0.850	pbr+real
EPOS (Hodaň et al., 2020)	0.696	0.626	0.677	0.783	pbr
Ours	0.575	0.506	0.567	0.654	pbr+real
CosyPose (Labbé et al., 2020)	0.574	0.516	0.554	0.653	pbr
Leaping (Jinhui Liu et al., 2020)	0.543	0.443	0.499	0.687	pbr+real
CDPNv2 (Z. Li et al., 2019)	0.532	0.396	0.570	0.631	pbr+real

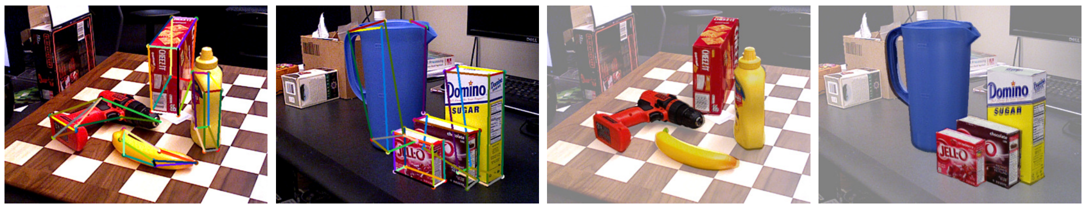

Figure 3.17: Qualitative results on the YCB-V dataset: (a) keypoints and connections, (b) transformed object models overlay on input image.

CosyPose (Labbé et al., 2020), which achieves the best result by a significant margin, we do not refine the initially estimated pose, which could further improve our result.

In Fig. 3.17, qualitative results on the YCB-V dataset are shown. 6 DoF object poses are estimated accurately despite occlusions and outlier keypoint detections.

3.4.1.5 *Ablation Studies*

Several systematic ablation studies are conducted in this work to evaluate the influences of different components and parameters of the proposed method.

PAFs. We investigate the benefit of PAFs for the YCB-V dataset, where a maximum of one instance per object class is present in an image. For this, we infer the keypoints of an object instance as the global maximum of the respective heatmaps and do not use the PAFs computed by the CNN to assemble keypoints into object instances. This heatmaps-only approach is compared to the full approach using the PAF output. The results are presented in Tab. 3.3. Using the PAFs improves the pose estimation result for almost all objects as well as the average accuracy. Without PAFs, recovering from wrong or ambiguous heatmap maxima is not possible, leading to inaccurate results especially in the case of occlusions and truncation. An exception is Object 18, where the pose estimation without PAFs is more accurate. This is due to the small object size leading to very short PAF vectors at the faces of the marker (cf. Fig. 3.15). These cannot be well detected by the model, leading to problems parsing the object skeleton.

Fig. 3.18 shows the accuracy-threshold curves for each metric with and without using PAFs. The improvement using PAF is most significant for small accuracy thresholds, demanding a precise estimation of the object pose. The two curves approach each other for higher thresholds.

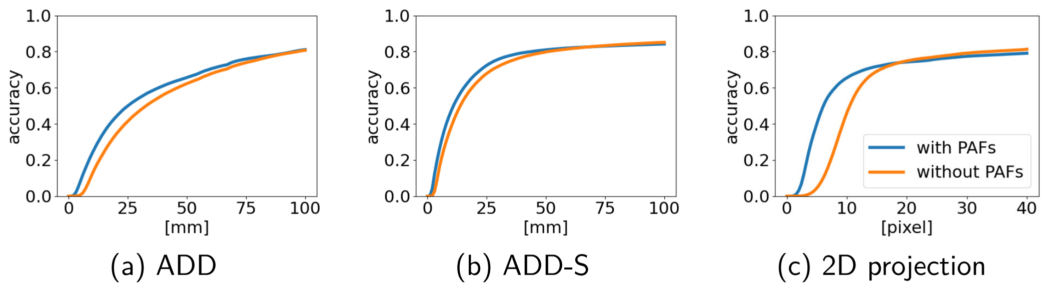

Figure 3.18: Accuracy curves for all YCB-V objects using PAFs vs. heatmaps only.

Table 3.3: AuC when using PAFs to connect keypoints into instances vs. only using global heatmap maxima. * denotes symmetric objects. Best results marked bold.

Object	using PAFs			heatmaps only		
	ADD	ADD-S	2D Proj.	ADD	ADD-S	2D Proj.
1	**49.9**	**80.7**	**54.8**	48.0	80.1	50.2
2	**80.5**	**88.4**	**84.3**	73.1	83.3	72.7
3	**85.5**	**92.4**	**88.8**	79.2	89.4	76.6
4	**68.5**	**81.4**	**84.8**	59.5	75.6	75.0
5	**87.0**	**93.3**	**89.8**	80.4	90.8	79.3
6	**79.3**	**89.7**	**81.7**	68.3	84.5	72.4
7	**81.8**	**89.5**	**88.7**	74.0	84.3	81.2
8	**89.4**	**94.0**	**92.9**	81.8	90.3	81.0
9	**59.6**	**70.0**	**69.0**	54.3	66.3	60.4
10	**36.5**	**58.3**	**55.0**	35.2	55.3	53.0
11	**78.1**	**86.9**	**78.0**	72.9	84.3	67.4
12	**56.7**	**67.1**	**66.2**	51.7	64.3	57.8
*13	**12.2**	**23.5**	**4.1**	11.5	23.2	**4.1**
14	**54.0**	**76.9**	**75.2**	49.1	72.0	63.3
15	**82.8**	**91.0**	**88.2**	76.8	88.3	77.1
*16	**16.7**	**29.6**	**29.5**	15.7	27.4	23.2
17	**46.0**	**64.1**	**76.7**	37.3	56.2	65.1
18	9.8	11.9	20.8	**36.1**	**43.1**	**65.1**
*19	**20.0**	**47.4**	8.9	17.8	45.1	**10.1**
*20	**14.1**	**45.5**	3.5	11.8	43.0	**4.7**
*21	**12.1**	**29.7**	**2.3**	6.2	18.3	0.2
average	**59.0**	**72.7**	**65.0**	54.5	70.4	58.9

Selection of Keypoints on Object Models. In Tab. 3.4, we compare evaluation results using manually and automatically chosen object keypoints for an exemplary subset of the object classes. The estimated pose generally is more accurate using the manually defined keypoints. As discussed in Sec. 3.4.1.2, these keypoints are easier to find and can be more precisely localized by the CNN architecture, as they are placed on distinct spots of the object geometry and texture (cf. Figs. 3.15 and 3.16). The locations of the automatically chosen keypoints, on the other hand, are often weakly constrained along edges or on the object surface, thus being predicted less precisely.

Number of Object Classes per Model. We also investigate the influence of training the CNN model to detect keypoints and PAFs for objects of one or of several classes. The results of this comparison are shown in Tab. 3.5 exemplary for a subset of the object classes. The performance of the models trained for only one object class

Table 3.4: AuC for manually and automatically picked keypoints.

Object	manually picked keypoints			automatically picked keypoints		
	ADD	ADD-S	2D Proj.	ADD	ADD-S	2D Proj.
1	**49.9**	**80.7**	54.8	47.7	78.5	**56.7**
2	80.5	88.4	84.3	**81.0**	**89.8**	**87.1**
3	**85.5**	**92.4**	88.8	82.4	90.8	**89.6**
4	**68.5**	**81.4**	**84.8**	68.4	80.8	84.3
5	**87.0**	**93.3**	89.8	83.7	92.0	**92.0**
6	**79.3**	**89.7**	81.7	77.3	88.9	**85.2**
7	**81.8**	**89.5**	**88.7**	75.9	85.0	85.2
8	**89.4**	**94.0**	**92.9**	81.0	89.6	92.4

Table 3.5: AuC for ADD(-S) and 2D-projection metrics comparing models trained to detect one object class versus models detecting two object classes.

Object	1-Object models			2-Object models		
	ADD	ADD-S	2D Proj.	ADD	ADD-S	2D Proj.
1	**49.9**	**80.7**	**54.8**	18.0	31.5	23.0
4	**68.5**	**81.4**	**84.8**	38.5	43.6	46.8
5	**87.0**	**93.3**	**89.8**	20.3	23.8	23.9
6	**79.3**	**89.7**	**81.7**	53.6	63.1	60.3
7	**81.8**	**89.5**	**88.7**	38.5	42.3	42.5
8	**89.4**	**94.0**	**92.9**	39.8	44.4	45.3
9	**59.6**	**70.0**	**69.0**	25.2	33.3	35.5
10	**36.5**	**58.3**	**55.0**	8.3	12.4	16.9
average	**66.2**	**81.3**	**75.2**	32.2	39.8	39.2

is significantly better. The plot of the accuracy-threshold curves shown in Fig. 3.19 confirms these results. Also, rescaling the width of the CNN model, i.e., doubling the number of channels at each stage of the network, does not significantly improve the results of the 2-object model, as is shown in Tab. 3.6. Therefore, a separate model per object class is used in this work.

3.4.2 *Multi-View Object-Level Mapping on Behave Dataset*

In the following set of experiments, we evaluate our approach for multi-view object pose estimation and object-level mapping on parts of the public Behave dataset (Bhatnagar et al., 2022). Keypoint estimation CNN, ICP pose refinement and multi-view fusion are employed as introduced in Sec. 3.3 and deployed on the embedded sensor boards. As indicated by the results of the previous study, keypoints are manually defined on distinct points of the considered object models, and a separate keypoint CNN is trained

Table 3.6: AuC using a 2-object model with and without rescaling layer width.

Object	doubled layer width			normal layer width		
	ADD	ADD-S	2D Proj.	ADD	ADD-S	2D Proj.
1	**19.1**	**31.5**	**25.8**	18.0	**31.5**	23.0
4	**41.5**	**47.1**	**47.4**	38.5	43.6	46.8
average	**32.2**	**40.7**	38.5	**32.2**	39.8	**39.2**

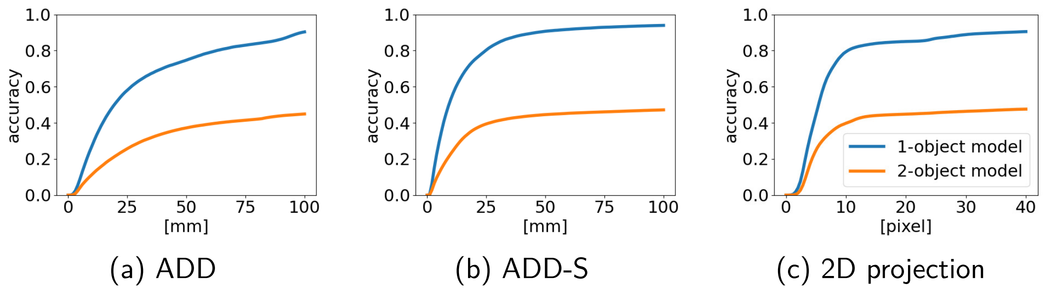

Figure 3.19: Accuracy curves for all YCB-V objects using models that detect one object class vs. models detecting two object classes.

per object class. PAFs are not part of the efficient model deployed on the smart edge sensors, where a top-down approach is employed for better scale invariance, as the sensor nodes are mounted significantly farther away from the observed objects than the camera in the YCB-V dataset. A detector provides bounding boxes of object instances and keypoints are estimated for each single-instance crop.

The Behave dataset comprises 321 video sequences totaling ∼15k frames, captured from four Kinect RGB-D cameras at a frame rate of 1 Hz. In the different scenarios, eight different persons interact with 20 different objects in five different environments. For each frame, annotations of the camera poses, 2D object and person segmentation masks, and pseudo-ground-truth object poses are available.

3.4.2.1 *Accuracy of Pose and Geometry Estimation*

We evaluate the accuracy of the proposed keypoint-based pose estimation with ICP refinement on five sequences of the Behave dataset, comprising ∼1k frames, where a person interacts with two different chairs and a table. We use the 3D mesh models of the chairs and table provided by the dataset as the basis for the PnP pose estimate.

We calculate translation and orientation error w.r.t. the pseudo ground-truth object poses from the dataset, using the quaternion geodesic distance for the orientation, and compare the PnP-only raw pose estimate with the proposed ICP refinement in Tab. 3.7 and Tab. 3.8. We discern two different options for the pose refinement via ICP alignment: refinement locally on the sensor boards, as described in Sec. 3.3.4, and refinement with the merged point cluster on the backend. The latter requires transmitting the point cloud segments from sensors to backend.

Tab. 3.7 reports the mean and standard deviation of the pose error when using the ground-truth point cloud segments and object boxes from the dataset as input to our keypoint estimation CNNs. The evaluation thus focuses on the keypoint-based pose

Table 3.7: Pose evaluation with ground-truth segmentation: transl. error (cm) and rot. error (°).

Scenario	Type	E_{trans}	σ_{trans}	E_{rot}	σ_{rot}
chairblack hand	PnP only	5.87	3.60	4.48	3.98
	PnP + ICP (local)	3.40	3.08	**3.33**	2.78
	PnP + ICP (backend)	**2.88**	3.11	3.52	2.92
chairblack sit	PnP only	5.23	2.46	**5.64**	6.97
	PnP + ICP (local)	**5.03**	2.70	5.74	6.95
	PnP + ICP (backend)	6.16	2.38	6.24	6.78
chairwood hand	PnP only	10.79	6.21	10.09	11.14
	PnP + ICP (local)	6.37	4.68	8.23	11.37
	PnP + ICP (backend)	**6.35**	4.73	**6.16**	8.73
chairwood sit	PnP only	13.17	5.03	7.14	6.23
	PnP + ICP (local)	5.50	1.31	5.44	5.98
	PnP + ICP (backend)	**5.30**	1.53	**5.38**	5.73
tablesquare move	PnP only	12.56	7.81	17.51	11.65
	PnP + ICP (local)	**7.82**	8.29	**3.37**	3.69
	PnP + ICP (backend)	8.01	8.19	3.93	6.08

Table 3.8: Pose evaluation with online segm. and det.: transl. error (cm) and rot. error (°).

Scenario	Type	E_{trans}	σ_{trans}	E_{rot}	σ_{rot}
chairblack hand	PnP only	7.48	9.22	7.27	8.49
	PnP + ICP (local)	**6.99**	8.23	**6.50**	10.05
	PnP + ICP (backend)	7.42	7.93	6.75	9.53
chairblack sit	PnP only	8.09	6.49	11.34	13.02
	PnP + ICP (local)	**7.54**	6.48	**8.71**	8.58
	PnP + ICP (backend)	8.18	6.53	10.79	11.34
chairwood hand	PnP only	9.05	6.05	10.52	15.37
	PnP + ICP (local)	**6.47**	4.42	10.51	15.16
	PnP + ICP (backend)	7.36	4.04	**8.32**	14.09
chairwood sit	PnP only	5.42	3.40	8.52	4.63
	PnP + ICP (local)	**5.38**	3.04	6.96	3.37
	PnP + ICP (backend)	5.50	3.17	**6.29**	2.56
tablesquare move	PnP only	15.76	11.12	18.34	11.39
	PnP + ICP (local)	**8.65**	6.85	**6.83**	8.95
	PnP + ICP (backend)	8.95	6.51	7.52	9.19

estimation part, excluding other error sources present in real-world input data. The ICP-based pose refinement significantly improves the translation error in all cases and the orientation error in all but one scenario. There are only little differences in accuracy between the refinement locally on the sensor board and on the backend.

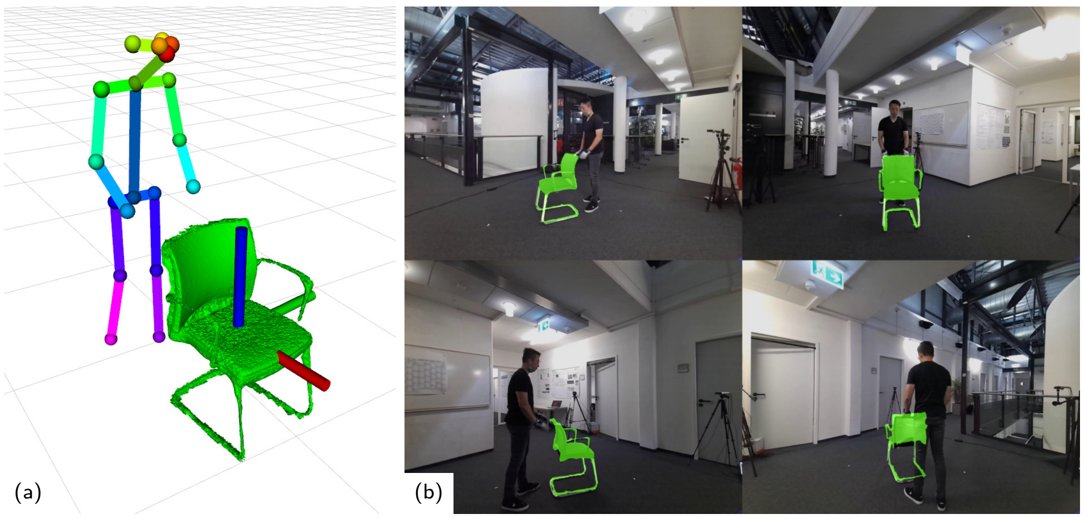

Figure 3.20: Mesh backprojected into the camera images: The object model mesh (a) transformed with the estimated object pose is rendered in each camera view (b). The mesh is a 3D scan of the used object. Therefore, even fine elements align well.

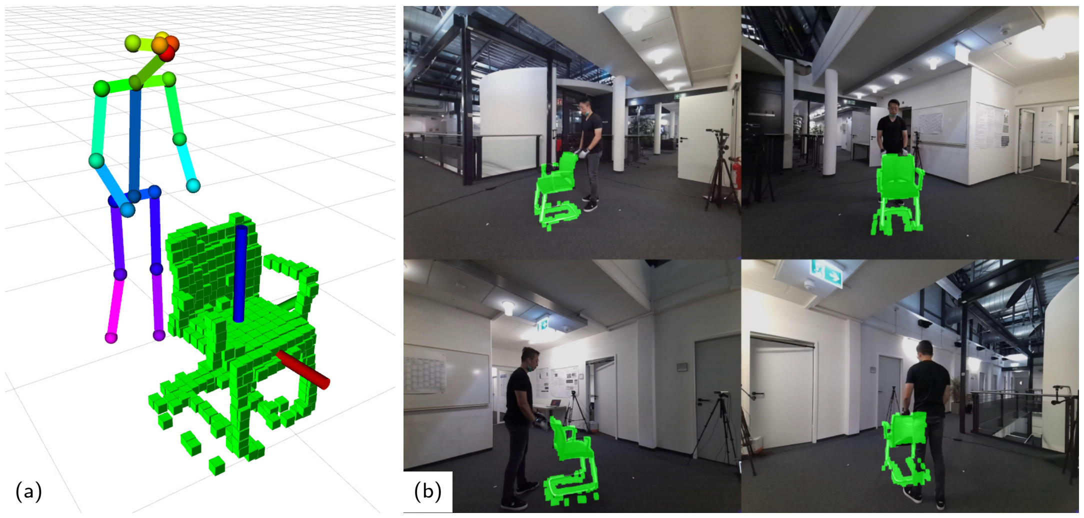

Figure 3.21: Sub-map backprojected into the camera images: Each voxel of the sub-map (a) is backprojected into the individual camera views (b). Because of the discrete resolution, fine elements like the legs are not accurately represented.

Tab. 3.8 reports the mean and standard deviation of the pose error when using online segmentation and object detection together with our keypoint estimation CNNs and thus evaluates the method's performance in real-world conditions. The ICP-based refinement again significantly decreases both translation and orientation errors in all scenarios. The ICP-refinement locally on the sensor boards consistently performs better w.r.t. the translation error than refinement on the backend. Therefore, we use the local ICP refinement for further evaluation and real-world experiments. In this case, the transmission of the point cloud segments is not necessary when using the mesh-based representation, significantly decreasing the required network bandwidth (cf. Sec. 3.4.3.4).

We further evaluate the pose and geometry estimation accuracy by calculating the intersection over union (IoU) between the estimated object models reprojected into the individual camera views and the respective ground-truth segmentation mask from the

Table 3.9: IoU scores for scenarios with ground truth segmentation and PnP + ICP refinement on sensor boards.

		Cam 1		Cam 2		Cam 3		Cam 4		Total
Scenario	Type	E_{IoU}	σ	E_{IoU}	σ	E_{IoU}	σ	E_{IoU}	σ	E_{IoU}
chairblack hand	Mesh	**0.77**	0.08	**0.81**	0.08	**0.78**	0.08	**0.57**	0.08	**0.73**
	Sub-map	0.61	0.07	0.68	0.09	0.57	0.09	0.54	0.08	0.60
chairblack sit	Mesh	0.39	0.16	**0.55**	0.19	**0.50**	0.13	0.57	0.13	**0.50**
	Sub-map	**0.42**	0.12	0.33	0.16	0.46	0.10	**0.65**	0.09	0.47
chairwood hand	Mesh	0.62	0.1	**0.65**	0.09	0.62	0.07	**0.69**	0.10	0.61
	Sub-map	**0.66**	0.11	0.63	0.14	**0.68**	0.12	0.67	0.12	**0.66**
chairwood sit	Mesh	0.57	0.13	**0.56**	0.07	0.57	0.09	**0.64**	0.16	**0.59**
	Sub-map	**0.61**	0.10	**0.56**	0.11	**0.61**	0.08	0.54	0.11	0.58
tablesquare move	Mesh	**0.75**	0.13	**0.75**	0.11	0.69	0.11	0.53	0.09	0.68
	Sub-map	0.72	0.14	0.74	0.15	**0.78**	0.15	**0.65**	0.15	**0.72**

dataset annotations. The reprojection of the 3D object model into the camera views is illustrated in Fig. 3.20 and Fig. 3.21 for the object model mesh and volumetric sub-map representations, respectively. As the 3D object mesh originates from an offline 3D scan of the object, it represents fine structures, such as armrests or legs in high detail. The fine structures of the chair align well with the images of all four camera perspectives in Fig. 3.20, showing high accuracy of the estimated object pose. The sub-map, on the other hand, has a discrete spatial resolution and cannot accurately represent the chair legs. Furthermore, it is affected by noisy point cloud measurements.

Tab. 3.9 and Tab. 3.10 report quantitative results of the IoU evaluation, using ground-truth and online point cloud segments and object detections as input, respectively, and compare the mesh- and sub-map-based object representations. For a fair comparison between mesh and sub-map, the annotated image segmentation masks are extended with a 10×10 dilation kernel to match the discrete 5 cm spatial resolution of the sub-maps. With ground-truth point cloud segment inputs, the mesh-based object representation performs better than the sub-maps in three of five scenarios, averaged over the four cameras. With online segmentation and detection inputs, the mesh-based representation performs better or equal in all five scenarios. The IoU is slightly lower in the real-world scenarios, accounting for the higher noise in the input data.

3.4.3 *Real-World Experiments*

We evaluate the proposed sensor network for 3D semantic scene perception system in challenging, cluttered real-world indoor scenes with multiple persons and different furniture objects.

Table 3.10: IoU scores for scenarios with online segmentation and detection and PnP + ICP refinement on sensor boards.

		Cam 1		Cam 2		Cam 3		Cam 4		Total
Scenario	Type	E_{IoU}	σ	E_{IoU}	σ	E_{IoU}	σ	E_{IoU}	σ	E_{IoU}
chairblack hand	Mesh	**0.70**	0.10	**0.75**	0.09	**0.68**	0.11	**0.56**	0.11	**0.67**
	Sub-map	0.55	0.10	0.64	0.08	0.54	0.10	**0.56**	0.09	0.57
chairblack sit	Mesh	**0.48**	0.26	**0.50**	0.26	**0.53**	0.27	0.52	0.16	**0.51**
	Sub-map	0.43	0.13	0.32	0.16	0.41	0.16	**0.61**	0.11	0.44
chairwood hand	Mesh	**0.56**	0.10	**0.57**	0.11	**0.61**	0.09	**0.65**	0.12	**0.59**
	Sub-map	0.51	0.09	0.49	0.10	0.55	0.08	0.58	0.09	0.53
chairwood sit	Mesh	**0.56**	0.14	**0.61**	0.08	**0.55**	0.10	**0.60**	0.12	**0.58**
	Sub-map	0.43	0.16	0.43	0.10	0.31	0.11	0.33	0.14	0.38
tablesquare move	Mesh	**0.68**	0.13	**0.68**	0.12	0.67	0.09	0.46	0.11	**0.62**
	Sub-map	0.60	0.10	0.61	0.11	**0.69**	0.10	**0.58**	0.14	**0.62**

3.4.3.1 *Implementation Details*

Our sensor network consists of 20 smart edge sensors, thereof 4 based on the Jetson NX board, as introduced in this chapter, and 16 based on the Google Edge TPU(cf. Chapter 2). The boards are connected to mains power supply and the power consumption of an NX board is 20-25 W during inference, thereof ∼5 W for powering the RGB-D camera, compared to 7 W for the Edge TPU board. The sensors cover an area of roughly 12×22 m. The cameras face downward towards the center and run at 30 Hz. We conduct experiments with the proposed, extended sensor network, with 8 persons moving in the covered area, which are evaluated in the following using a sequence of 106 s containing ∼3,000 frames per camera. Further evaluation is performed on three shorter sequences with a focus on object-level mapping and interactions between persons and objects

3.4.3.2 *Quantitative Results*

To analyze the consistency between local and globally-fused human pose estimation, we evaluate the error between 2D poses detected in the individual sensor views and reprojected fused 3D poses in Tab. 3.11. The reprojection error decreases for all joint classes when using the semantic feedback introduced in Sec. 2.4.4 over a purely feed-forward pipeline. Adding the occlusion information, as proposed in Sec. 3.3.7, further decreases the reprojection error, as unreliable occluded keypoint detections can be discarded and excluded from multi-view triangulation. Constraining the data association using the local depth estimates of the RGB-D cameras gives a further small improvement. The proposed pipeline leads to the lowest reprojection error for all joint classes, amounting to 4.51 px on average, indicating that the consistency between local and globally-fused pose estimation increases through the semantic feedback with occlusion information and by using local depth estimates in the data association step for multi-view triangulation.

Table 3.11: Evaluation in real-world multi-person scenes with 20 cameras and 8 persons: Reprojection error (px) per joint class between detected 2D poses and fused 3D poses.

Feedback	Cams	Pers	Head	Hips	Knees	Ankls	Shlds	Elbs	Wrists	**Avg**
w/o fb	20	8	5.09	5.98	5.75	6.87	4.67	5.53	6.95	5.69
fb	20	8	4.76	5.51	4.98	5.94	4.34	4.88	5.66	5.08
fb + occl.	20	8	4.36	4.68	4.37	5.44	3.97	4.38	**5.04**	4.56
fb + occl. + local depth	20	8	**4.30**	**4.63**	**4.32**	**5.42**	**3.91**	**4.33**	**5.04**	**4.51**

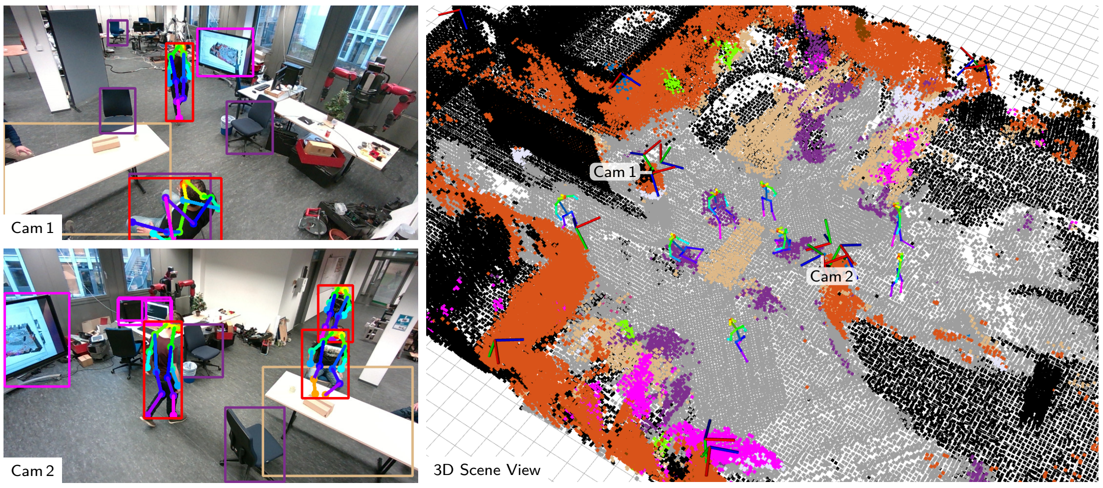

Figure 3.22: Experiments in real-world multi-person scenes: Local detections in two reference sensor views and 3D semantic scene model with 3D poses of eight persons estimated in real time. The room geometry is accurately represented, fusing prior map (black) and observations of smart edge sensors with semantic classes (e.g., tables, chairs, computers). Interactions between persons and the scene, e.g., persons sitting on chairs (violet), are explained in a physically plausible way.

3.4.3.3 *Qualitative Results*

Multi-Person Pose Estimation and 3D Scene Geometry. An exemplary scene of the real-world multi-person experiments is shown in Fig. 3.22. Local detections and pose estimation in two reference camera views are depicted together with the 3D semantic scene view. 3D poses of eight persons are estimated online, in real time during the experiment. The semantic map represents the 3D geometry of the scene, fusing prior map and current sensor observations, including semantic class probabilities. Interactions between persons and objects in the scene, e.g., persons sitting on chairs, are explained in a physically plausible manner by the scene model. Here, the scene is represented by a single, allocentric voxel-map, without object instances. Experiments with object-level mapping are shown in the following paragraph. A video of the experiments is available on our website[3].

3 https://www.ais.uni-bonn.de/videos/IAS_2022_Bultmann

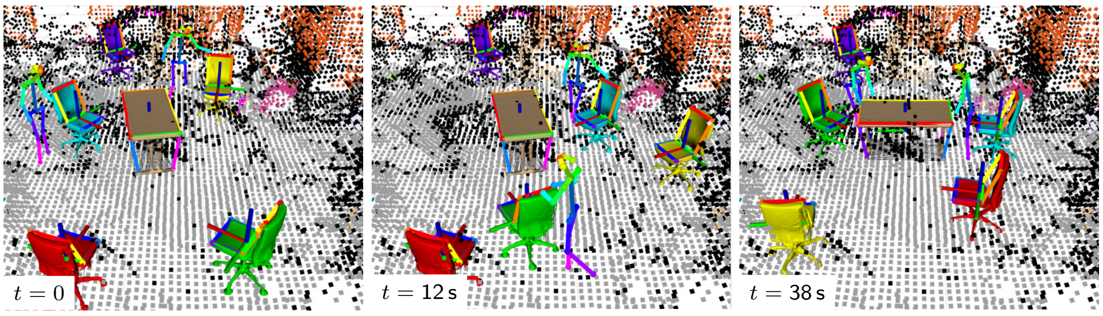

Figure 3.23: Scenario 1: Two persons interacting with five chairs and a table, colored by instance ID, in our cluttered lab environment. Four chairs and the table are being moved. The purple chair is standing still.

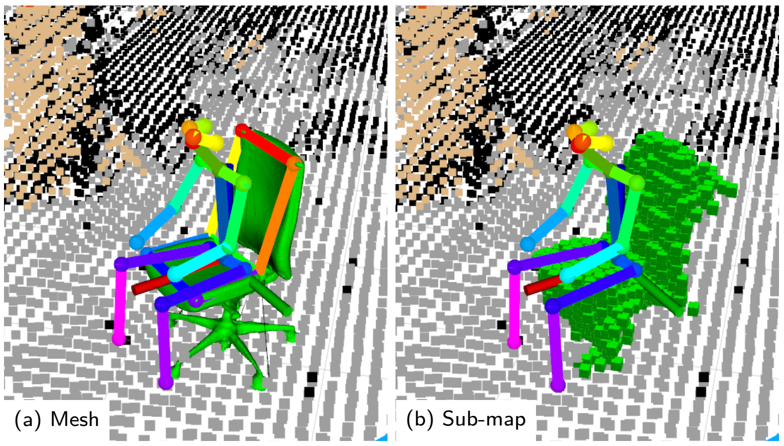

Figure 3.24: Scenario 2: A chair being occluded by a sitting person. The pose estimate and geometry representation remain stable under high occlusion and interaction between person and object is explained in a physically plausible manner by the scene model.

Object-Level Scene Perception in three Scenarios. We further demonstrate the performance of our object-level mapping in the highly-cluttered, dynamic environment in three different scenarios. Here, we employ offline 3D scans of the office chairs and table present in the environment as object models. Object pose estimation and geometry update are calculated online, in real time. A video of the experiments is available on our website[4].

In Scenario 1, two persons interact with five chairs and a table in our lab environment, as shown in Fig. 3.23. Point measurements of the tracked objects are not included in the allocentric map of the static geometry. The chairs and table are represented by their respective prior 3D mesh model transformed to the estimated pose. The movement of the objects through the scene is tracked online, in real time.

In Scenario 2 (Fig. 3.24), a person is sitting on a chair, occluding it partially. The estimated object models remain stable also under high occlusion and interactions between persons and objects are explained in a physically plausible manner.

In Scenario 3 (Fig. 3.25), a chair is being moved while being occluded by a sitting person. The movement of the object through the scene can be tracked by our proposed approach even under high occlusions.

4 https://www.ais.uni-bonn.de/videos/IRC_2022_Hau_Bultmann

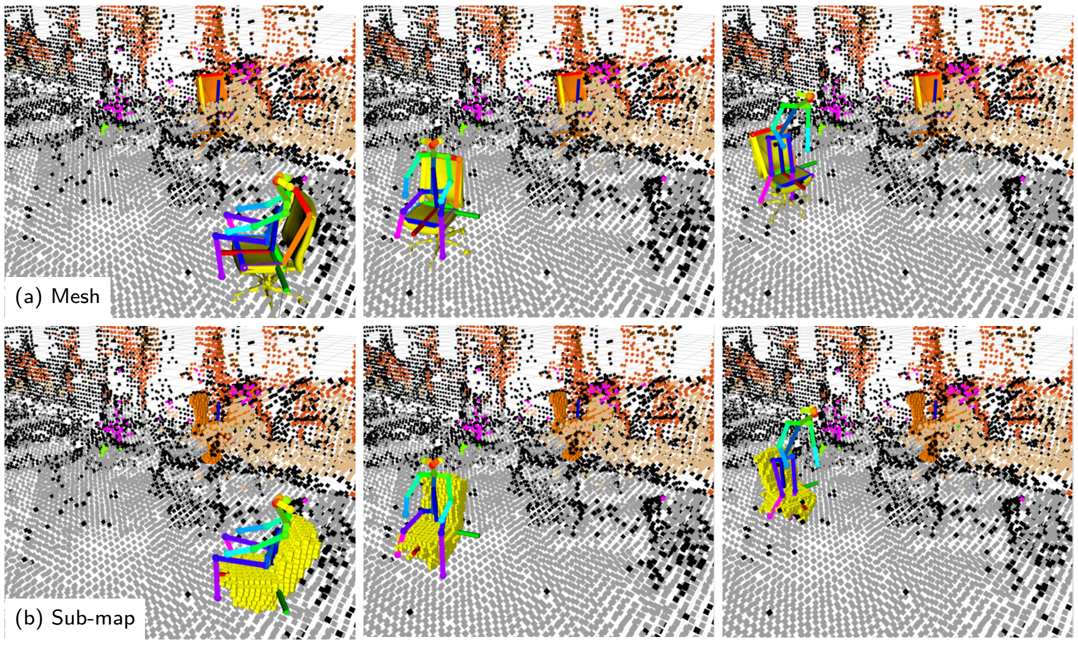

Figure 3.25: Scenario 3: The yellow chair is being moved (from left to right) while being occluded by a sitting person. The object movement can be tracked consistently despite the occlusion.

3.4.3.4 *Run Time and Network Bandwidth*

RUN TIME ANALYSIS. We analyze the run time of CNN inference and the validation score on the respective training dataset (cf. Sec. 3.3.2) on different embedded hardware accelerators and for different numerical precision for the employed models in Tab. 3.12. Thermal detector and RGB segmentation are only executed on the Jetson NX, as the Edge TPU does not have enough computational power to run all models in parallel. The run times on Jetson NX in 16-bit floating-point mode (fp16) are comparable to 8-bit quantized (int8) inference on the Edge TPU. The inference times roughly halve when using int8 precision on Jetson NX for the detectors and also decrease for the segmentation. For pose estimation, here stated for a single crop and batch size 1, the difference is less significant. The inference time is only about 4 ms, and the numerical precision is less relevant compared to other overhead from the inference framework.

The validation score is given as bounding-box or keypoint mean average precision (mAP) for the *person* class as defined for the COCO dataset (Lin et al., 2014) for the detectors or pose estimation, respectively, and as mean intersection over union (mIoU) for the semantic segmentation. It decreases between 0.2 and 1.2 % when using int8 precision instead of fp16. The slightly better performance of the RGB detector on the Edge TPU can be explained as it was trained for the *person* class only, while the detector used on Jetson NX was trained for *person* and 12 indoor object classes.

Table 3.13 shows the scaling of processing time for pose estimation, including CNN inference and post-processing, with an increasing number of person detections per image. During our experiments with 8 persons in the scene, a maximum of 6 persons were visible at a time in one camera. On the Edge TPU sensors, crops are processed one by one, as only a batch size of one is supported, and the run time scales linearly. Up to three persons can be tracked at the full camera frame rate of 30 Hz. On the Jetson NX

Table 3.12: Average inference time and validation score (given as mAP for detectors and pose estimation and mIoU for segmentation) of CNN models (batch size 1) on different embedded hardware and for different numerical precision.

Model	Input Res.	Edge TPU (int8)		Jetson NX (fp16)		Jetson NX (int8)	
		time	val. score	time	val. score	time	val. score
RGB det.	848 × 480	-	-	24.1 ms	36.2 %	**11.8 ms**	36.0 %
RGB det.	640 × 480	21.5 ms	**36.7 %**	-	-	-	-
Pose est.	192 × 256	4.5 ms	68.4 %	4.0 ms	**69.3 %**	**3.5 ms**	68.6 %
Thermal det.	160 × 120	-	-	13.9 ms	**25.4 %**	**6.0 ms**	24.9 %
RGB segm.	849 × 481	-	-	27.0 ms	**50.0 %**	**20.0 ms**	48.8 %

Table 3.13: Average processing time for pose estimation (inference + post-processing) for increasing number of person detections per image. Batch processing can be used on Jetson NX while crops are processed one by one on the Edge TPU.

Sensor Type	Precision	1	2	3	4	5	6
Edge TPU	int8	14.4 ms	23.6 ms	33.0 ms	43.9 ms	54.3 ms	65.9 ms
Jetson NX	int8	12.1 ms	**15.0 ms**	**18.0 ms**	27.4 ms	**31.0 ms**	**38.8 ms**
Jetson NX	fp16	**11.8 ms**	17.4 ms	21.9 ms	**26.4 ms**	31.7 ms	41.5 ms

platform, batch-processing is possible, and therefore the run times scale sub-linearly. Up to five persons can be tracked at the full camera frame rate. For pose inference, there is only a small difference in run time between fp16 and int8 mode on Jetson NX.

We run CNN inference in fp16 mode on the Jetson NX smart edge sensors during our online experiments to benefit from the higher numerical precision, as 8-bit quantization gives only little gains in run time for the pose estimation with strong real-time constraints, and the fp16 inference time is sufficient for the other models that run with lower priority at a 1 Hz update rate.

NETWORK BANDWIDTH. We further evaluate the required communication bandwidth for the different object representations in the semantic map. Object detection and tracking run at an update rate of 1 Hz. The semantic object properties, consisting of object pose, point segment variance, and data association score amount to only 84 Bytes per detected object instance, requiring very low network bandwidth. The object model and keypoint definitions are a-priori known on both backend and sensors and need not be transmitted during online operation. The required network bandwidth significantly rises when using the sub-map representation, as raw point cloud segments are transmitted from sensors to backend for each detected object. A point cloud segment of an average size of 250 points amounts to ∼9 kB of transmitted data.

3.5 Discussion

In this chapter, we presented a network of distributed smart edge sensors for multi-view 3D semantic scene perception, including static or slowly moving geometry, furniture objects, and dynamic human motions. RGB-D and thermal camera images are processed locally on the sensor boards with vision CNNs for person and object detection, semantic segmentation, and human and object pose estimation. 2D human keypoint detections, augmented with the RGB-D depth estimate, pose and shape information of detected *chair* and *table* objects, and semantically annotated point clouds are streamed from the sensors to a central backend, where multiple viewpoints are fused into an allocentric 3D semantic scene model.

The individual sensors incorporate global context information into their local models via a semantic feedback channel. For this, the globally fused 3D human poses are projected into the sensor views, where they are fused with the local detections. The estimated 3D geometry enables to add occlusion information for each joint to the semantic feedback, so that unreliable, occluded joint detections can be discarded and the local models complemented by the more reliable feedback joint positions.

Our method enables pose estimation and tracking of dynamic objects through the static scene geometry. Objects are represented via an a-priori known 3D mesh model or a volumetric sub-map that is learned online. Only a few semantic object properties, such as estimated pose and point distribution variance are transmitted from the sensors to the backend, requiring little network bandwidth. Only when volumetric sub-maps are required on the backend, the raw point cloud segments associated to object instances are additionally transmitted, significantly increasing the network traffic.

The object pose estimation follows a two-stage approach of keypoint detection and PnP pose estimation. Keypoints are defined on prominent geometric features of the object model (i.e., corners) to form a cuboid-like structure. Object poses are then calculated via the PnP-RANSAC algorithm using 2D-3D correspondences between detected and model keypoints. When depth data is available, poses are further refined using associated point cloud segments via ICP alignment. As the keypoint detection CNNs for deployment on the embedded sensor boards are trained only on synthetic data, the method can easily be extended to different object classes.

In a first study, our approach for object pose estimation from a single view is evaluated on the YCB-V dataset, containing 21 typical household objects important for domestic service robot applications. Our method achieves accuracy comparable to recent state-of-the-art methods using RGB images only. The usage of manually selected keypoints is advantageous over automatically defined keypoint locations. Models trained for a single object class perform significantly better than models trained for multiple object classes.

We then quantitatively evaluate the pose estimation accuracy of our approach for multi-view object pose estimation and object-level mapping on the public Behave dataset, showing pose errors below 9 cm and 9° with online input data processing using lightweight CNN architectures efficient on the embedded sensor hardware.

We built a sensor network of 20 smart edge sensors, thereof 4 based on the novel Jetson NX board, covering an area of about 12×22 m in a real-world lab environment. We demonstrate the application of the proposed system for multi-view 3D semantic scene perception in challenging, cluttered real-world scenes with up to 8 persons and different furniture objects and evaluate its performance. Dynamic human motions are estimated in real time and the semantically annotated 3D geometry provides a complete scene view that also explains interactions between persons and objects in the scene. Multiple chairs and a table are tracked through the scene online, in real time, even under high occlusions.

Future work includes using the 3D semantic scene model and human poses estimated by the smart edge sensors to enable anticipatory robot behavior and safe human-robot interaction in a shared workspace. Mobile sensor nodes could further be added to the sensor network for active exploration of areas not covered by the permanently installed sensors (cf. Chapter 5). With regards to object pose estimation, directions for future work include improving the handling of symmetric objects, which show sub-optimal performance in our evaluation. Furthermore, a method for automatic keypoint selection without sacrificing accuracy compared to manual selection should be investigated, enabling an efficient extension the method to novel object classes.

4 Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles with Label Propagation for Cross-Domain Adaptation

Preface

This chapter is adapted from Bultmann, Quenzel, and Behnke (2021), previously published by IEEE and presented at the 10th European Conference on Mobile Robots (ECMR 2021), and its extension Bultmann, Quenzel, and Behnke (2023), previously published by Elsevier in the Robotics and Autonomous Systems journal.

Statement of Personal Contribution

The author of this thesis substantially contributed to the following aspects of the publication (Bultmann, Quenzel, and Behnke, 2021): the literature survey, the conception, formalization, design, and implementation of the proposed methods for semantic perception and multi-modality fusion, the preparation of the manuscript, as well as the revision and final editing of the version to be published. He further contributed to the evaluation of the proposed approach, the preparation and conduct of outdoor UAV experiments and the analysis and interpretation of the experimental results.

The author of this thesis substantially contributed to all aspects of the publication (Bultmann, Quenzel, and Behnke, 2023), in particular to the aspects covering label propagation for cross-domain adaptation that extend the conference version, including the literature survey, the conception, formalization, design, and implementation of the proposed methods, the preparation and conduct of experiments for the evaluation of the proposed approach, the analysis and interpretation of the experimental results, the preparation of the manuscript, as well as the revision and final editing of the version to be published.

The content presented in this chapter, unless otherwise stated, is the contribution of the author of this thesis. The methods for volumetric semantic mapping were provided by the co-author Jan Quenzel. Hence, Sec. 4.3.4 was significantly adapted to match the thesis author's contributions and a reference to the original publication (Bultmann, Quenzel, and Behnke, 2023) is given for a detailed description of the implementation.

Abstract

Unmanned aerial vehicles (UAVs) equipped with multiple complementary sensors have tremendous potential for fast autonomous or remote-controlled semantic scene analysis, e. g. for disaster examination.

Here, we propose a UAV system for real-time semantic inference and fusion of multiple sensor modalities. Semantic segmentation of LiDAR scans and RGB images, as well as object detection on RGB and thermal images, run online onboard the UAV computer

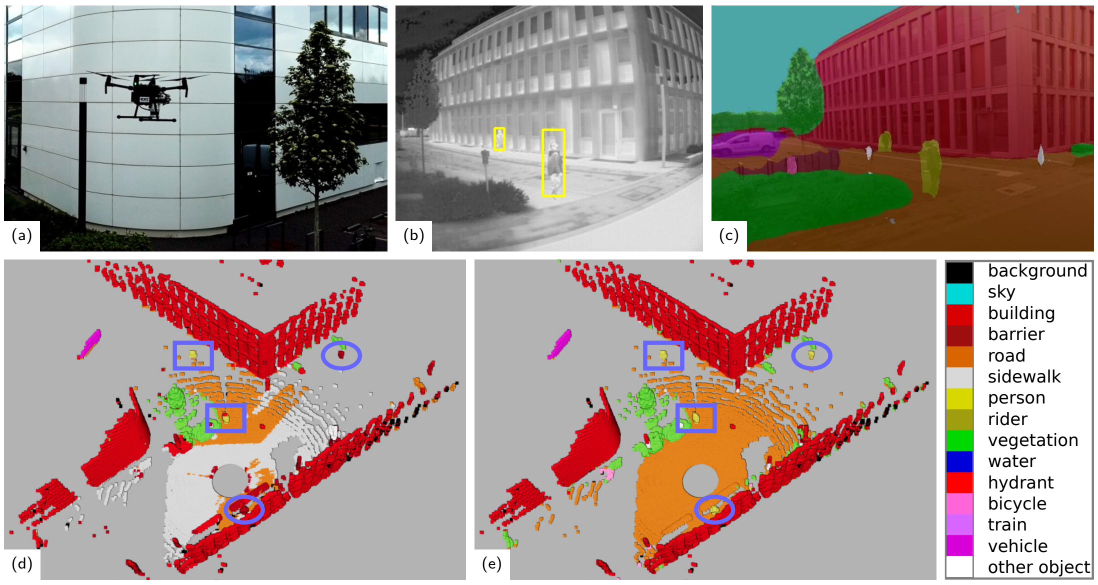

Figure 4.1: Semantic perception with UAV (a): (b) Person detections in thermal camera, (c) fused image segmentation, and point cloud segmentation (d) before and (e) after label propagation. Persons inside (outside) camera FoV are highlighted with blue rectangles (circles) in (d), (e). Right side: employed semantic classes.

using lightweight CNN architectures and embedded inference accelerators. We follow a late fusion approach where semantic information from multiple sensor modalities augments 3D point clouds and image segmentation masks while also generating an allocentric semantic map. Label propagation on the semantic map allows for sensor-specific adaptation with cross-modality and cross-domain supervision.

Our system provides augmented semantic images and point clouds with $\approx 9\,\text{Hz}$. We evaluate the integrated system in real-world experiments in an urban environment and at a disaster test site.

4.1 Introduction

Semantic scene understanding is an important prerequisite for solving many tasks with unmanned aerial vehicles (UAVs) or other mobile robots, e. g. for disaster examination in search and rescue scenarios (Kruijff-Korbayová et al., 2021), inspection, or surveillance tasks. Modern robotic systems employ a multitude of different sensors to perceive their environment, e. g. 3D light detection and ranging (LiDAR) scanners, RGB, RGB-D, and thermal cameras, that capture complementary information about the environment. A LiDAR provides accurate range measurements independent of the lighting conditions, while cameras provide dense texture and color in the visible spectrum. Thermal cameras are especially useful in search and rescue missions as they detect persons or other heat sources regardless of lighting or visibility conditions. The combination of all these sensor modalities enables a complete and detailed interpretation of the environment. A semantic map aids inspection tasks (Nguyen et al., 2019), perception-aware path planning (Bartolomei, Teixeira, and Chli, 2020), and increases robustness and accuracy of simultaneous localization and mapping (SLAM) through the exclusion of dynamic objects during scan matching (X. Chen et al., 2019). In the 2020 Mohamed Bin Zayed

International Robotics Challenge (MBZIRC 2020), robust real-time on-board semantic perception enabled our UAVs to autonomously track and chase different targets through visually guided control, i.e. stationary balloons anchored to the ground (Beul et al., 2020) and a dynamically moving target UAV with attached payload (Beul et al., 2022).

In this chapter, we propose a framework for online multi-modal semantic fusion onboard a UAV combining 3D LiDAR range data with 2D color and thermal images. We use label propagation for cross-modality supervision to significantly improve the LiDAR point cloud semantic segmentation compared to a baseline network pre-trained on a large automotive LiDAR-semantic dataset. Our method is evaluated with real-world UAV flights in an urban campus environment and on a disaster test site. We extend and adapt approaches from the previous chapters, where we proposed a network of multiple, stationary sensors with local, on-board perception in lab-scale indoor environments to the fundamentally different domain of large-area outdoor UAV flights, where we use a single, but moving sensor suite with different sensor modalities and perspectives. Examples of the semantic perception onboard the UAV are illustrated in Fig. 4.1.

An embedded inference accelerator and the integrated graphics processing unit (iGPU) run inference online, onboard the UAV for mobile optimized CNN architectures to obtain pixel- resp. pointwise semantic segmentation for RGB images and LiDAR scans, as well as object bounding box detections on RGB and thermal images. We aggregate extracted semantics for two different output views: (i) A fused segmentation mask for the RGB image which can, e. g. be streamed to the operator for direct support of their situation awareness, and (ii) a semantically labeled point cloud, providing a 3D semantic scene view which is further integrated into an allocentric map. This late fusion approach is beneficial for multi-rate systems, increasing adaptability to changing sensor configurations and enabling pipelining for efficient hardware usage. The semantic map further allows to adapt specific CNNs to new sensors with unique characteristics using cross-modality supervision from, e. g. thermal and color segmentation, through propagating labels via 3D projection.

In summary, our main contributions are:

- the adaptation of efficient CNN architectures for image and point cloud semantic segmentation and object detection for processing onboard a UAV using embedded inference accelerator and iGPU,
- the Bayesian fusion of semantic information from point cloud, RGB, and thermal modalities into a joint image segmentation mask and a semantically labeled 3D point cloud,
- using label propagation to overcome domain adaptation issues of the LiDAR segmentation network, thereby significantly improving the accuracy of point cloud segmentation, and
- the real-world deployment, proof of viability, and evaluation of the proposed system with large-area UAV flights in an urban environment and at a disaster test site.

4.2 Related Work

Lightweight Vision and Point Cloud CNNs for UAVs. Lightweight vision CNN architectures for efficient inference on embedded hardware have already been discussed in Sec. 3.2. These include e.g., replacing classical backbone networks such as ResNets (K. He et al., 2016) with MobileNets (Howard et al., 2019; Sandler et al., 2018) that replace standard convolutions with depthwise-separable convolutions, or using single-stage architectures such as SSD (W. Liu et al., 2016) or YOLO (Redmon et al., 2016) for object detection that use predefined anchors instead of additional region proposal networks.

P. Zhang, Zhong, and X. Li (2019) further optimize YOLOv3 for usage onboard a UAV through channel pruning of convolutional layers using L1 regularization to enforce sparsity. The network is trained and evaluated on a dataset with drone-specific aerial perspectives. However, the authors evaluate their network, called Slim-YOLOv3, only on a powerful discrete GPU which is not feasible for integration onboard a typical UAV. In our work, we employ a MobileDet detector optimized for embedded inference accelerators such as the Google Edge TPU, similar to the network used on the static smart edge sensors (cf. Chapter 3).

For semantic image segmentation, efficient architectures for inference onboard UAVs have mostly been proposed for specific applications, such as UAV tracking and visual inspection (Nguyen et al., 2019), or weed detection for autonomous farming (Sa et al., 2018). The DeepLab v3+ architecture (L.-C. Chen et al., 2018) shows state-of-the-art performance on large, general datasets and includes elements of MobileNet architectures such as depthwise-separable convolutions for efficient computation. In our work, we employ a DeepLab v3+ model with MobileNet v3 backbone for image segmentation.

For point cloud semantic segmentation, projection-based methods (Cortinhal, Tzelepis, and Aksoy, 2020; Milioto et al., 2019; C. Xu et al., 2020) utilize the image-like 2D structure of rotating LiDARs. This allows to perform efficient 2D convolutions and use well-known techniques from image segmentation. The downside of this approach is the restriction to single LiDAR scans in contrast to larger aggregated point clouds (Charles R Qi et al., 2021). In this work, we adopt the SalsaNext architecture (Cortinhal, Tzelepis, and Aksoy, 2020), trained on the large-scale SemanticKITTI dataset (Behley et al., 2019) for autonomous driving, as it shows a good speed-accuracy trade-off.

Multi-Modal Semantic Fusion. Mobile robotic systems, such as UAVs or self-driving cars, are often equipped with both camera and LiDAR sensors, as they provide complementary information. A LiDAR accurately measures ranges sparsely and independent of lighting conditions while cameras provide dense textures and colors. Hence, research focused on the fusion of camera and LiDAR for 3D detection and segmentation in the context of autonomous driving.

D. Xu, Anguelov, and Jain (2018) propose PointFusion, a two-stage pipeline for 3D bounding-box detection. It first processes a LiDAR scan with PointNet (Charles R. Qi et al., 2017) and an image with ResNet (K. He et al., 2016) independently, before fusing them on the feature level with a multi-layer perceptron (MLP).

Meyer et al. (2019) take a similar sequential feature-level fusion approach, addressing both 3D object detection and dense segmentation. The LiDAR scan is represented as a range image and the 360° horizontal FoV of the LiDAR is reduced to only 90°, as range and color image are cropped to the overlapping FoV for the feature-level fusion.

By projecting LiDAR points into the image and assigning segmentation scores of the pixels to the corresponding points, Sourabh Vora et al. (2020) propose to in-paint point clouds with image semantic segmentation. The augmented point cloud is then processed by a 3D object detection network.

LIF-Seg by L. Zhao et al. (2023) improves upon the LiDAR segmentation network Cylinder3D (Zhu et al., 2021) through early- and mid-level fusion with color images. Image patches around the projected points provide per-point color context for early-fusion. Additionally, mid-fusion concatenates semantic features from LiDAR and image, processed with Cylinder3D and DeepLab v3+, respectively, before refining them with an additional Cylinder3D-based sub-network to obtain the final semantic labels.

Semantic Mapping. Many high-level robotic tasks benefit from or require semantic information about the environment. For this, semantic mapping systems build an allocentric semantic environment model, anchored in a fixed, global coordinate frame.

SemanticFusion (McCormac et al., 2017) uses *surfels*, where a Gaussian approximates the point measurement distribution to model surfaces. The approach uses an RGB-D camera and builds on ElasticFusion (Whelan et al., 2015) for SLAM. A CNN generates pixel-wise semantic class probabilities from the color image. Their semantic fusion takes a Bayesian approach, storing a probability vector over all semantic classes per surfel, and assuming that individual semantic segmentations are independent. Kimera (Rosinol et al., 2021) is a modular metric-semantic stereo-inertial-SLAM framework. Its semantic mapping module builds truncated signed distance field (TSDF) maps of the surface geometry of room-scale indoor environments, adopting Voxblox (Oleynikova et al., 2017). It integrates semantic class probabilities per voxel via a similar Bayesian fusion approach. A metric-semantic mesh, with the argmax semantic class per vertex, is extracted from the TSDF volume via marching cubes. Grinvald et al. (2019) provide object-level information for higher-level reasoning by representing individual object instances of known and previously unseen classes in the semantic map. For long-term mapping within changing environments, Sun et al. (2018) propose Recurrent-OctoMap. Each cell within the OctoMap (Hornung et al., 2013) contains a long short-term memory (LSTM) fusing point-wise semantic features. All LSTMs share weights to learn a fusion model that generalizes over the entire map without over-fitting to individual cells.

Landgraf et al. (2020) compare different fusion strategies, namely labeling individual views followed by Bayesian fusion versus labeling a joint map in a single step. Both strategies show similar results with the view-based fusion more strongly influenced by noise in the depth measurements while the map-based fusion depends on correct poses.

Other works propose alternatives to the probabilistic Bayesian update for fusing semantic labels from multi-view 2D images into a 3D map. Mascaro, Teixeira, and Chli (2021) build a sparse diffusion graph connecting 2D pixels to 3D points and 3D points to their K nearest neighbors to propagate labels from a 2D image segmentation to the 3D model. After graph construction, iterative multiplication of the label matrix with a probabilistic transition matrix yields the diffused semantic labels. Berrio et al. (2022) use an adapted softmax weighting scheme based on class prevalence within SLIC superpixels to weight individual per-pixel class scores. Motion correction and masking of occluded points are further employed to improve the semantic projection accuracy.

For more specific application scenarios, Maturana, Arora, and Scherer (2017) propose to extend existing digital elevation maps (DEMs) with detections of cars from UAVs. For real-time service robotics applications, Dengler et al. (2021) propose an object-centric 2D/3D map representation including a 2D polygon of object shape and an object-oriented bounding box in the x-y-plane together with the center of mass and object point cloud. The approach uses an RGB-D camera and faster R-CNN (Ren et al., 2017) to detect objects from the color images. Small objects are geometrically segmented from the depth measurements via Euclidean clustering and projected onto the x-y-plane. Larger clusters, where multiple instances of the same object class may have been erroneously merged e.g. due to incorrect odometry, are refined after a certain number of fusion steps to separate them again. In surfel-based LiDAR mapping, a surfel's semantic class can be used to further improve the registration accuracy by penalizing associations of different semantic classes during scan matching and surfel update (X. Chen et al., 2019). In their approach called SuMa++, the projection-based RangeNet++ (Milioto et al., 2019) provides semantic class probabilities per point measurement.

Rosu, Quenzel, and Behnke (2020) propose to represent semantics and geometry at different resolutions. The scene is represented by a geometric mesh, extracted from an aggregated point cloud coupled with a semantic texture at independent resolution. The transfer from image segmentation masks to the semantic texture is enabled by the projection of mesh faces into the images. While projection and fusion happen in real time, the required mesh generation and UV-unwrapping are done in pre-processing. Semantic classes are sparsely represented in the texture that retains only a small number of classes with high probability and discards all others to meet GPU memory limitations and since only the argmax class is of interest for the output semantic scene model. The textured semantic mesh enables label propagation to generate pseudo ground-truth for retraining the image segmentation network to obtain more consistent segmentations between different views. While Rosu et al. only improve consistency within one modality, we use propagated labels for sensor-specific adaptation across modalities.

Domain Adaptation and Label Propagation. In real-world robotics scenarios, a lack of annotated training data is a major issue. In recent years, substantial research efforts have been made to develop domain adaptation techniques that help neural networks to transfer perceptual skills learned from widely-available standard datasets to application-specific target environments. This often includes adaptation to other sensors with differing characteristics, such as wavelength, resolution, or FoV, compared to the sensor used for capturing the source dataset. In this context, label propagation automatically provides annotations for the target domain in a semi-supervised manner, e. g. by projecting labels from one sensor modality to another.

A first line of work investigates domain adaptation between different LiDAR sensors and datasets. Langer et al. (2020) tackle domain transfer between LiDAR sensors with different sampling patterns (i.e. 64-beam vs. 32-beam) by fusing scans from the source data domain and raycasting into the target sensor to obtain transferred training examples. During retraining, weights are shared for source and target and geodesic correlation alignment prevents unwanted domain shift. Yi, Gong, and Funkhouser (2021) perform a similar adaptation between LiDAR sensor types but instead use a two-stage CNN, where first, scene completion obtains a denser canonical point cloud before labeling it in the second stage. Alonso et al. (2021) examine data alignment strategies to make different LiDAR datasets more similar and include an alignment loss between source and target dataset based on the KL-divergence. While these works efficiently handle different LiDAR resolutions, the evaluated sensors have similar FoVs, and datasets all stem from urban driving scenarios. Our work, in contrast, handles more drastic viewpoint changes to aerial UAV perspectives and a LiDAR with different resolution and a significantly larger vertical FoV compared to the source data domain.

Several recent works cope with the limited availability of annotated training data through label propagation. Z. Liu, X. Qi, and Fu (2021) use weak supervision to generate pseudo-labels for 3D data using partitioned super-voxels. A graph relates the super-voxels and propagates pseudo-labels to iteratively train two complementary networks for point segmentation and super-voxel relations. B. Liu et al. (2019) propagate labels for 2D image data from a small target data domain towards a large unlabeled set with a similarity function pretrained on a source domain.

Most closely related to our approach are methods that apply cross-modal label propagation from 2D images to 3D point clouds. Piewak et al. (2018) transfer semantic annotations automatically inferred by an image segmentation CNN from the closest image to point clouds by projection, taking linear ego-motion from wheel odometry into account. We use a similar approach to automatically obtain labels for point clouds from the image modality, but use a spatio-temporally aggregated 3D semantic map as pseudo-label source, instead of one or multiple individual cameras.

Jaritz et al. (2020) present a two-branch network for 3D semantic segmentation. Individual networks compute feature maps for LiDAR and camera before retaining only features at valid projected points in the camera FoV. In parallel to the concatenation of both feature maps before a fused segmentation head, each branch performs single-modality segmentation. During training, the single heads should mimic the fused output by minimizing a cross-modal loss based on the KL-divergence. This requires labels for both modalities within the source domain. After initial adaptation to the target data domain, e. g. a different dataset without labels, the generation of pseudo-labels in the target data domain and retraining from scratch provides further improvements. Similarly, Z. Wang et al. (2021) use a two-branch network for 3D bounding box detection from LiDAR and images. A gated adaptive fusion sub-network introduces point-wise projected image features into the LiDAR branch on every layer within the feature encoder. A KL-divergence loss regularizes class predictions between the image and LiDAR branch. Due to the close coupling between image and LiDAR modalities, the above two methods only use 3D points inside the camera FoV. Our method, on the other hand, segments all LiDAR points in the complete 360° horizontal FoV.

In our work, different networks process LiDAR scan, RGB, and thermal images individually. We adopt a projection-based approach similar to Sourabh Vora et al. (2020) for multi-modal fusion in a multi-rate system. When multiple modalities are available, class probabilities from different sensors are merged using Bayesian fusion. Semantic mapping integrates augmented point clouds into a sparse voxel hash-map with full class probabilities per voxel.

While being less popular in recent work, such a late fusion approach has important practical advantages for deployment on an integrated robotic system. Different FoVs and data rates are easy to handle and intermediate results, such as image segmentation or detections, are useful as stand-alone outputs. Pipelining allows for reducing the latency of sequentially executed individual networks during online operation. Furthermore, the

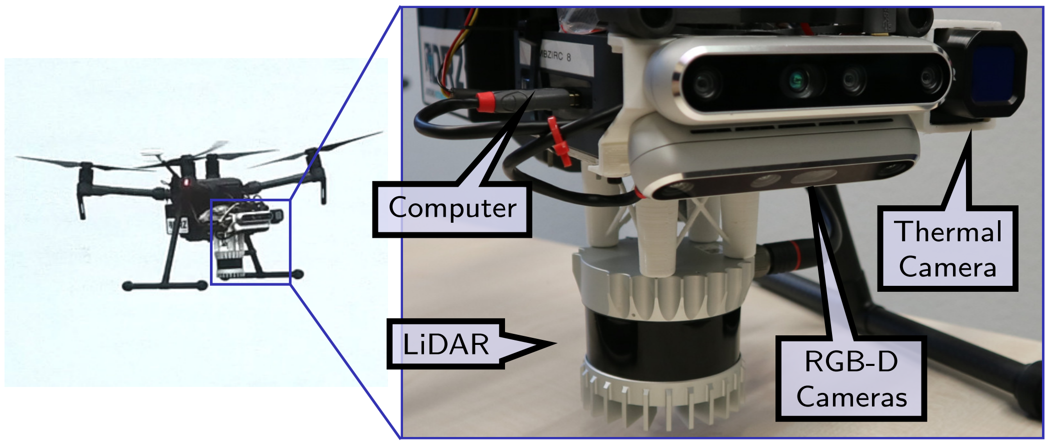

Figure 4.2: UAV system setup and hardware design.

smaller, simpler standard architectures of individual networks are easier to adapt and optimize for the embedded inference accelerators employed in this work.

As our target data domain of UAV aerial perspectives with large vertical FoV differs significantly from available large-scale training datasets from autonomous driving scenarios with different viewpoints and LiDAR sensors more focused towards the ground, we use label propagation to retrain the point cloud segmentation CNN with supervision for the target environment. Pseudo-labels are automatically obtained via 3D projection from RGB and thermal camera modalities, spatio-temporally aggregated into a semantic map to compensate for their narrower FoV compared to the LiDAR scanner. Through retraining with this cross-domain supervision, the point cloud segmentation is significantly improved, achieving higher mIoU scores on our dataset than the image segmentation used as pseudo-label source, and generalizing to the full LiDAR FoV.

4.3 Method

4.3.1 *System Setup*

An overview of our UAV system, based on the commercially available DJI Matrice 210 v2 platform, is shown in Fig. 4.2. We use an Intel Bean Canyon NUC8i7BEH with a Core i7-8559U processor and 32 GB of random-access memory (RAM) as the onboard computer. A Google Edge TPU is connected to the NUC over USB 3.0 and accelerates CNN inference together with the Intel Iris Plus Graphics 655 iGPU of the main processor. An Ouster OS0-128 3D-LiDAR[1] with 128 beams, 360° horizontal, and 90° vertical opening angles provides range measurements for 3D perception and odometry. For visual perception, our UAV additionally carries two Intel RealSense D455 RGB-D cameras, mounted on top of each other to increase the vertical field-of-view, and a FLIR ADK thermal camera for, e. g. person detection in search and rescue scenarios.

1 https://ouster.com/products/scanning-lidar/os0-sensor/, accessed: 2023-08-01

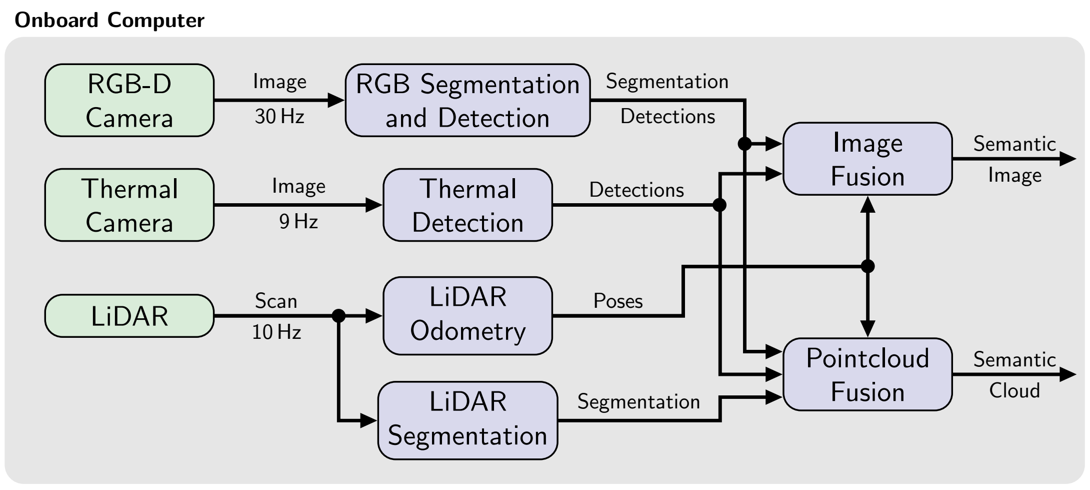

Figure 4.3: Online perception system overview.

4.3.2 *Semantic Perception*

An overview of the proposed architecture for multi-modal semantic perception is given in Fig. 4.3. We detail individual components in the following.

Image Segmentation

We employ the DeepLabv 3+ architecture (L.-C. Chen et al., 2018) with MobileNet v3 backbone (Howard et al., 2019) optimized for the Google Edge TPU Accelerator for semantic segmentation. We train the model on the Mapillary Vistas Dataset (Neuhold et al., 2017), reducing the labels to the 15 most relevant classes for the envisaged UAV tasks (cf. Fig. 4.1). We use an input image size of 849×481 px during inference, fitting the 16:9 aspect ratio of our camera.

Object Detection

The recent MobileDet architecture (Xiong et al., 2021) is the basis for our object detection. We train the RGB detector on the COCO dataset (Lin et al., 2014) for *person*, *vehicle*, and *bicycle* classes with an input resolution of 848×480 px. The thermal object detector uses the same architecture taking one-channel 8-bit thermal images at the full camera resolution of 640×512 px as input. We enable automatic gain correction (AGC) for the thermal camera to scale the 16-bit raw images down to 8-bit, exploiting the full 8-bit value range to provide contrast-rich images. The network is trained on the FLIR ADAS dataset[2], recorded with the previous generation of the employed sensor in autonomous driving scenarios, with annotations for *persons*, *vehicles*, and *bicycles*.

Point Cloud Segmentation

We adopt the projection-based SalsaNext architecture (Cortinhal, Tzelepis, and Aksoy, 2020) taking advantage of the image-like structure of LiDAR measurements. The network is pretrained on the large-scale SemanticKITTI dataset (Behley et al., 2019).

2 `https://www.flir.com/oem/adas/adas-dataset-form`, accessed: 2023-08-01

The OS0 LiDAR sensor provides measurements at a resolution of 1024×128. We compare using the full sensor resolution to subsampling the scans by a factor of two in both vertical and horizontal directions, leading to a network input resolution of 512×64. Subsampling enables real-time inference on our hardware. The input channels are range, x-, y-, z-coordinate, and intensity, normalized with the mean and standard deviation of the training dataset. Our LiDAR has a significantly larger vertical FoV of 90° compared to the 26.9° opening angle of the Velodyne HDL-64E sensor employed in the SemanticKITTI dataset. The HDL-64E mostly measures downward from the horizontal plane, thus seldomly measuring treetops or other higher obstacles. A different laser wavelength also changes the characteristics of intensity and reflections. Hence, we adjust the normalization parameters for z-coordinate and intensity to facilitate the cross-domain adaptation between training and observed data according to the statistics of the test data captured with our sensor setup, taking up the idea of input data distribution alignment from Alonso et al. (2021). The x- and y-coordinate normalization parameters remain the same, as the horizontal field-of-view is identical (360°) for both sensors. Fig. 4.11 highlights improvements of the segmentation results through the adaptation of the normalization parameters. The point cloud segmentation nonetheless remains noisier and less detailed than the image segmentation. To further overcome the domain adaptation issues, we retrain the point cloud segmentation network using label propagation (cf. Sec. 4.3.5).

Inference Accelerators

We run CNN model inference on two different accelerators onboard the UAV PC: The Google Edge TPU[3], attached as an external USB device, and the integrated graphics processing unit (iGPU) included in most modern processors which is otherwise unused in our system. The Edge TPU supports network inference via TensorFlow-lite (Abadi et al., 2016) and requires quantization of the network weights and activations to 8-bit (Jacob et al., 2018). The iGPU supports inference via the Intel OpenVINO framework[4] in 16- or 32-bit floating-point precision.

4.3.3 *Multi-Modality Fusion*

We adopt a projection-based approach to fuse semantic class scores from image and point cloud CNNs into the semantically labeled output cloud, similar to the semantic point cloud fusion on the stationary smart edge sensors (cf. Sec. 3.3.3), but additionally combining the LiDAR with the RGB and thermal modalities. Projection onto the image plane requires the transformation of LiDAR points into the respective camera coordinate frame. As we consider a moving sensor suite and LiDAR and cameras operate with different frame rates, the motion between the respective capture times has to be considered. The full transformation chain $\boldsymbol{T}$ from LiDAR to camera frame is:

$$\boldsymbol{T} = {}^{\text{cam}}\boldsymbol{T}_{\text{base}} \, {}^{\text{base}_{t_c}}\boldsymbol{T}_{\text{base}_{t_l}} \, {}^{\text{base}}\boldsymbol{T}_{\text{LiDAR}} \,, \tag{4.1}$$

using the continuous-time trajectory of the UAV base frame estimated by the LiDAR odometry (Quenzel and Behnke, 2021). Thus, the transformation chain models perspective changes between LiDAR and camera that occur due to dynamic UAV motions.

3 https://coral.ai/docs/accelerator/datasheet, accessed: 2023-08-01
4 https://docs.openvinotoolkit.org/, accessed: 2023-08-01

Bilinear interpolation at the projected point location gives the semantic class scores $\boldsymbol{c}_{\text{img}} \in \mathbb{R}^C$ from image segmentation. We apply the soft-max operation (Eq. (3.1)) to approximate a normalized probability distribution over all $C = 15$ classes used in this chapter (cf. Fig. 4.1), obtaining $\boldsymbol{p}_{\text{img}} \in \mathbb{R}^C$, with its elements $p_i \in [0, 1]$ and $\sum_i p_i = 1$. Similarly, the application of soft-max to the output of the point cloud CNN for a LiDAR point gives the LiDAR segmentation probability $\boldsymbol{p}_{\text{LiDAR}}$. The Bayesian update rule (McCormac et al., 2017) allows to compute the fused class probability under the assumption of independence between sensor modalities:

$$\boldsymbol{p}_{\text{fused}} = \frac{\boldsymbol{p}_{\text{img}} \circ \boldsymbol{p}_{\text{LiDAR}}}{\sum_{i=1}^{C} p_{i,\text{img}}\, p_{i,\text{LiDAR}}}\,, \tag{4.2}$$

with $\circ$ the coefficient-wise product. For better numerical stability, we use a logarithmic implementation of the Bayesian fusion (Bultmann, Quenzel, and Behnke, 2023).

Furthermore, if a projected point falls inside a detection box in either thermal or color images, the detected class is included in the result. We base the detection probability p_{det} on the detector score multiplied with a Gaussian factor with mean at the bounding box center and standard deviation of half the bounding box width resp. height, to reduce unwanted border effects for non-rectangular or non-axis-aligned objects. As the detector only outputs a score for the detected class, we reconstruct the full probability distribution $\boldsymbol{p}_{\text{det}}$ following the maximum entropy principle: The remaining probability mass $1 - p_{\text{det}}$ is equally distributed over the remaining $C - 1$ classes. Again, both estimates are fused using a Bayesian update:

$$\boldsymbol{p}_{\text{fused\_det}} = \frac{\boldsymbol{p}_{\text{fused}} \circ \boldsymbol{p}_{\text{det}}}{\sum_{i=1}^{C} p_{i,\text{fused}}\, p_{i,\text{det}}}\,. \tag{4.3}$$

However, side-effects of the rectangular detection bounding boxes have to be handled before detection fusion, as illustrated in Fig. 4.4 with the example of fusing person detections from the RGB or thermal image into the point cloud. Simple projection of all points into the bounding box will falsely label points in the background as the detected class (cf. Fig. 4.4 (b)). To alleviate this issue, points are clustered w.r.t. their distance in the camera frame before detection fusion. As a baseline approach, we include only points within a fixed threshold of the 25 % quantile of distances per cluster to focus on foreground objects (cf. Fig. 4.4 (c)). We extend this approach to Euclidean point clustering with an adaptive cluster tolerance threshold τ_{cluster}. This yields a more accurate segmentation and better generalizes to different object sizes and distances. Starting from the seed point at 25 % quantile distance d_{seed}, bounding box points within the distance τ_{cluster} are recursively added to the cluster. The cluster tolerance is proportional to d_{seed} and the angle increment between two adjacent scan lines, adapting to the LiDAR spatial resolution which covers a vertical FoV of 90° with 128 lines:

$$\tau_{\text{cluster}} = s \cdot d_{\text{seed}} \cdot \frac{\text{FoV}_{\text{vert}}}{\text{Res}_{\text{vert}}} = s \cdot d_{\text{seed}} \cdot \frac{\pi}{2 \cdot 128}\,, \tag{4.4}$$

where $s = 1.5$ is a tolerance factor set to yield complete foreground clusters without adding points on the floor or background (cf. Fig. 4.4 (d)). We assume that there is only one valid foreground cluster per bounding box. Only the clustered points are included into detection fusion. Furthermore, points not added to the cluster that are erroneously labeled as the cluster class, are reset to their original class probability from LiDAR segmentation to correct for border effects in the previous fusion stage (cf. background in

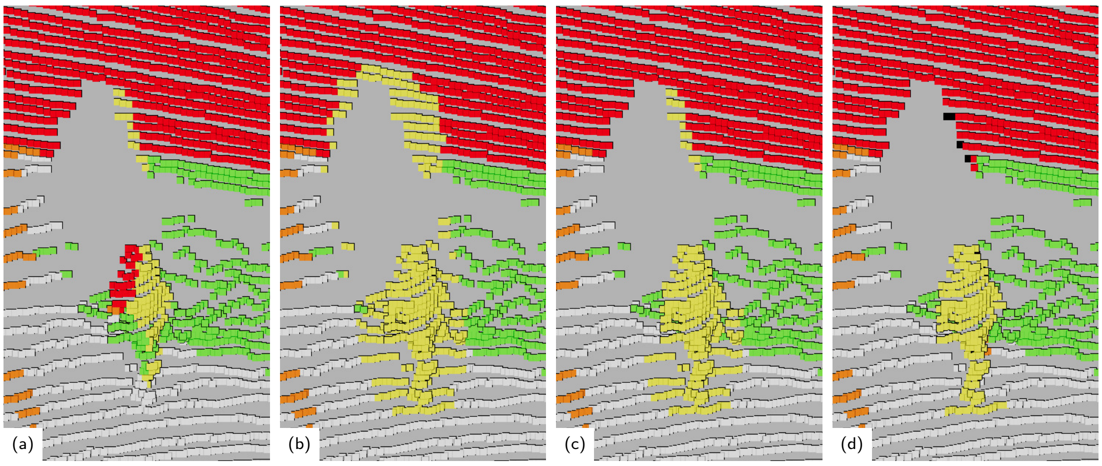

Figure 4.4: Person segmentation included into point cloud fusing (a) image segmentation only, (b) additionally detection bounding boxes, (c) clustering foreground points within the detection bounding boxes via depth threshold, and (d) Euclidean clustering. The initial segmentation (a), is incomplete and slightly misaligned. Naive bounding box fusion (b) creates many false positives in the background and on the floor. With depth threshold clustering (c), the person is completely segmented without adding many misclassified points in the background. With Euclidean clustering (d), the person is completely segmented and misclassified points in the background and on the floor are reset to their original label from LiDAR segmentation.

Fig. 4.4 (a) vs. (d)). The final segmented point cloud includes the full class probability vector and the argmax class color per point.

We proceed similarly for fusing the initial image segmentation and detections from RGB and thermal cameras into the output semantic image and additionally apply temporal smoothing. The RGB-D depth enables projection from RGB to thermal image and temporal smoothing provides a more coherent fused segmentation. For temporal fusion, we project the previous image at time $t-1$ with its depth into the current frame at time t and perform exponential smoothing:

$$\boldsymbol{p}_{\text{smoothed\_img}_t} = \text{normalize}\left(\boldsymbol{\alpha} \circ \boldsymbol{p}_{\text{img}_t} + (\mathbf{1} - \boldsymbol{\alpha}) \circ \boldsymbol{p}_{\text{fused\_img}_{t-1}}\right) , \tag{4.5}$$

$$\boldsymbol{p}_{\text{fused\_img}_t} = \frac{\boldsymbol{p}_{\text{smoothed\_img}_t} \circ \boldsymbol{p}_{\text{det}_t}}{\sum_{i=1}^{C} p_{i,\text{smoothed\_img}_t}\, p_{i,\text{det}_t}} . \tag{4.6}$$

The smoothing weights $\boldsymbol{\alpha}$ differ between the individual semantic classes. For (potentially) dynamic foreground objects, such as *persons* and *vehicles*, less smoothing is applied than for static structures such as buildings and roads. We chose $\alpha_{\text{dyn}} = 0.80$ for dynamic object classes and $\alpha_{\text{stat}} = 0.25$ for static background classes in our experiments. The higher α-coefficients for dynamic objects make the fused segmentation mask correctly follow dynamic foreground objects moving over image areas that were previously segmented as a background class, as the current frame's segmentation more directly influences the fused output for these semantic classes. The smoothing for background classes significantly reduces temporal jitter in the segmentation w.r.t. the initial CNN output. The temporal smoothing (4.5) can only be applied to pixels with valid depth measurements that have a corresponding projected point from the previous image. We directly use the segmentation class probabilities from the current frame $\boldsymbol{p}_{\text{img}_t}$ for fusion with the detections in (4.6) otherwise.

4.3.4 *Semantic Mapping*

We again employ a voxel-based representation of the environment as a semantic map, using sparse voxel hashing (Quenzel and Behnke, 2021) as a memory-efficient data structure, since a dense voxel grid may require a prohibitively large amount of memory although only sparse access occurs. In contrast to the previous scenario (cf. Chapter 3), where multiple stationary RGB-D sensors with calibrated poses provided the measurements to fill the map, here we use point clouds from a single, moving LiDAR sensor mounted on the UAV as input. The input point clouds are augmented with semantic class probabilities fused from multiple sensor modalities, as described above.

MARS LiDAR Odometry (Quenzel and Behnke, 2021) provides the sensor poses to integrate all augmented point clouds within a common map. The map origin is defined at the starting pose of the UAV. For each voxel, occupancy count and mean position are computed based on the corresponding point measurements. Only voxels with at least one associated measurement are instantiated into memory.

The semantic class probabilities of all point measurements falling into a voxel are fused in a probabilistic manner following the reasoning of SemanticFusion (McCormac et al., 2017) to use Bayes' rule assuming independence between semantic segmentations of the individual point measurements in the augmented point clouds (Eq. (3.5)). A naive implementation, as in SemanticFusion, suffers from numerical instability due to the finite floating-point precision of the multiplication of the normalized class probability vectors with multiple entries in the numerator of Eq. (3.5). In practice, this can lead to information loss also after the application of the normalization term and needs continuous re-initialization. An implementation using logarithmic probabilities (Bultmann, Quenzel, and Behnke, 2023) alleviates this issue.

4.3.5 *Label Propagation*

The employed LiDAR segmentation CNN, pretrained on the SemanticKITTI dataset (Behley et al., 2019), shows limited performance in our test scenarios (cf. Sec. 4.4.2). This is due to cross-domain adaptation issues between training and observed data, as the UAV employs a LiDAR sensor different from SemanticKITTI, with different vertical FoV, laser wavelength, and optical system. To the best of our knowledge, no large-scale semantically annotated training datasets are available using the employed Ouster OS0-128 sensor.

To overcome these issues, we retrain the CNN using our sensor's FoV parameters by (1) complementing the SemanticKITTI training data with the recently published Paris-CARLA-3D dataset (Deschaud et al., 2021) and (2) automatically generating pseudo-labels for cross-modal supervision from the fused semantics of RGB and thermal camera from outdoor flights with our UAV system.

The Paris-CARLA-3D dataset contains aggregated point clouds from three streets in Paris over about 550 m linear distance and a Velodyne HDL32 LiDAR sensor similar to the one used for SemanticKITTI. However, the sensor was tilted, allowing high structures such as buildings to be fully mapped. We only utilize the real-world part of the dataset. To obtain labeled single scans to complement the training data, we project points from the aggregated cloud into simulated viewpoints with the characteristics of the Ouster OS0 LiDAR at positions following the original vehicle trajectory from the

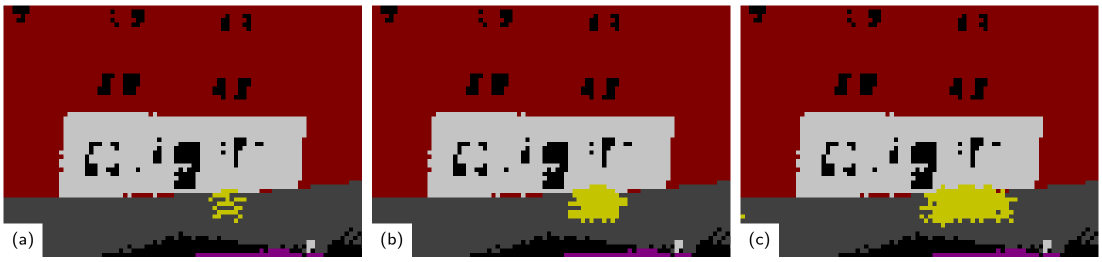

Figure 4.5: Paris-CARLA-3D dataset: Projection of dynamic persons (yellow) into the virtual OS0 LiDAR view using dataset scans from (a) only the current, (b) ±2, and (c) ±5 adjacent positions. The foreground person is incomplete when using just a single scan (a) but significant movement artifacts appear for a larger scan window (c). We use a window of ±2 (b) for our experiments as a compromise between complete scans and remaining artifacts.

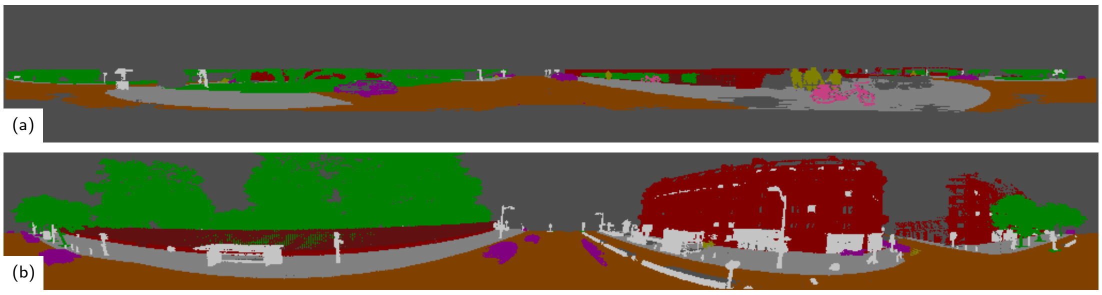

Figure 4.6: Point cloud labels projected into the FoV of the employed Ouster OS0 LiDAR for sample scenes from (a) SemanticKITTI (Behley et al., 2019) and (b) Paris-CARLA-3D (Deschaud et al., 2021) datasets. SemanticKITTI covers a significantly smaller vertical FoV than our sensor, while the aggregated cloud from Paris-CARLA-3D covers the entire FoV.

dataset, but at larger height, further adapting to our UAV use case. The projection of a LiDAR point $(x, y, z)^\intercal$ in the sensor frame to image coordinates $(u, v)^\intercal$ is given as in (Cortinhal, Tzelepis, and Aksoy, 2020) by:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0.5\left(1 - \operatorname{atan2}(y, x)\,\pi^{-1}\right) w \\ \left(1 - \left(\arcsin\left(zr^{-1}\right) + f_{\text{down}}\right) f^{-1}\right) h \end{pmatrix}, \tag{4.7}$$

with h, w denoting the height, resp. width of the projected image, $r = \sqrt{x^2 + y^2 + z^2}$ the range of each point and $f = |f_{\text{down}}| + |f_{\text{up}}|$ the vertical field-of-view ($f_{\text{down}} = f_{\text{up}} = 45°$ for OS0). When multiple points are projected to the same image coordinates, only the closest one is retained. The maximum range is set to 50 m.

To avoid artifacts from dynamic objects (e.g. cars and persons), points of these semantic classes are only projected from a limited number of scans captured close to the current simulated position. Here, a compromise must be made between complete, dense scans and remaining artifacts as illustrated in Fig. 4.5. We choose a window of ±2 scans for our experiments.

The differences between the two datasets are illustrated in Fig. 4.6, where labels from SemanticKITTI and Paris-CARLA-3D are shown projected in the Ouster OS0 sensor FoV. The SemanticKITTI data covers only a small part of the vertical FoV with the top of buildings rarely visible, while the scans obtained from Paris-CARLA-3D cover

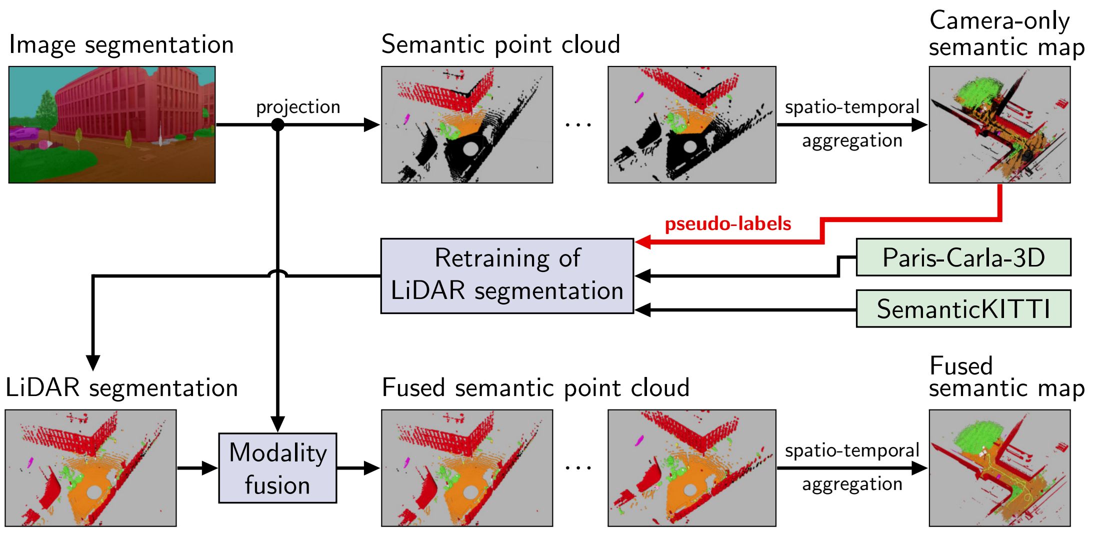

Figure 4.7: Overview of label propagation approach: Image segmentation is projected to point cloud and aggregated into camera-only semantic map, used as source for pseudo labels. LiDAR segmentation is retrained combining own data with pseudo-labels, Paris-CARLA-3D, and SemanticKITTI. Resulting semantic point clouds, fusing image and retrained point cloud segmentation, are completely and accurately annotated.

the full FoV and show complete building structures, similar to the data observed by the UAV system. Yet, the variability of Paris-CARLA-3D (550 m distance, single city area) is limited compared to SemanticKITTI (39.2 km distance, different urban, rural, and highway areas (Geiger, Lenz, and Urtasun, 2012)), and some artifacts from moving objects remain. Furthermore, as a different LiDAR sensor is used, the intensity channel, computed from the magnitude of laser reflection, is still only partially comparable to the Ouster OS0 which uses a different wavelength and optical system.

To obtain training data from the actual sensor, we use label propagation for cross-modal domain adaptation to automatically generate pseudo-labels for data captured during flights with our UAV system. An overview of the proposed approach is given in Fig. 4.7. We again take up the idea of semantic feedback at this point (red arrow): Parts of the fused allocentric semantic model are reprojected into individual sensor views and used as a reliable source of semantic information (i.e., more reliable than individual single-scan segmentation).

We only use the camera semantics as a pseudo-label source because the semantic information from the RGB and thermal modalities is significantly more reliable than the initial point cloud segmentation (cf. Sec. 4.4.2). For this, we fuse the RGB and thermal camera semantic information into the point cloud, without including the outputs of the initial, SemanticKITTI-pretrained LiDAR segmentation CNN. The obtained pseudo-labels are illustrated in Fig. 4.8. As the FoV of the cameras is significantly smaller than that of the LiDAR, only a small part of each individual scan can be labeled with this cross-modal supervision. However, after aggregation over the complete flight, the semantic map can provide pseudo-labels for almost the complete scan. Only the operator of the UAV (person to the bottom right of Fig. 4.8) is not annotated as they always stayed behind the UAV and never were in the camera FoV. For reference, we also compare to using pseudo-labels from the map aggregated from fused semantic clouds. We find, however, that this supervision is too imprecise to achieve significant improvements

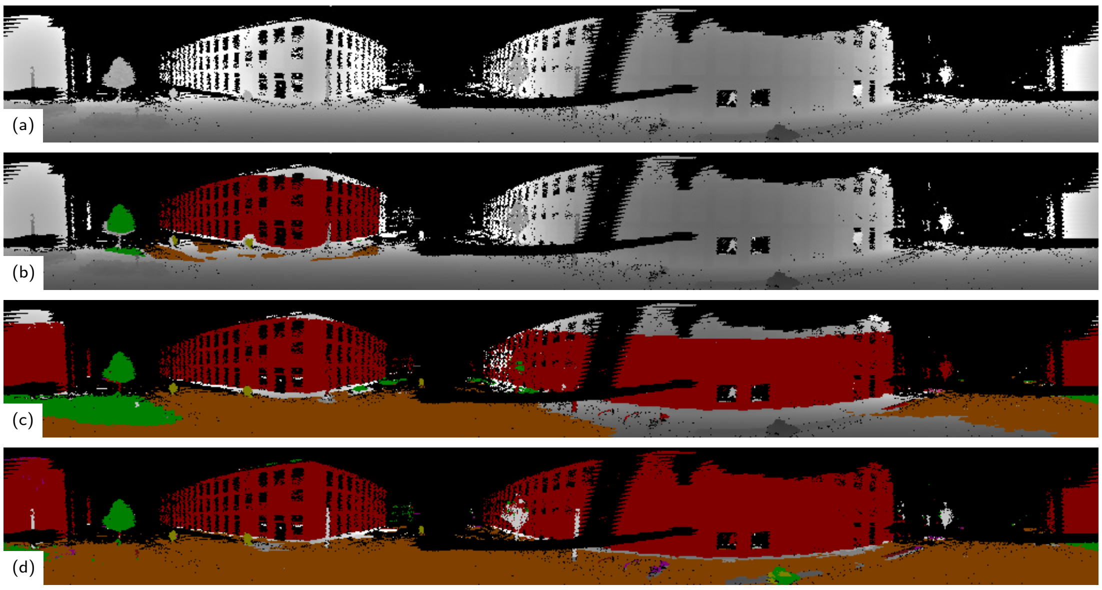

Figure 4.8: Label propagation using cross-modal supervision: (a) range channel of projected scan, (b) pseudo-labels from single camera overlay, (c) from aggregated camera-only map, and (d) from aggregated fused map (cf. Fig. 4.13 (b, c)). Unlabeled areas are depicted in grayscale. UAV legs lead to areas without valid measurements (black) at close range. Single camera overlay provides supervision only for a small part of the scan. Supervision from aggregated maps is more complete and the camera-only map (c) is more accurate than the fused map (d).

(cf. Tab. 4.5) since the noisier raw point cloud segmentation is included (e. g. wrongly labeled vegetation to the left and person to the bottom right of Fig. 4.8 (d)). It is crucial that the pseudo-labels used for re-training are as accurate as possible and it is better to leave uncertain parts unlabeled than to fill them with imprecise labels.

The semantic map which serves as source for pseudo-labels was generated fully automatically from captured data, using the proposed system for multi-modality fusion and semantic mapping, without any manual supervision. Additionally, a single, automated post-processing step improves the label quality: Points on the ground plane that are not labeled as a ground class (i.e. *road*, *sidewalk*, or *vegetation*) are reset to *background*, to correct for unwanted artifacts at object borders. Similar to Rosu, Quenzel, and Behnke (2020), we use only high-confidence pseudo-labels (minimum confidence of 80 %) leading to unlabeled gray regions between areas of different semantic classes in Fig. 4.8 (b). After aggregation of multiple viewpoints in the semantic map, most labels have confidence close to 1 and the borders between semantic classes are sharp.

For retraining the point cloud segmentation network, we complement the SemanticKITTI training data with scans obtained from Paris-CARLA-3D and self-recorded UAV flights with pseudo-labels. The amount of additional data is chosen to be comparable to the original $\approx$ 20k scans of SemanticKITTI, as proposed by Rosu, Quenzel, and Behnke (2020). We increase the batch size to 64 to compensate for the lower signal-to-noise ratio due to the noisier pseudo-labels. Furthermore, we increase the magnitude of the data augmentation transformations w.r.t. the original SalsaNext training parameters (Cortinhal, Tzelepis, and Aksoy, 2020), to account for the lower variability of the additional scenes. After retraining, we compare networks using the five original input channels (range, x-, y-, z-coordinate, and intensity) against ones not using intensity information,

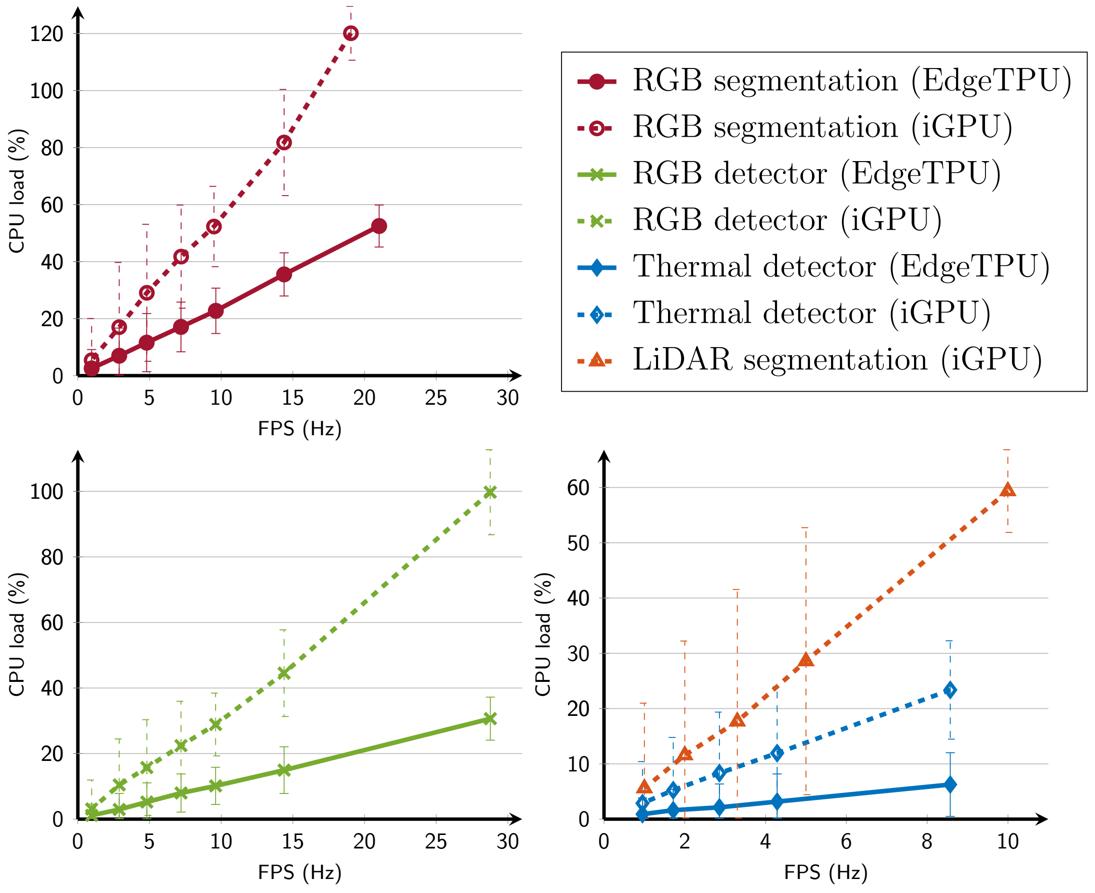

Figure 4.9: CPU load of the CNN inference of different models depending on the used accelerator and the output frame rate. The iGPU (dashed lines) results in higher CPU load than the Edge TPU (solid lines) for all models.

as this channel is the most difficult one to adapt for between changing sensor types. If not stated otherwise, the network is retrained with an input resolution of 512×64, using SemanticKITTI, Paris-CARLA-3D, and scans from our own data collection with pseudo-labels from the camera-only semantic map. Different parameters are compared in the ablation studies in Sec. 4.4.

4.4 Evaluation

We first evaluate inference speed and computational efficiency of the employed CNN models and then show results from outdoor UAV flights in an urban environment and on a disaster test site.

4.4.1 *CNN Model Efficiency*

In real-time systems with limited computational resources, such as UAVs, efficiency is of key importance and resources need to be distributed with care between the different system components. Semantic perception, while important for many high-level tasks, has less severe real-time constraints than, e. g. flight control or odometry. It is thus important that the CNN inference uses as few central processing unit (CPU) resources as possible

Table 4.1: Average inference time on the respective accelerator.

Model	Input Resolution	Edge TPU	iGPU
RGB segmentation	849×481	40.5 ms	50.0 ms
RGB detector	848×480	17.5 ms	24.0 ms
Thermal detector	640×512	12.0 ms	18.0 ms
LiDAR segmentation	512×64	-	32.0 ms
LiDAR segmentation	1024×128	-	140.0 ms

Table 4.2: Average CPU load and output frame rate of different model combinations. Image segmentation and detection models run on Edge TPU and point cloud segmentation on iGPU.

point cloud fusion	image fusion	point cloud seg.	thermal det.	RGB det.	RGB seg.	CPU load	avg. FPS (image)	avg. FPS (cloud)
-	-	-	-	-	✓	52.5 %	21.0 Hz	-
-	-	-	-	✓	✓	54.2 %	13.2 Hz	-
-	-	-	✓	✓	✓	57.3 %	12.6 Hz	-
-	✓	-	✓	✓	✓	120.6 %	9.9 Hz	-
-	-	✓	✓	✓	✓	116.6 %	12.6 Hz	10.0 Hz
-	✓	✓	✓	✓	✓	180.0 %	9.5 Hz	10.0 Hz
✓	✓	✓	✓	✓	✓	204.3 %	8.9 Hz	9.5 Hz

to not interfere with the hard real-time constraints of low-level control, localization, and state estimation. For this, we analyze the CPU load of the employed CNNs for object detection and segmentation, depending on the used accelerator. Although the main computational load of inference is distributed to a dedicated accelerator (Edge TPU or iGPU), the preparation of input data, data transfer, and post-processing require CPU resources. This is handled with differing degrees of efficiency w.r.t. CPU load and depends on the in- and output frame rate, as shown in Fig. 4.9. Models running on the Edge TPU produce lower CPU load in all cases while achieving higher or equivalent maximum frame rates. Tab. 4.1 shows the average inference latency per model. The LiDAR segmentation is only executed on the iGPU, as the *pixel-shuffle* layer from SalsaNext (Cortinhal, Tzelepis, and Aksoy, 2020) is not supported by the Edge TPU and the model thus cannot be converted to the required 8-bit quantized format.

For the following experiments, we choose to run the image CNNs on the Edge TPU, while the LiDAR segmentation runs on the iGPU at 512×64 input resolution. Tab. 4.2 shows the average computational load and output rate for different combinations of CNNs. As to be expected, the maximum achievable output frame rate drops and CPU load increases with a growing number of vision models used. The computation of RGB segmentation and detections, as well as thermal detections, achieves an average frame rate of 12.6 Hz at a CPU load of about 60 %. The inclusion of the image fusion module

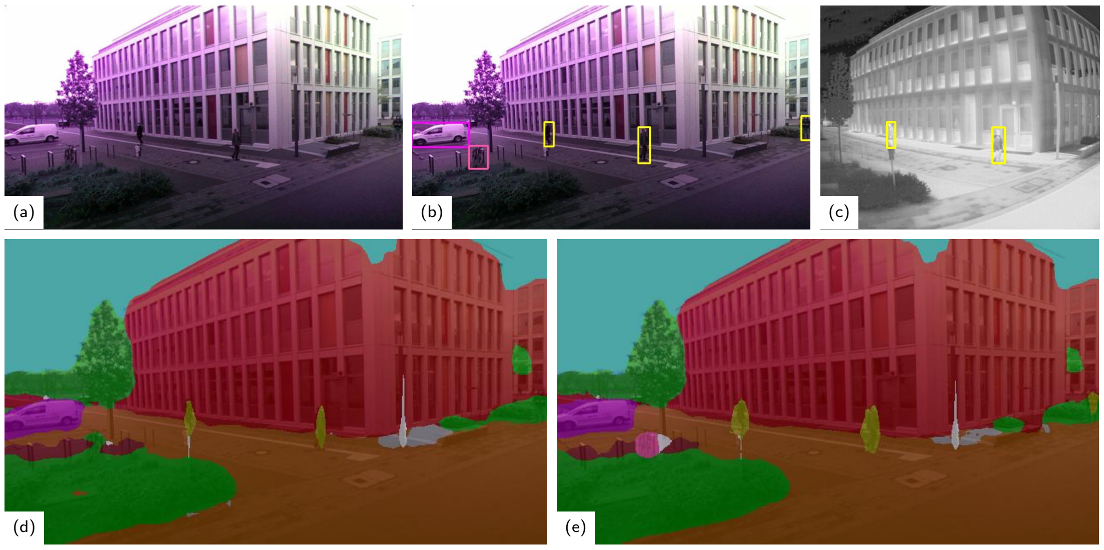

Figure 4.10: Semantic interpretation of RGB and thermal images: (a) RGB input image, (b) RGB and (c) thermal detections. (d) RGB segmentation. (e) fused segmentation mask. Persons and bicycle, not or only partially segmented in the initial segmentation mask, are fully visible in the fused output.

almost doubles the CPU utilization while the frame rate drops to 9.9 Hz. This is due to the transformations and projections necessary to calculate at image resolution for temporal smoothing and to include thermal detection into the fused image segmentation. The total CPU usage for the fusion of both image and point cloud semantics sums up to about 2 CPU cores with an output rate of around 9 Hz.

Reducing the input frequency of the semantic segmentation and detection can free additional resources for other system components if necessary while still providing semantic image and point cloud, e. g. at 1-5 Hz—sufficient for many high-level tasks like planning or keyframe-based mapping.

4.4.2 *Outdoor Experiments*

In Fig. 4.10, we show results of semantic image fusion for an exemplary scene from our test flights. Fig. 4.10 (b) - (d) show the outputs of the individual CNNs. While the large structures are well segmented (d), the persons are only partially recognized. A bicycle and the person at the right image border are missed altogether. The RGB detector (b) recognizes all persons and the bicycle. The thermal detector (c) confirms both person detections inside the thermal camera's FoV. The fused output segmentation mask (e) includes all detections together with the initial segmentation. All persons and the bicycle are clearly visible.

Fig. 4.11 shows the point cloud segmentation results for the same scene. When using the original LiDAR segmentation, without adaptation of the normalization parameters, parts of buildings are misclassified as vegetation or vehicle. This is likely due to differing vertical field-of-views of our and the trained LiDAR. In the KITTI dataset (Geiger, Lenz, and Urtasun, 2012), the FoV is only 2° upwards and ≈ 25° downwards (compared to ±45° of our sensor). Therefore, in SemanticKITTI the top of building structures is rarely visible while treetops are measured from below. Furthermore, the intensity input

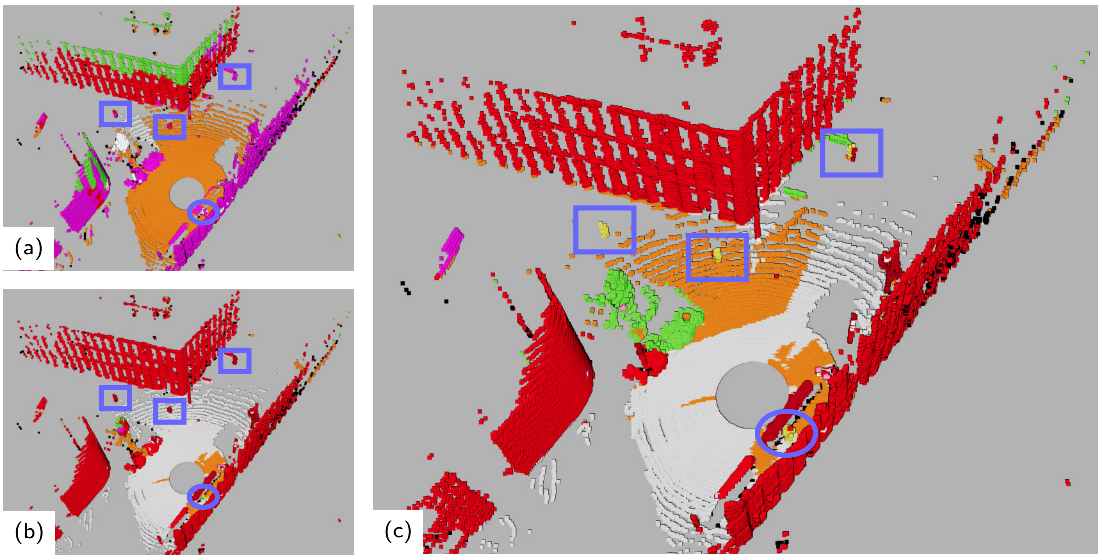

Figure 4.11: Point cloud segmentation: (a) Initial LiDAR segmentation without adaptation of z and intensity normalization parameters and (b) after adaptation to our dataset mean and std. (c) fused point cloud segmentation. Persons inside (outside) camera FoV are highlighted with blue rectangles (circles). After normalization adaptation, the CNN segments large structures well within the LiDAR scan but misses small objects. Persons, vegetation, and small structures are well segmented in the fused output scan inside the camera FoV.

channel, measured as the magnitude of laser reflection, differs between the sensors, as they use a different wavelength and optical system. After normalization adaptation, SalsaNext segments the building and road structures well within the LiDAR scan, and the person closest to the sensor is recognized (Fig. 4.11 (b)). Independent of the normalization, the point cloud network does not detect persons at larger distances, often misclassifying them as barrier or building. Fig. 4.11 (c) shows the fused point cloud segmentation, combining image segmentation and detections with the initial point cloud segmentation. Persons, also at larger distances, vegetation, and the car are well segmented in the output scan and exhibit less noise when inside the camera FoV.

The point cloud segmentation after retraining with label propagation is shown in Fig. 4.12. The scene is segmented very accurately, including persons and small structures, even when using the LiDAR segmentation alone, without fusing the image semantics. The difference between including intensity as input channel or not becomes apparent for the UAV operator (person to the bottom of the scene), who was not annotated in the pseudo-labels used as supervision for label propagation (cf. Sec. 4.3.5). Without intensity input, the generalization works better and this person is also correctly segmented. Additional fusion with the camera semantics makes only little difference after retraining. The LiDAR CNN has learned the relevant segmentation skills from cross-modal supervision.

Figure 4.13 depicts the aggregated semantic map of the outdoor test flight with manually annotated semantic labels (a) and with scans either labeled from image segmentation (b) or fused semantic point clouds (c, d). The camera-only map (b) misses annotations due to the camera's limited FoV but depicts most classes, such as persons, cars, or vegetation, more accurately since the noisier raw point cloud segmentation is not included. The tracks of moving persons are visible in yellow on the maps. Only the

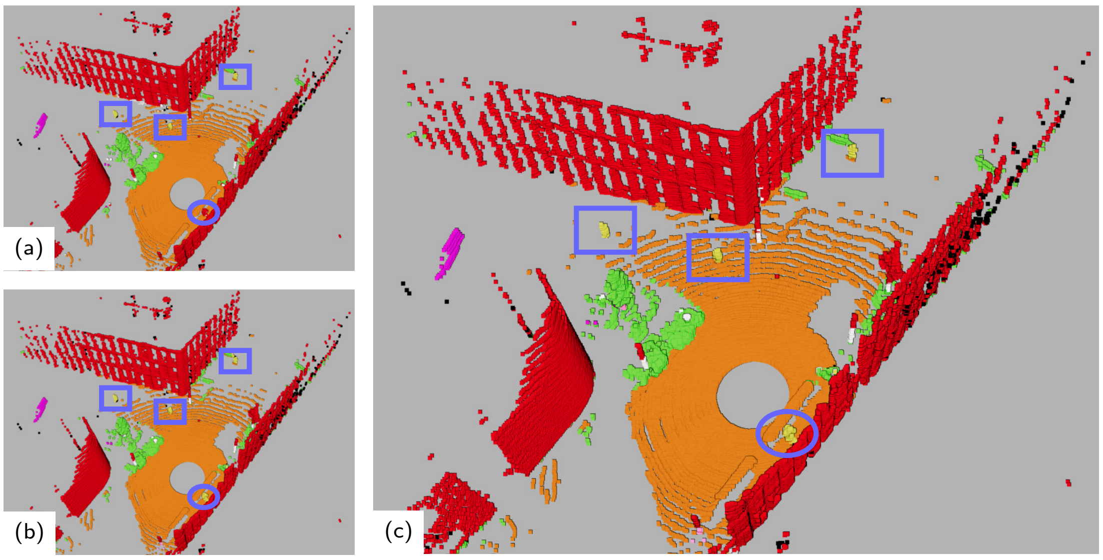

Figure 4.12: Point cloud segmentation after retraining with label propagation: (a) LiDAR segmentation with intensity input channel and (b) without intensity. (c) fused point cloud segmentation (based on (b)). Persons inside (outside) camera FoV are highlighted with blue rectangles (circles). The scene is accurately segmented in all cases, including persons and small structure. Without intensity input (b) the CNN generalizes better and the bottom person is correctly segmented. Fusing with image semantics gives little gain after retraining.

track of the operator, who always stayed behind the UAV and thus was not visible in the camera, is not segmented (b) or mislabeled (c). The direct segmentation of persons or small structures in the LiDAR scans initially is noisy due to domain adaptation issues with the CNN. Label propagation for cross-domain supervision is employed to overcome these issues. We use the accurate, but incomplete camera-only map as source for pseudo-labels for retraining the LiDAR segmentation. The resulting semantic map, Fig. 4.13 (d), aggregated from fused semantic point clouds using the retrained LiDAR CNN, depicts the semantics of the entire scene very accurately with the person tracks completely segmented, including the parts unlabeled in the camera-only map. This underlines the efficiency of label propagation and shows the generalization capabilities of the resulting CNN. Note, that the manually annotated map is not included during retraining and is only used for evaluation. For visual assessment of the map quality, Fig. 4.14 depicts detailed closeups of the static parts of the final semantic map after removing the dynamic person tracks.

The aggregated maps from fused semantic clouds of two further experiments are shown in Fig. 4.15 and Fig. 4.16. For the first flight, the scene semantics are shown (a) before and (b) after retraining with label propagation. Before retraining, person detection works sufficiently well within the camera frustum (right part of the scene), while they are misclassified as vehicle or vegetation elsewhere. After retraining, person tracks are segmented in all parts of the scene and the lawn is correctly labeled as vegetation. The semantic map from the second, significantly longer UAV flight around the university campus, using the retrained LiDAR CNN, is shown in Fig. 4.16. A coherent 3D semantic representation of the environment can be created also at large scale by our proposed system. Please note, that data from these two flights was not used for label propagation.

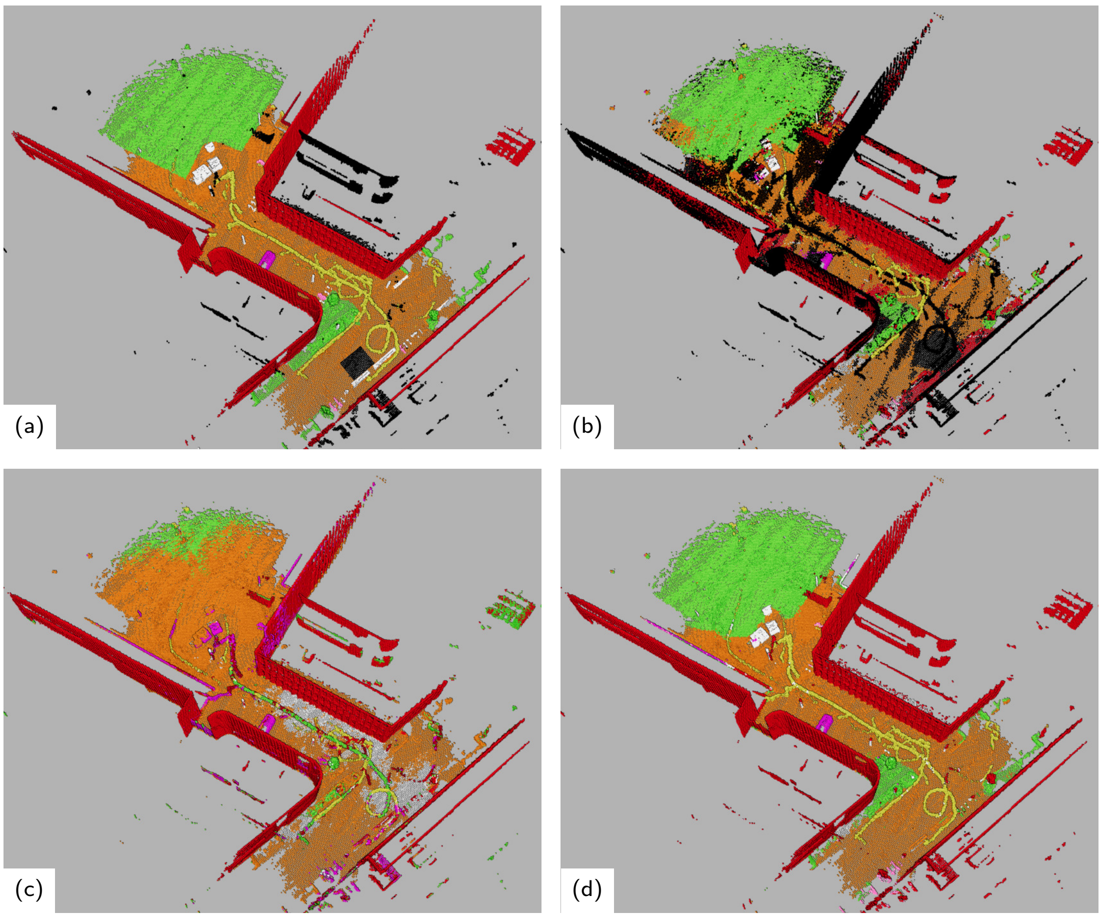

Figure 4.13: Semantic map of the urban campus outdoor flight. (a) manually annotated ground-truth. Map created from (b) scans labeled only by projected image semantics or from fused semantic clouds (c) before and (d) after retraining with label propagation. The camera-only map (b) misses annotations due to the camera's limited FoV but depicts most classes more accurately than (c). The semantic map after retraining (d) depicts all person tracks accurately and is very close to the ground-truth.

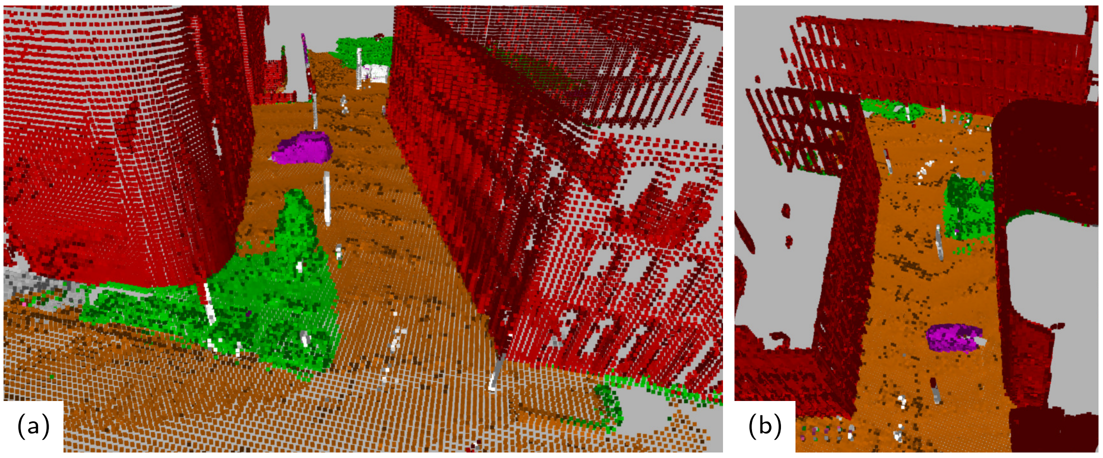

Figure 4.14: Closeups from Fig. 4.13 for detailed view of the semantic map of the urban campus outdoor flight after removing the tracks of dynamic persons.

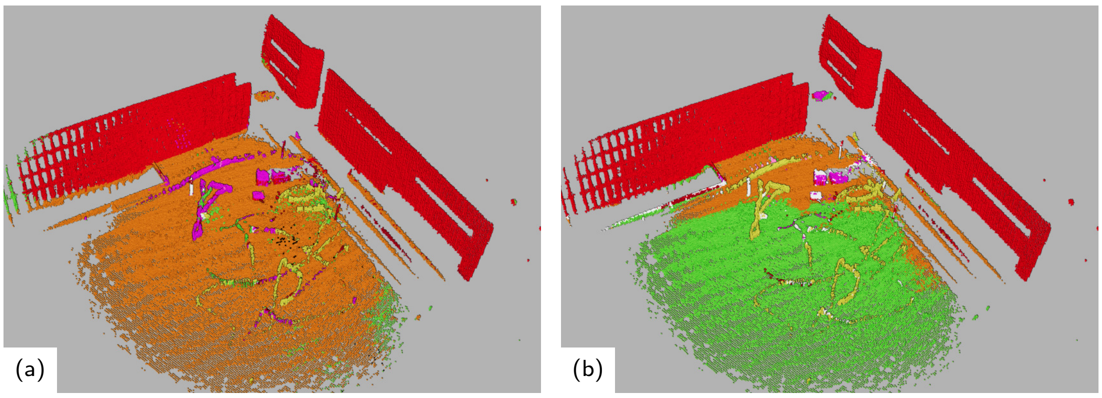

Figure 4.15: Aggregated maps from fused semantic clouds of another experiment (a) before and (b) after retraining with label propagation. Within the camera frustum (right part of the scene), person detection works sufficiently well, while they are misclassified elsewhere by the original LiDAR CNN. After retraining (b), persons tracks are segmented in the entire scene and the lawn is correctly labeled as vegetation.

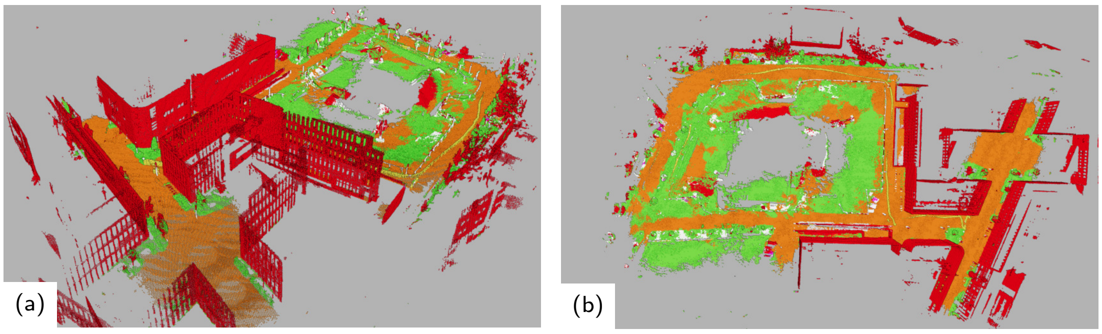

Figure 4.16: Aggregated semantic map for large area flight: (a) side-view, (b) top-view.

IoU Evaluation

To quantitatively evaluate the coherence of different point cloud segmentations, we calculate the intersection over union (IoU):

$$\mathrm{IoU}_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c + \mathrm{FN}_c}, \tag{4.8}$$

where TP, FP, and FN are the true positives, false positives, and false negatives, respectively. We compare for each segmented point its arg max class against the corresponding aggregated voxel label and average per class over the whole dataset. Tab. 4.3 shows the results for all classes that occur for a significant number of points in our recorded data. We use the manually annotated aggregated semantic map with a voxel size of 25 cm as ground-truth (cf. Fig. 4.13 (a)).

Applying the proposed adaptation of the normalization parameters improves the overall segmentation accuracy. The improvement is most significant for the building class, as the top half of the buildings are correctly labeled (cf. Fig. 4.11). The fused semantic cloud improves the segmentation coherence for all classes, especially for persons and vegetation. Persons and small objects, such as bicycles, are more reliably detected in the RGB and thermal modalities, improving the fused output inside the camera frustum. Using a higher input resolution for LiDAR segmentation gives significant improvements

Table 4.3: IoU evaluation against manually annotated semantic map: Average IoU per class (in %) and mean IoU for different point cloud segmentations, LiDAR CNN input resolutions, and FoVs.

Method	Res	FoV	Build.	Road	Veg.	Pers.	Bike	Car	Obj.	Mean
single w/o adapt*	512	LiDAR	32.7	72.9	2.2	2.7	3.7	0.6	3.4	16.9
single w/ adapt*	512	LiDAR	83.4	75.5	2.2	17.7	7.2	2.8	5.5	27.8
single w/ adapt	1024	LiDAR	82.7	71.8	4.0	43.3	6.3	4.1	7.4	31.4
fused	512	LiDAR	**84.3**	**77.0**	17.8	23.3	**8.4**	3.9	8.0	31.8
fused	1024	LiDAR	84.0	73.8	**18.7**	**47.2**	7.0	**5.3**	**9.5**	**35.1**
fused	512	camera	*94.0*	*62.2*	*48.6*	*36.6*	*17.1*	*34.0*	*32.1*	*46.4*
fused	1024	camera	***94.3***	*63.8*	*50.3*	*36.9*	***17.6***	*35.2*	*37.1*	*47.9*
img proj.	n/a	camera	*92.6*	***76.5***	***77.1***	***37.1***	*17.2*	***46.4***	***39.1***	***55.1***

*differences w.r.t. original results from Bultmann, Quenzel, and Behnke (2021) are due to a bug-fix in exporting the *pixel-shuffle* layer of the LiDAR CNN to the employed OpenVINO inference framework.

Table 4.4: IoU evaluation against manually annotated semantic map after retraining with label propagation: Average IoU per class (in %) and mean IoU for point cloud segmentation after retraining with label propagation.

Method	Res	Intensity	Build.	Road	Veg.	Pers.	Bike	Car	Obj.	Mean
single	512	✓	95.8	**88.7**	77.6	49.4	14.4	56.2	22.8	57.8
single	512	-	95.5	88.3	75.3	65.0	19.5	**66.9**	**31.6**	**63.2**
single	1024	-	95.4	88.5	74.1	67.7	**21.3**	55.2	22.9	60.7
fused	512	✓	**95.9**	**88.7**	**77.8**	49.5	14.5	55.2	22.7	57.8
fused	512	-	95.6	88.4	75.8	65.4	19.1	60.8	31.2	62.3
fused	1024	-	95.5	88.5	74.4	**67.8**	21.2	53.7	22.7	60.5

for the person class (43.3 % vs. 17.7 %) and a small improvement of mean IoU. The track of the operator, who always stayed behind the UAV and was not visible in the camera, is often misclassified, significantly impacting the mIoU of the person class as it constitutes a large number of the points annotated as person in the ground-truth map. The LiDAR segmentation with full input resolution segments larger parts of the operator track correctly. Results for the semantic cloud from projected image segmentation and the fused semantic cloud in the reduced FoV of the camera show significantly improved mIoU values also for vegetation, cars, and other foreground objects.

The semantic fusion without retraining the LiDAR segmentation CNN with label propagation successfully creates a coherently labeled 3D semantic interpretation of global structure in the full 360° LiDAR FoV and for both global structure and small dynamic objects in the camera FoV.

To improve the accuracy for the difficult semantic classes in the entire FoV, label propagation is used to retrain the LiDAR segmentation with pseudo-labels obtained

Table 4.5: Ablation on data used for label propagation: Mean IoU (in %) of point cloud segmentation using different datasets measured against the manually annotated map.

Datasets	Label Propagation	Intensity	Mean IoU
Sem.KITTI (pretrained*)	-	✓	27.8
Sem.KITTI + Paris	-	✓	32.7
Sem.KITTI + Paris	-	-	41.2
Sem.KITTI + Paris	fused map	✓	37.5
Sem.KITTI + Paris	fused map	-	38.5
Sem.KITTI + Paris	single scan	✓	49.4
Sem.KITTI + Paris	single scan	-	57.0
Sem.KITTI	camonly map	✓	57.6
Sem.KITTI	camonly map	-	60.4
Sem.KITTI + Paris	camonly map	✓	57.8
Sem.KITTI + Paris	camonly map	-	**63.2**

*pretrained by Cortinhal, Tzelepis, and Aksoy (2020).

from the camera-only map (cf. Sec. 4.3.5). The results are shown in Tab. 4.4. The segmentation accuracy for the whole 360° horizontal FoV almost doubles from 31.8 % to 62.3 % and also is significantly higher than the 55.1 % previously achieved in the camera FoV only. The retrained LiDAR segmentation CNN generalizes better without using the intensity input channel for the person and small object classes. Fusing the LiDAR segmentation with the image semantics gives a small gain for the person class but overall performs slightly worse. This underlines that the LiDAR CNN has learned the relevant segmentation skill from the RGB and thermal image modalities through cross-modal supervision during retraining. Small gains from an increased input resolution can still be observed for person and bicycle classes, but the mean IoU does not improve. Furthermore, the inference speed drops below the 10 Hz measurement frequency of the LiDAR at the 1024×128 input resolution (cf. Tab. 4.1). Therefore, we use the CNN at 512×64 input resolution in our online experiments.

We further perform an ablation study on the employed training data and pseudo-label sources in Tab. 4.5. The addition of data from Paris-CARLA-3D (Deschaud et al., 2021), covering the entire sensor FoV, improves the mean IoU by 5-13 %, with or without using intensity input, respectively. As observed in Tab. 4.4, networks not using the intensity channel generalize better to our dataset. The most significant gain is achieved adding data recorded by our sensor, using label propagation from the camera modality as cross-domain supervision. This also reduces the differences between using or not using intensity. Pseudo-labels obtained from the fused semantic map, including the noisy raw point cloud segmentation, are too imprecise to achieve significant improvements after retraining. Using pseudo-labels from the camera overlay of single scans improves the segmentation quality despite only a small part of each scan being labeled (cf. Fig. 4.8 (b)).

Table 4.6: Mean IoU (in %) of aggregated semantic maps measured against the manually annotated ground-truth.

Method	fused cloud	camera-only	camera-only (camera FoV)	fused cloud with label propagation
mIoU	29.5	41.8	60.3	68.0

The best results are achieved using pseudo-labels from the spatio-temporally aggregated camera-only semantic map, with an improvement of 35.4 % over the initial point cloud segmentation in our scenario. Using only own data for retraining, without Paris-CARLA-3D, performs slightly worse, as the training data is less diverse.

While in the previous evaluations, the IoU was calculated for individual semantic point clouds and averaged over the dataset, we show the IoU of the aggregated semantic maps in Tab. 4.6. As for the single scans, the semantic segmentation of the camera modalities is significantly more accurate than the fused cloud including the raw point cloud segmentation. For the *camera FoV* evaluation, we use only points inside the camera frustum and do not count unlabeled points as false negatives. With label propagation, the improvement upon the camera-only map, used as source for pseudo-labels during retraining, amounts to $\approx 8\,\%$ for the limited camera FoV and $\approx 26\,\%$ for the full $360°$ horizontal FoV. The accurate, automatically generated supervision from the image domain with narrow FoV generalizes to the full LiDAR FoV via label propagation.

Disaster Test Site

For qualitative evaluation, further flights were conducted on a disaster test site of the German Rescue Robotics Center (Kruijff-Korbayová et al., 2021). These experiments demonstrate the generalization capabilities of the proposed system to environments significantly different from the urban campus area where the data used for label propagation was recorded. The retrained LiDAR CNN is directly employed for these experiments, without any further adaptation.

Figure 4.17 shows the semantic perception of the disaster test site. Multiple persons, a fire truck, and even distant cars are reliably detected in the image and included into the fused image segmentation and point cloud. The point cloud segmentation also labels persons not visible in the camera correctly. Figure 4.17 (d) depicts the aggregated semantic map of the scenario. Some noise is visible in one of the person tracks and the higher trees are erroneously labeled as building structure, but the overall perception remains coherent also in this challenging scenario.

Figure 4.18 highlights the benefits of the thermal camera also at daylight: Persons are detected great distance, while only larger vehicles are detected in the RGB image. For the thermal camera, transfer from the training dataset with autonomous driving scenarios to the aerial perspectives of the UAV flights was possible without explicit domain transfer techniques, as the sensor characteristics in the dataset and of the employed thermal camera are similar.

To further improve the results for the disaster test site, another iteration of label propagation could be performed, using pseudo-labels automatically obtained from the aggregated semantic map of the environments.

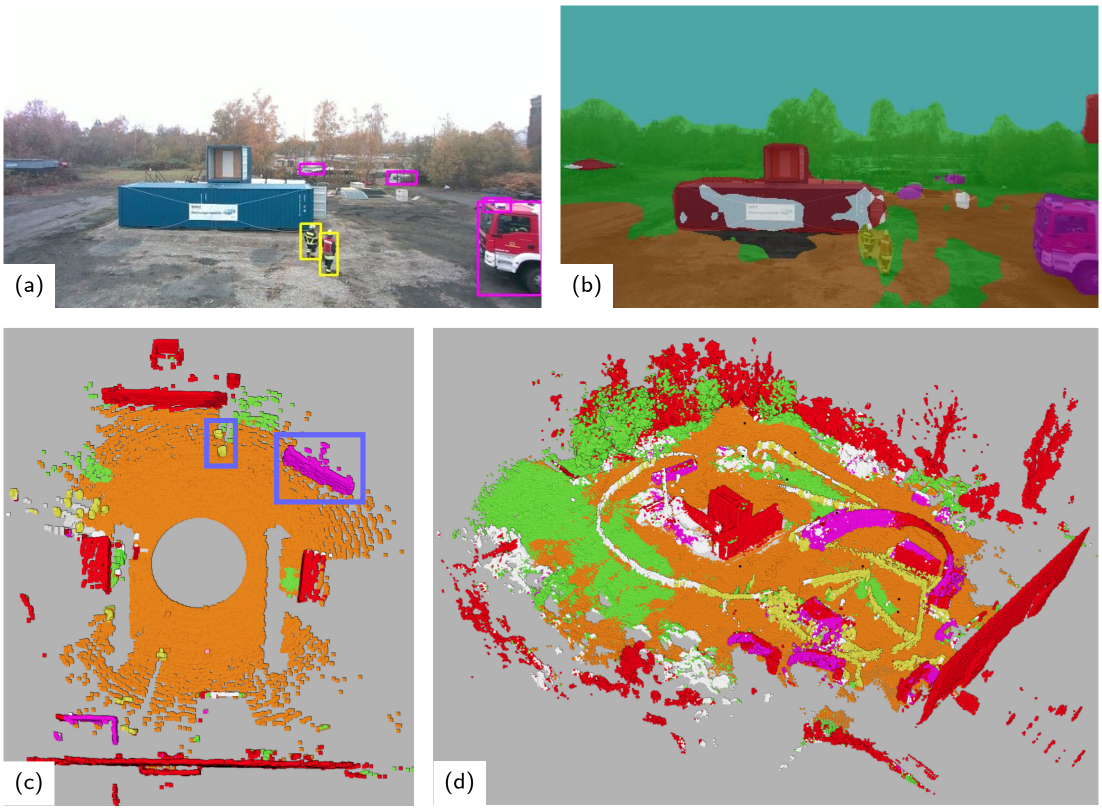

Figure 4.17: Semantic perception on disaster test site: (a) RGB detections, (b) fused RGB segmentation, (c) fused semantic cloud, (d) aggregated semantic cloud. Persons and fire truck inside the camera FoV are highlighted with blue rectangles in (c).

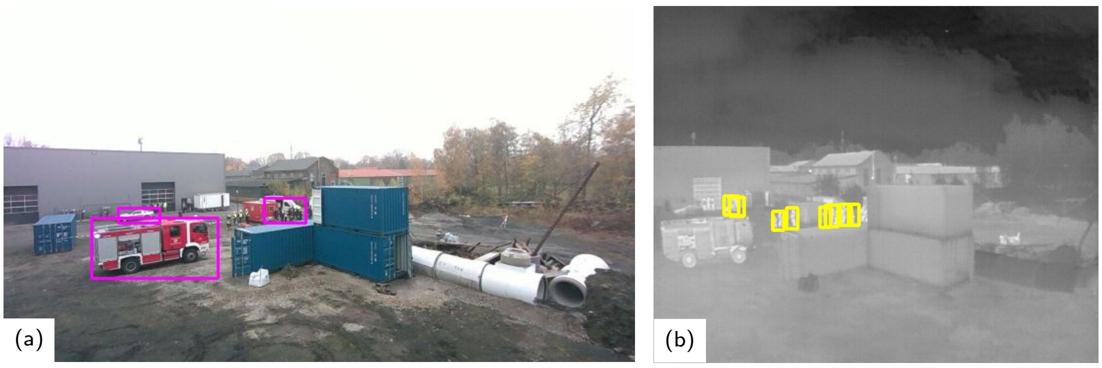

Figure 4.18: Comparison of (a) RGB and (b) thermal detections on disaster test site. Persons are detected in the thermal image also at very high distance, while only the larger vehicles are detected in the RGB image.

4.5 Discussion

In this chapter, we presented a UAV system for semantic image and point cloud analysis as well as multi-modal semantic fusion. The inference of the lightweight CNN models runs onboard the UAV computer, employing an inference accelerator and the integrated GPU of the main processor for computation. The Edge TPU performs inference in 8-bit quantized mode and showed more efficient CPU usage. The iGPU is more flexible, e. g.

to directly run pre-trained models, as it uses 16- or 32-bit floating-point precision and does not require model quantization.

The proposed framework for semantic scene analysis provides a 2D image segmentation overlay and a 3D semantically labeled point cloud which is further aggregated into an allocentric semantic map. The initial point cloud segmentation suffered from domain adaptation issues since available large-scale training datasets stem from autonomous driving scenarios with different viewpoints and sensors more directed towards the ground compared to the the UAV's LiDAR sensor. We addressed this issue by retraining the LiDAR segmentation CNN with data captured on our UAV using pseudo-labels automatically obtained from the aggregated semantic map. The pseudo-labels thereby stem from the RGB and thermal camera modalities, providing cross-domain supervision for the 3D point cloud. This takes up the idea of semantic feedback introduced in previous chapters, reprojecting parts of the fused allocentric semantic model into individual sensor views to provide a reliable source of semantic information.

With label propagation, the 3D segmentation accuracy of the proposed system significantly improves for the full LiDAR FoV. We evaluated the system in real-world experiments in an urban environment and at a disaster test site, showing coherent semantic perception of diverse and challenging scenes.

Future work includes retraining also the image CNNs with label propagation to improve task-specific performance, and adding RGB color channels to the point cloud segmentation input, e. g. using a 360° fisheye camera together with the LiDAR sensor.

The UAV system with onboard real-time perception can be considered as a mobile smart edge sensor compared to the stationary sensor boards introduced in the previous Chapters 2 and 3. Moving sensors provide changing viewpoints and can cover significantly larger areas than stationary sensors with fixed perspectives, thus extending the coverage of the scene model. However, it would be difficult to integrate the outdoor UAV platform into the lab-scale indoor environment covered by the previously proposed smart edge sensor network. To this end, we will instead use a mobile service robot that is tracked by the sensor network and integrate observations from its changing viewpoints into the allocentric scene model, as introduced in the following chapter.

5 External Camera-based Mobile Robot Pose Estimation

Preface

This chapter is adapted from Bultmann, Memmesheimer, and Behnke (2023), previously published by IEEE and presented at the International Conference on Robotics and Automation (ICRA 2023).

Statement of Personal Contribution

The author of this thesis substantially contributed to all aspects of the publication (Bultmann, Memmesheimer, and Behnke, 2023), including the literature survey, the conception, formalization, design, and implementation of the proposed methods, the preparation and conduct of experiments for the evaluation of the proposed approach, the analysis and interpretation of the experimental results, the preparation of the manuscript, as well as the revision and final editing of the version to be published.

The content presented in this chapter, unless otherwise stated, is the contribution of the author of this thesis.

Unpublished Content

In addition to the above publication, this chapter contains further unpublished content. In particular, Section 5.4.5, which describes an experiment on human-aware anticipatory navigation, is additionally presented.

Abstract

We present an approach for estimating a mobile robot's pose w.r.t. the allocentric coordinates of a network of static cameras using multi-view RGB images. The images are processed online, locally on smart edge sensors by deep neural networks to detect the robot and estimate 2D keypoints defined at distinctive positions of the 3D robot model. Robot keypoint detections are synchronized and fused on a central backend, where the robot's pose is estimated via multi-view minimization of reprojection errors. Through the pose estimation from external cameras, the robot's localization can be initialized in an allocentric map from a completely unknown state (*kidnapped robot problem*) and robustly tracked over time. We conduct a series of experiments evaluating the accuracy and robustness of the camera-based pose estimation compared to the robot's internal navigation stack, showing that our camera-based method achieves pose errors below 3 cm and 1° and does not drift over time, as the robot is localized allocentrically. With the robot's pose precisely estimated, its observations can be fused into the allocentric scene model. We show a real-world application, where observations from mobile robot

and static smart edge sensors are fused to collaboratively build a 3D semantic map of a $\sim$240 m^2 indoor environment. Furthermore, through semantic feedback of human pose observations from the external sensors, the robot can anticipate persons appearing from behind occlusions and anticipatorily adapt its navigation path to keep a safe distance.

5.1 Introduction

Semantic scene understanding is an important requirement for intelligent robot action, such as collision-free navigation, object manipulation, or human-robot interaction. Scene interpretation from a single sensor view, however, has a limited field of perception, measurement range, and resolution, and suffers from occlusions. Collaborative perception between mobile robots and distributed static smart edge sensors alleviates these issues and enables to build 3D semantic models of large scenes without being limited by the measurement range or occlusions of a single sensor.

A key prerequisite for fusing observations from different perspectives is knowing the relative sensor poses. While the extrinsic calibration of static sensors can be performed beforehand, a mobile sensor's pose w.r.t. the sensor network must be initialized and tracked to fuse the robot observations into the allocentric model in a consistent manner.

In this chapter, we propose to estimate a mobile robot's pose w.r.t. the allocentric coordinates of a network of static smart cameras using multi-view RGB images. For this, we build upon previous work on 3D semantic scene perception using distributed smart edge sensors (cf. Chapter 3), where images are processed locally on smart edge sensor boards by deep neural networks for semantic image interpretation. For robot pose estimation, we process the images with convolutional neural networks (CNNs) for robot detection and estimation of 2D projections of keypoints defined at distinctive positions of the 3D robot model. Unlike classical robot-to-camera pose estimation systems, our method does not require attaching fiducial markers to the robot. Furthermore, the CNN for keypoint estimation is trained only on synthetic data obtained through randomized scene generation (Schwarz and Behnke, 2020). Robot keypoint detections are streamed to a central backend where they are synchronized and the robot's pose is estimated via multi-view minimization of reprojection errors. Through the pose estimation from external camera views, the robot's localization is initialized in the global map from a completely unknown state, and robustly tracked over time. Furthermore, using multiple sources for localization, i.e. the external camera views together with the robot's internal 2D LiDAR-based navigation, increases robustness in highly cluttered, dynamic real-world environments, where few distinct features, such as walls or columns, are visible in the LiDAR due to occlusions.

With the robot's pose precisely estimated, it is integrated as a mobile sensor node into the camera network. We deploy the CNNs for semantic image interpretation from Chapter 3 onto the robot's inference accelerator and use its RGB-D camera to obtain semantically annotated point clouds of the robot's view. The robot's observations are then fused into the allocentric scene model to build a 3D semantic map of a large room in collaboration with the static smart edge sensors.

Fig. 5.1 shows the employed Toyota human support robot (HSR) and static smart edge sensors together with a 3D view of the robot's semantic observations to be fused into the 3D scene model, initialized from a prior without any semantic annotations. As the robot pose is initialized globally coherently through the external keypoint-based

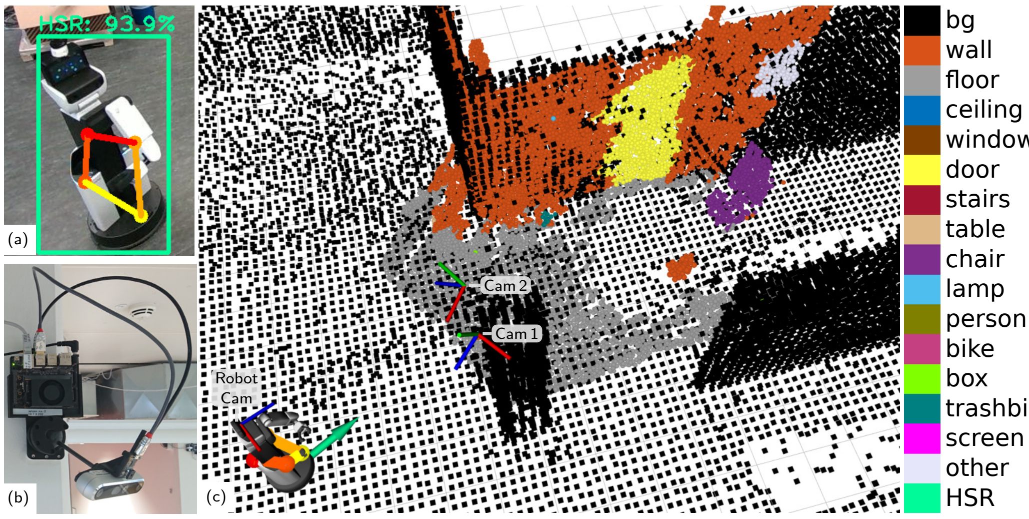

Figure 5.1: Robot pose estimation for collaborative perception: (a) Toyota Human Support Robot (HSR) with 2D bounding box and keypoints used for pose estimation detected by static smart edge sensors (b). (c) 3D scene view with robot model and 3D keypoints at the estimated robot pose (green arrow) and robot's semantic observations (colored point cloud) to be fused into the allocentric scene model (black squares). Robot observations fit the allocentric model, showing that the robot's pose is initialized globally coherently through localization by the smart edge sensors.

pose estimation, its observations consistently fit the allocentric model.
In summary, the main contributions of this chapter are:

- A novel method for marker-less mobile robot pose estimation using multi-view keypoint detections to initialize the robot's localization in a global map (*kidnapped robot problem*) and robustly track it over time,
- integration of a mobile robot into a network of static smart edge sensors, fusing the robot's observations from changing viewpoints into the global scene model,
- quantitative evaluation of the pose estimation accuracy and robustness, and
- demonstration of collaborative perception in real-world scenes between mobile robot and sensor network to build a globally consistent 3D semantic map.

5.2 Related Work

Visual robot detection and localization have been research topics of high interest for a variety of autonomous systems like drones (Ashraf, Sultani, and Shah, 2021), mobile robot platforms (Meger et al., 2009; Pizarro et al., 2008, 2010), humanoids (Amini, Farazi, and Behnke, 2022), autonomous cars (Gawel et al., 2018), and underwater robots (Joshi et al., 2020; Shkurti et al., 2017). External localization of moving objects is also employed for traffic monitoring (Severi et al., 2018; Strand, Honer, and Knoll, 2022) or action recognition (W. Wang et al., 2016).

Detection methods localize robots in images and allow for tracking applications like underwater convoying (Shkurti et al., 2017) or monitoring (Ashraf, Sultani, and Shah, 2021). In contrast, we aim to estimate a robot's pose from multiple views to initialize its localization in a map and to compensate for localization errors in highly cluttered, dynamic environments.

Gawel et al. (2018) presented a semantic graph-based multi-view approach for robot localization in autonomous driving scenarios. They use semantic estimates from various views like front and aerial views and integrate them in separate target and query graphs. Lu, Richter, and Yip (2022) presented an approach for marker-less robot pose estimation via keypoint detection (Mathis et al., 2018). They aim to define keypoints on the robot maximizing the localization performance in 2D and 3D. Their approach utilizes synthetic training data and transfers well to real sensor data. They estimate the pose of stationary robots from single views, whereas our approach globally tracks a mobile manipulation platform. Lee et al. (2020) consider the inverse problem of localizing the camera in relation to an articulated robot. They estimate 2D keypoints of robot joints and recover the camera extrinsics w.r.t. the robot manipulator given that the camera is intrinsically calibrated and the robot joint positions are available.

Pizarro et al. (2008, 2010) presented approaches for robot localization from a single view. Shim and Cho (2015) proposed an approach for multi-view mobile robot localization. They first project the camera images onto a common plane and then localize the robot using its contours after removing shadows. Similarly, Chakravarty and Jarvis (2009) utilize a surveillance camera system to localize the robot in the image plane by background subtraction of a color thresholded image. Similar to our work, the surveillance camera view is extended by the robot's view. Obviously, such simple approaches would fail in highly dynamic and cluttered environments.

The field-of-view of autonomous mobile robots can be extended in comparison to single-view systems using smart edge sensors for global collaborative perception. In prior work, we utilized smart edge sensors to estimate 3D human poses from multiple views (cf. Chapter 2) and integrate 3D semantic scene perception (cf. Chapter 3). In this chapter, we focus on the integration of a mobile robot as an additional smart edge sensor node, laying the foundations for many potential applications like external, camera-based control and task planning as well as augmentation of the internal robot view by the integration of human poses or semantic segmentation estimates that are robust against occlusions.

Rekleitis et al. (2001) presented an approach for collaborative exploration of visual maps using two mobile robots of different capability levels, where one robot was actively collecting visual data for mapping and the second, passive robot visually refined the pose of the moving robot. S. Dong et al. (2019) proposed a multi-robot collaborative approach for dense reconstruction. They associate task views of uncertain or unexplored map areas to the robots based on the traveling salesman problem. In contrast to our approach, they concentrate on an actively moving set of mobile robot platforms and omit semantics. Similarly, Yue et al. (2020), propose to combine an unmanned aerial vehicle (UAV) and an unmanned ground vehicle (UGV) for collaborative semantic mapping. Ahmed et al. (2021) propose to integrate top-view surveillance cameras for collaborative robotics but assume that the estimations from the surveillance cameras are provided to the robot without actively fusing information of the robot's sensors.

With PoseCNN, Xiang et al. (2018) propose a 6D pose estimation approach jointly estimating semantic labels on a pixel level. The translation is estimated by regressing the object center in camera coordinates and the rotation is separately regressed using a newly introduced ShapeMatch-Loss. Chen Wang et al. (2019) present a dense fusion method for 6D pose estimation. They first estimate semantic segmentation and bounding boxes, and then densely fuse RGB and depth images in an embedding. Finally, an iterative pose integrator refines the 6D pose. Keypoint-based approaches for 6D pose estimation, as introduced in Chapter 3, adopt a two-stage pipeline, first estimating keypoints for each object instance and, second, retrieving the object poses via PnP (Lepetit, Moreno-Noguer, and Fua, 2008). They have been widely used in recent years (Amini, Periyasamy, and Behnke, 2022; Lee et al., 2020; Peng et al., 2019; Rad and Lepetit, 2017). Although most pose estimation approaches are object-centric, here, we propose to use a keypoint-based estimator for the visual pose estimation of a mobile robot.

5.3 Method

We consider scenarios where N externally mounted cameras $C_i, i = 1, \dots, N$ observe a mobile robot from different viewpoints. The intrinsic and extrinsic calibration of the external cameras are performed beforehand (cf. Sec. 2.5), i.e., we assume the transformation ${}^{C_i}_{W}\boldsymbol{T} \in \mathbb{R}^{4\times4}$ from world to camera coordinates and the projection $\Pi_i(\cdot)$ to the image plane of C_i to be known. The cameras observe 2D projections $\boldsymbol{k}_{ij} \in \mathbb{R}^2$ of M keypoints $\boldsymbol{p}_j \in \mathbb{R}^3, j = 1, \dots, M$ defined on distinct positions of the 3D robot model. From these observations, we aim to estimate the robot's pose in world coordinates ${}^{W}_{R}\boldsymbol{T} \in \mathbb{R}^{4\times4}$. As the employed mobile robot moves on the ground plane, we consider three degrees of freedoms (DoFs): $\boldsymbol{x} = [x, y]^\top \in \mathbb{R}^2$, and $\theta \in (-\pi, \pi]$, with

$$
{}^{W}_{R}\boldsymbol{T}(\boldsymbol{x}, \theta) = \begin{bmatrix} \boldsymbol{R}(\theta) & \boldsymbol{0} & \boldsymbol{x} \\ \boldsymbol{0}^\top & 1 & 0 \\ \boldsymbol{0}^\top & 0 & 1 \end{bmatrix}, \boldsymbol{R}(\theta) \in \mathcal{SO}(2) \ . \tag{5.1}
$$

Extending this formulation to more DoFs is straightforward. We estimate the robot pose in a two-stage process, similar to Lee et al. (2020) but exploiting multi-view observations: First, robot keypoints are detected on the images, locally on the sensor boards (Sec. 5.3.1), and, second, the robot pose is estimated from a synchronized set of observations via multi-view minimization of reprojection errors (Sec. 5.3.2).

5.3.1 *Robot Keypoint Detection*

We aim to estimate the robot's base pose, independent of the articulation of the robot head or arm, and therefore define the keypoints used for pose estimation on distinct points of the rigid robot base (see Fig. 5.2 (c)). Keypoints defined manually at distinct points of the 3D model were shown to perform better than automatically defined ones in Sec. 3.4.1.2.

5.3.1.1 *Network Architecture*

We employ a top-down approach for robot keypoint detection, i.e., first detecting a bounding box of the robot in the full image (848×480 px) and then estimating keypoints on the crop of the robot. Top-down methods achieve better scale invariance than estimating keypoints on the full input image, as the crop of a detected robot is interpolated to a fixed resolution for keypoint estimation (here 192×256 px).

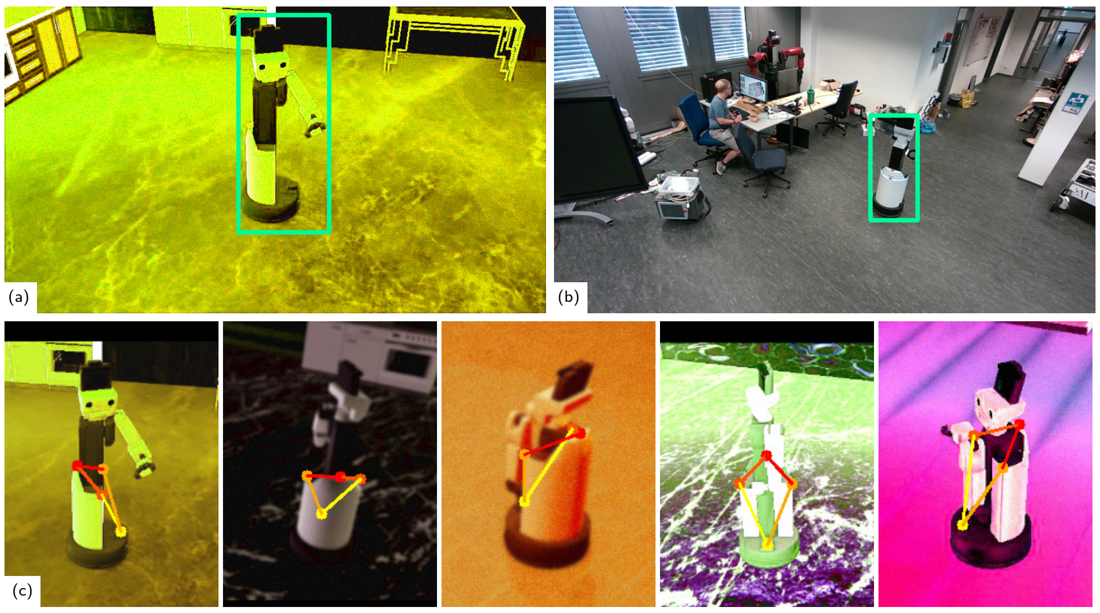

Figure 5.2: Examples of training images: (a) synthetic and (b) real images used for the detector; (c) synthetic images used for the keypoint estimation.

Camera images are processed locally on the smart edge sensors, using an Nvidia Jetson Xavier NX embedded inference accelerator (cf. Fig. 5.1 (b)). We employ the CNN architectures used in prior work for efficient person keypoint estimation on the embedded hardware (cf. Chapters 2 and 3): The recent MobileDet architecture (Xiong et al., 2021) is used for robot detection and the network of Xiao, Wu, and Wei (2018) with a MobileNet V3 backbone (Howard et al., 2019) for keypoint estimation.

5.3.1.2 *Training Data*

To reduce the labeling effort to a minimum, we train the networks predominantly on synthetic data. The CNN for keypoint estimation is trained purely on simulated data (36k samples), while we combine synthetic data and manually annotated real images (12k resp. 3.5k samples) for robot detection. The combination of real and synthetic data helps to boost detector performance in highly cluttered real-world environments and bounding-box labels are less costly to obtain than keypoint annotations. The keypoint estimation CNN generalizes well from only synthetic data.

We employ an extension of the *stilleben*-framework (Boltres et al., 2022; Schwarz and Behnke, 2020) for photorealistic, randomized scene rendering to generate multiple scenes of our robot moving through varying indoor environments, using the 3D robot model. Randomizing scene parameters in addition to image augmentation helps to bridge the reality gap for sim-to-real transfer. Fig. 5.2 shows examples of the employed training data. Note, that the network also learns to detect occluded keypoints from the surrounding image context.

5.3.2 *Robot Pose Estimation*

2D robot keypoints are sent over a network to a central backend, where detections from multiple cameras are software-synchronized via their timestamps and associated to corresponding framesets. The robot pose ${}^{W}_{R}\boldsymbol{T}$ (5.1) is recovered by solving a weighted nonlinear least squares problem via minimization of multi-view reprojection errors:

$$ {}^{W}_{R}\boldsymbol{T} = \underset{{}^{W}_{R}\boldsymbol{T}}{\arg\min} \sum_{i=1}^{N} \sum_{j=1}^{M} w_{ij} \left\| \boldsymbol{k}_{ij} - \Pi_i \left({}^{C_i}_{W}\boldsymbol{T}\, {}^{W}_{R}\boldsymbol{T} \boldsymbol{p}_j \right) \right\|^2, \tag{5.2} $$

with the weights w_{ij} depending on the confidence of the keypoint detection in the respective camera, similar to the human pose triangulation introduced in Sec. 2.4.2. We use the Levenberg-Marquardt algorithm as implemented in the Ceres library (Agarwal, Mierle, and Team, 2022) for optimization. We discern two different ways to initialize the optimization:

(i) When the robot pose has already been initialized in the global map and a current pose estimate is available from its internal navigation stack, we use this to initialize the optimization.

(ii) When no prior estimate is available, e.g., to initialize the robot's pose or when the communication link to the robot is unavailable, we use the PnP algorithm (Lepetit, Moreno-Noguer, and Fua, 2008) to obtain pose candidates from each individual camera view with $M \geq 4$ detected keypoints. The candidate poses are transformed to world coordinates using ${}^{C_i}_{W}\boldsymbol{T}^{-1}$ and projected to the ground plane. The initialization for Eq. (5.2) is then obtained via interpolation between the candidates, using spherical linear interpolation for the orientation.

In our experiments (Sec. 5.4), we observe that the pose estimation is well constrained when the robot is detected in $N \geq 2$ cameras, but less stable when visible in only a single camera and far away from the camera. This is due to the small width of 35 cm of the robot body, providing only a narrow baseline for orientation and depth estimation. To alleviate this issue, we implement a simple yet effective bearing-only heuristic for outlier detection: For pose estimates from a single camera view, we prevent unrealistically high changes in orientation or distance to the camera above a threshold d_θ resp. d_{depth} by updating only the translation orthogonal to the camera's viewing direction.

Lastly, we employ a pose graph (Dellaert and Kaess, 2017) to fuse the absolute pose estimations from external cameras, which occur sparsely at waypoints where the robot is static (cf. Sec. 5.3.3), with the continuous robot-internal odometry. The pose estimates from the robot's internal navigation stack are inserted into the pose graph as binary odometry constraints, while the pose estimates from external cameras are used as absolute, unary constraints. The covariance of the camera pose estimates is proportional to the reprojection error residual (5.2) and inversely to the number of cameras used in the optimization, giving the highest confidence to consistent estimates with low reprojection error in a large number of cameras. The pose graph is optimized each time a new external camera pose estimate occurs.

5.3.3 *Collaborative Localization and Perception*

Fig. 5.3 gives an overview of the proposed sensor network architecture for collaborative localization and perception. The mobile HSR robot is integrated into the network of static smart edge sensors for 3D semantic scene perception from Chapter 3. The robot pose is initialized in the allocentric scene from the external cameras. During operation, external robot pose estimates are sent as pose-correction feedback to the robot sparsely at waypoints where the robot is static, updating the robot's internal particle filter-based

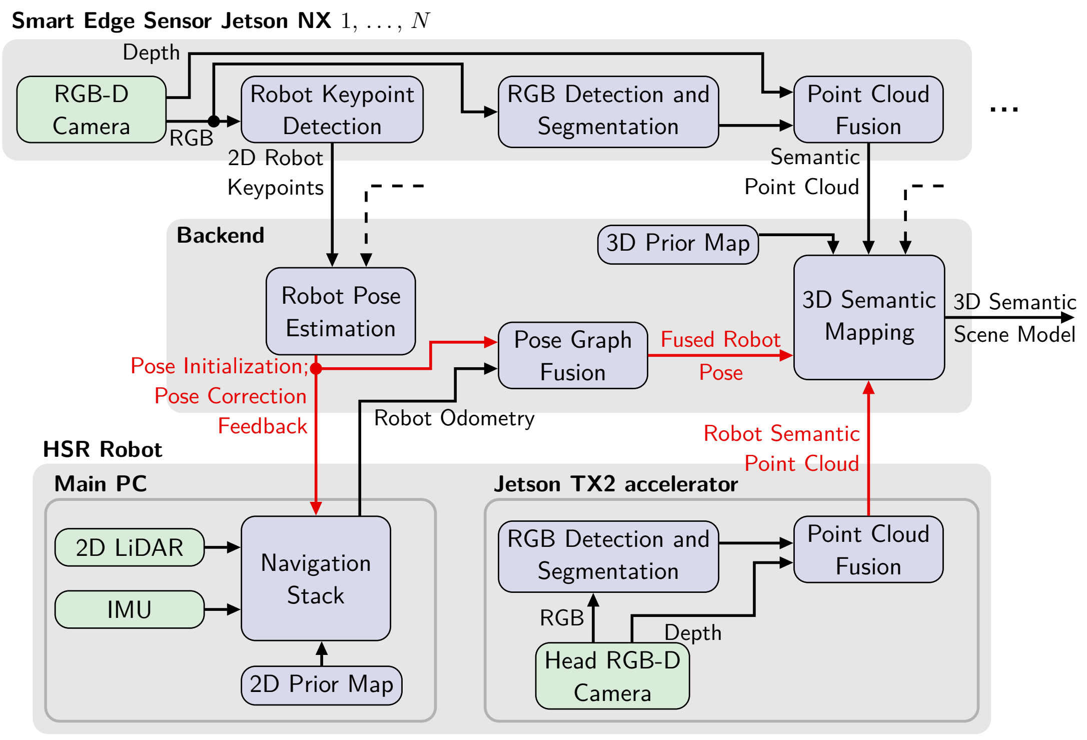

Figure 5.3: Overview of the proposed sensor network architecture for collaborative localization and perception. N external smart edge sensors observe mobile HSR robot and scene from static viewpoints. Robot pose is initialized and corrected via external camera pose estimation. Robot observations from changing viewpoints are fused into the allocentric scene model.

localization (Thrun et al., 2001) to keep it globally coherent within the scene model. The interface provided by the robot's navigation stack permits to update the localization from external measurements only when not in movement. To simplify system integration and for better interoperability, we use the robot's navigation stack as provided by the manufacturer and consider it as a closed module. We do not aim to fully control the robot localization via the external cameras for the robot to keep its local autonomy: Its navigation stack integrates information from the camera network when available, but does not depend on it. When, e.g., the communication link is lost, the robot can still navigate autonomously using its integrated sensors.

For collaborative perception, the mobile robot provides changing viewpoints of areas outside the field-of-view of the static smart edge sensors, and its semantic percepts are fused into the allocentric scene model.

5.3.3.1 *Semantic Perception onboard the Robot*

The HSR's computing system comprises a main PC used for communication, robot navigation, and control and an embedded CNN inference accelerator (Nvidia Jetson TX2). We employ the latter to deploy CNNs for object detection and semantic segmentation as introduced in Sec. 3.3.2 and obtain semantic point clouds using the Asus Xtion RGB-D camera mounted on the robot's head. The robot's inference accelerator hardware is a previous generation of the compute boards of the static smart edge sensors, with lower, but sufficient resources for semantic point cloud estimation at 1 Hz.

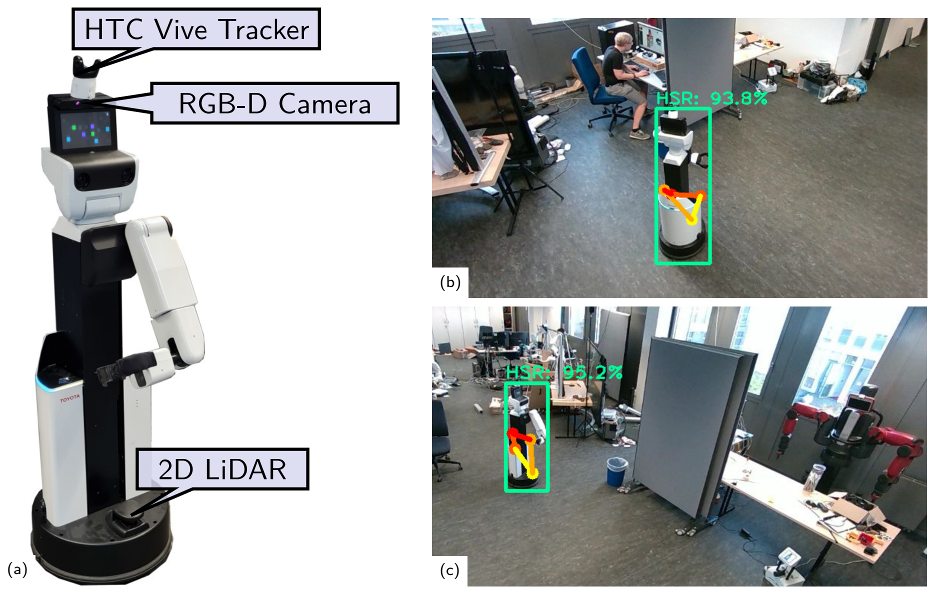

Figure 5.4: Robot setup (a) and exemplary robot keypoint detections (b, c).

5.3.3.2 *Semantic Map Fusion*

The robot's semantic observations from changing viewpoints are integrated into the allocentric 3D semantic map using the Bayesian probabilistic fusion proposed in Sec. 3.3.3. Because we consider the mobile robot's pose less reliable than the a-priori calibrated static sensor poses, we perform an ICP alignment step before integrating the robot's data into the map. This compensates for the drift accumulated between pose corrections and tolerances in the robot's forward kinematics. As a good initialization is critical for ICP, precise robot localization is necessary for initializing the alignment.

5.4 Evaluation

5.4.1 *Experiment Setup*

During the experiments, the HSR robot operates in a challenging, cluttered indoor environment of ∼240 m^2 size. Four external smart edge sensors are mounted at ∼2.5 m height in the center of the room to initialize and correct the robot localization. As a reference for pose estimation, we employ the affordable and easy-to-use HTC Vive Pro tracking system, which was shown to yield position accuracies within a few millimeters (Bauer, Lienhart, and Jost, 2021). For this, we place an HTC tracker on the robot's head. Fig. 5.4 shows the robot setup and exemplary keypoint detections from two external camera views.

To evaluate the pose estimation accuracy, we define seven waypoints in the observed area, visible from different numbers of cameras, as shown in Fig. 5.5, and connect them to three different trajectories (cf. Fig. 5.6). As the tracking system and the camera network don't operate in the same reference frame, trajectories are rigidly aligned via Procrustes analysis (Sturm et al., 2012) before evaluation.

A video of the experiments described in the following is available on our website[1].

1 https://www.ais.uni-bonn.de/videos/ICRA_2023_Bultmann/

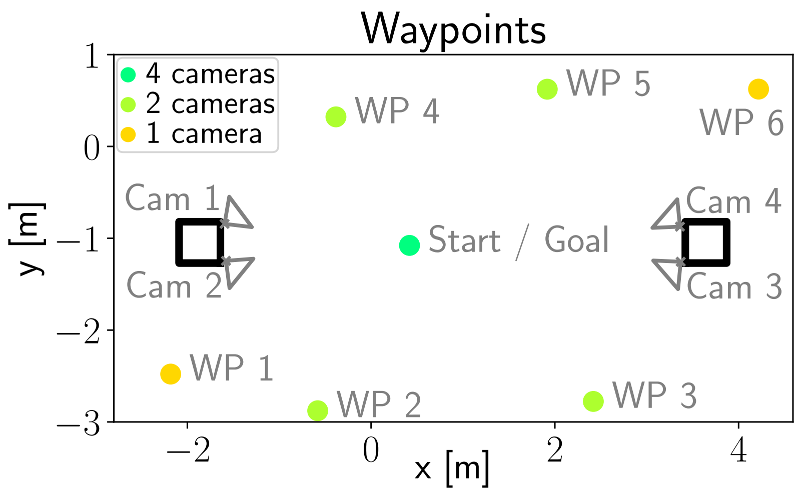

Figure 5.5: Evaluation of robot pose estimation: Room-setup with camera positions and waypoints colored by number of cameras from which they are visible.

5.4.2 *Pose Estimation Accuracy*

During a first set of experiments, we record a dataset of three iterations of each trajectory in both forward and reverse direction, resulting in 18 sample trajectories. At the beginning of each trajectory, the robot's pose is initialized from observations of all four cameras and we verify that robot observations are consistent with the global model (cf. Fig. 5.1 (c)) to confirm a good initialization. The 2D grid map used by the robot for LiDAR navigation is initialized with the prior model of the empty room. The pose correction from external cameras is not sent to the robot during these experiments to measure the deviation over the full trajectory.

Fig. 5.6 shows an exemplary iteration of each trajectory, comparing the path estimated by the robot's internal navigation stack and the fused trajectory estimate, obtained using pose corrections from the external cameras and pose graph optimization, with the ground-truth obtained from the HTC reference tracking system. Furthermore, the evolution of translation error is shown over the traveled distance. While the robot's internal localization quickly accumulates errors of 20-30 cm, the error of the external camera pose estimation stays below 5 cm when the robot is observed from at least two cameras. The error can be higher when observing the robot in only a single camera, e.g. WP 1 (1st WP of Traj. 1, resp. 5th WP of Traj. 2), but always improves upon the robot's internal estimate. The single-camera observations can further be improved by fusing with robot odometry via the pose graph. In Fig. 5.6 (Traj. 1), we illustrate the outlier detection for pose estimation from a single camera: The raw pose estimate shows an unrealistically high change in distance to the observing camera; therefore, the pose correction is restricted to the lateral direction.

Tab. 5.1 and 5.2 report a quantitative evaluation of the translation and orientation errors at the waypoints, ordered by the number of cameras from which the robot is observed. The pose error is lowest when the robot is observed from two or four cameras and amounts to 2.93 cm and 0.88° averaged over all waypoints, significantly improving

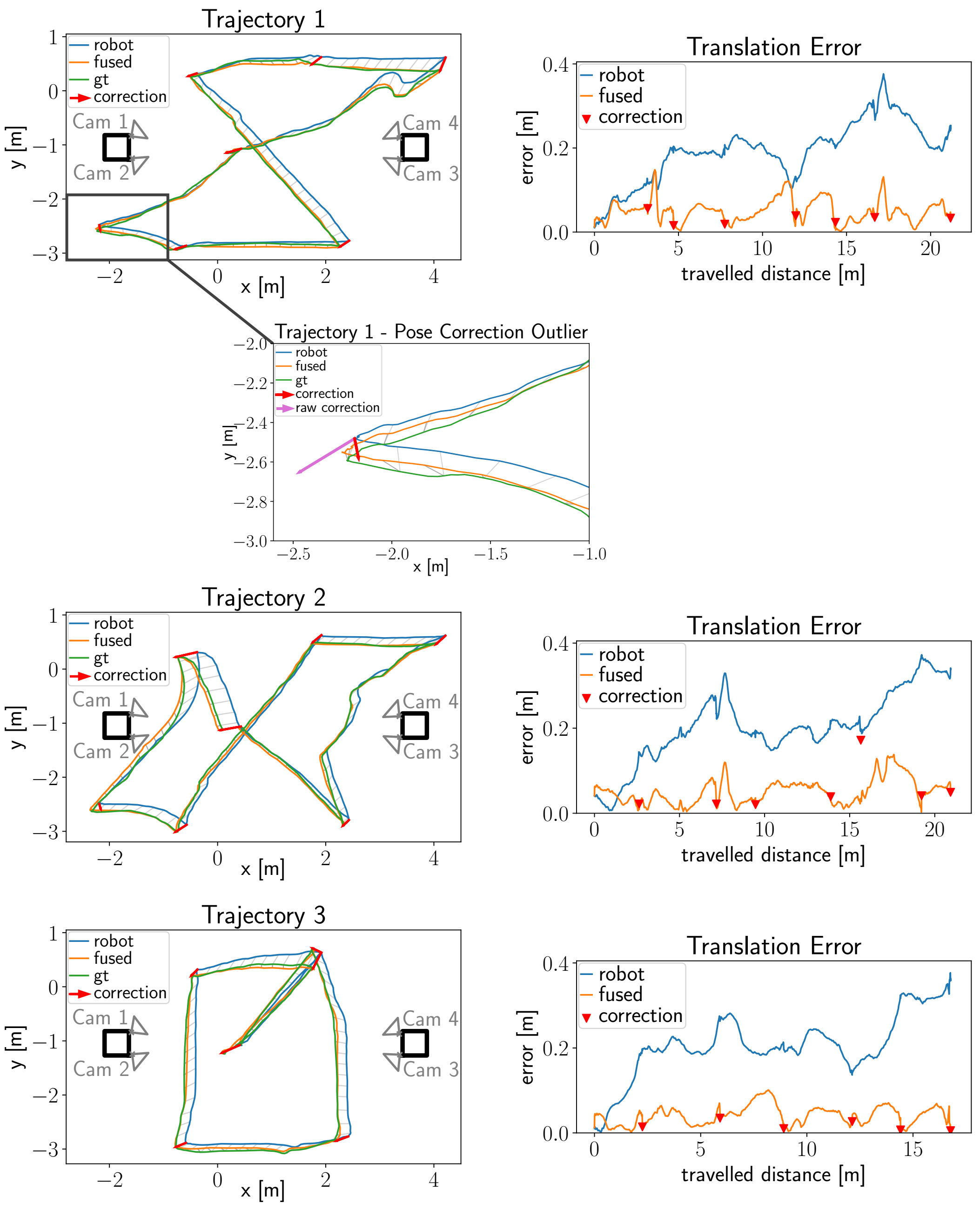

Figure 5.6: Evaluation of robot pose estimation: Plots of one resp. iteration of the three trajectories used for evaluation of the translation error over the traveled distance; Traj. 1: example for outlier detection at WP 1 observed from only a single camera. The pose correction from the external cameras is not sent as feedback to the robot during these experiments to measure the deviation over the full trajectory length. See Sec. 5.4.2 for further explanations.

over the robot internal localization with an average error of 19.1 cm and 1.11°. The outlier detection for single-camera pose estimation significantly improves the accuracy in the resp. waypoints by 5.6 cm resp. 2.0° w.r.t. the raw estimate and prevents a worsening of the average orientation estimate w.r.t. the robot's internal localization. Averaging

Table 5.1: Translation error (mean ± std) at waypoints, by number of cameras with robot detections and pose estimation source.

Pose Estimation	1 Camera	2 Cameras	4 Cameras	**Average**
Robot	20.6 ± 7.6 cm	17.1 ± 6.9 cm	25.0 ± 6.9 cm	19.1 ± 7.6 cm
Cameras (raw)	13.8 ± 10.1 cm	2.59 ± 1.49 cm	2.88 ± 1.42 cm	4.65 ± 6.22 cm
Cameras (1 frame)	8.23 ± 5.23 cm	2.59 ± 1.49 cm	2.88 ± 1.42 cm	3.65 ± 3.36 cm
Cameras (5 frames)	7.77 ± 5.47 cm	**2.58 ± 1.48 cm**	2.82 ± 1.43 cm	3.54 ± 3.31 cm
Fused	**4.25 ± 1.57 cm**	2.64 ± 1.48 cm	**2.73 ± 1.41 cm**	**2.93 ± 1.60 cm**

Table 5.2: Orientation error (mean ± std) at waypoints, by number of cameras with robot detections and pose estimation source.

Pose Estimation	1 Camera	2 Cameras	4 Cameras	**Average**
Robot	$1.17 \pm 1.19°$	$1.03 \pm 1.23°$	$1.30 \pm 1.96°$	$1.11 \pm 1.38°$
Cameras (raw)	$3.39 \pm 2.87°$	$0.97 \pm 0.79°$	$1.02 \pm 0.76°$	$1.44 \pm 1.72°$
Cameras (1 frame)	$1.40 \pm 1.31°$	$0.97 \pm 0.79°$	$1.02 \pm 0.76°$	$1.06 \pm 0.92°$
Cameras (5 frames)	$1.18 \pm 1.27°$	$0.86 \pm 0.68°$	**$0.97 \pm 0.74°$**	$0.94 \pm 0.84°$
Fused	**$1.12 \pm 1.08°$**	**$0.79 \pm 0.65°$**	**$0.97 \pm 0.76°$**	**$0.88 \pm 0.77°$**

the pose estimates of multiple framesets and fusion with the robot odometry via the pose graph give further improvements.

Tab. 5.3 shows the root mean square (RMS) of the translation error for the trajectories. The shorter Traj. 3, without WP 1 and WP 6 visible in only one camera, has the lowest trajectory error. The fused trajectory estimate using external camera pose estimation and robot odometry gives a significant improvement from 18.5 cm to 4.48 cm, averaged over the dataset.

In a second set of experiments, we compare the fused trajectory calculated on the central backend with the robot's internal estimate when integrating the pose correction feedback at static waypoints. For this, we record two iterations of a longer trajectory (~40 m, 3 times Traj. 3), with and without applying the feedback on the robot. Fig. 5.7 shows the translation error over the traveled distance. The pose correction feedback significantly improves the robot's localization, reducing the error to the order of magnitude of the fused path calculated on the backend. Comparing the RMS translation error of the complete trajectories, the robot-internal estimate without correction gives an

Table 5.3: Root mean square translation error in cm.

Pose Estimation	Traj. 1	Traj. 2	Traj. 3	**Avg.**
Robot	20.5	19.4	15.8	18.5
Fused	**5.76**	**4.64**	**3.03**	**4.48**

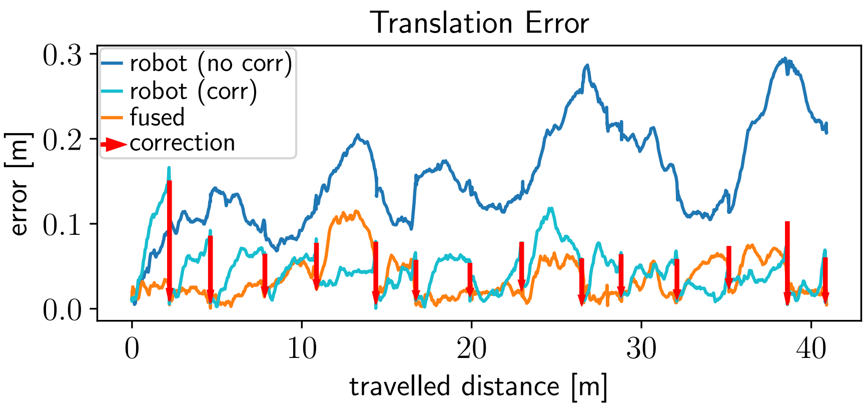

Figure 5.7: Translation error with and without applying pose correction feedback to the robot's localization for the second experiment.

error of 17.5 cm that is improved significantly to 5.51 cm by the pose correction feedback. The pose graph fusion, calculated on the central backend, further improves the RMSE to 3.34 cm. To integrate the pose graph fusion onto the robot, a closer coupling with the backend or a re-implementation of the robot navigation stack would be required.

5.4.3 *Collaborative Semantic Mapping*

In further experiments, we demonstrate the integration of the robot as a mobile sensor node into the smart edge sensor network. Fig. 5.8 shows the consistency of the robot observations with the allocentric scene model. Without pose correction feedback, error accumulates in the localization and the observations have low consistency with the model. After pose correction through the feedback from the smart edge sensors, the observations fit the model well and can consistently be fused into the semantic map.

We show the collaborative semantic mapping in a lab-scale experiment in Fig. 5.9. The semantic map is initialized from a prior model of the empty room, without any semantic information. The static smart edge sensors observe the room only partly, due to occlusions and limited measurement range, providing semantic information for ∼50 % of the voxels. The mobile robot provides changing sensor perspectives and can actively perceive the areas not observed by the static sensors. The waypoints for completing the map were defined manually in the experiment but could be set automatically using approaches for exploration and viewpoint optimization (S. Dong et al., 2019). Through collaboration, mobile robot and static sensors build a complete semantic map of the ∼240 m^2 environment with semantic information for over 90 % of the voxels.

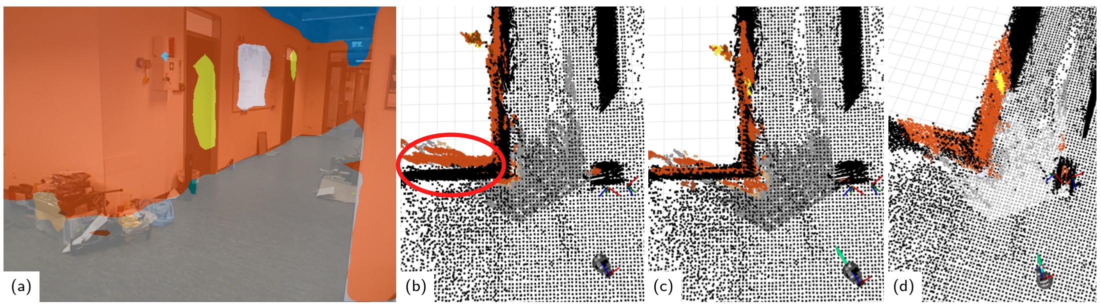

Figure 5.8: Consistency of robot observations with global map: (a) local robot view; (b) 3D view with accumulated drift after Traj. 3 and (c) after pose correction (green arrow); (d) robot observations fused into map.

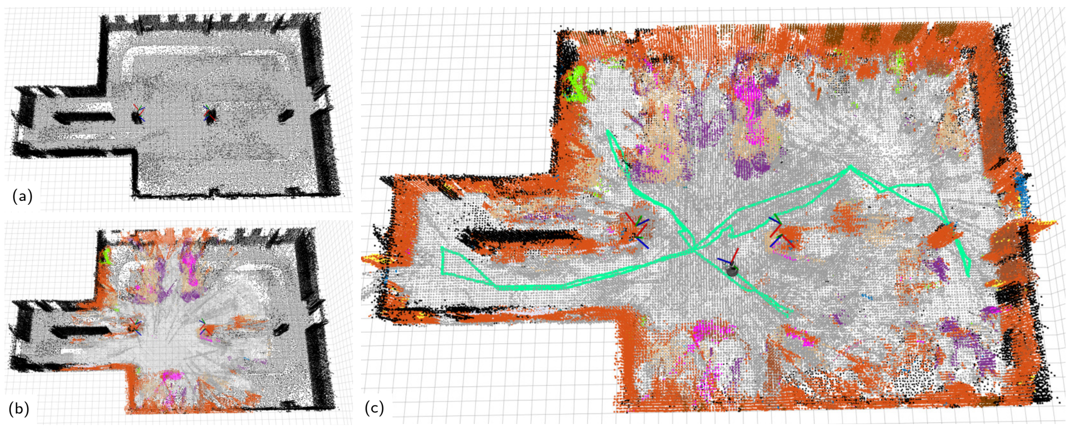

Figure 5.9: Completion of semantic map by fusing with robot observations: (a) initial 3D map, (b) incomplete semantic map with observations from static smart edge sensors, (c) semantic map completed with robot observations and exploration path.

5.4.4 *Localization Robustness*

To evaluate the robustness of our collaborative localization approach, we repeat the lab-scale experiment ten times, five times with pose correction feedback and five times without, and report the success rate in Tab. 5.4. Using only the internal LiDAR-based localization, the robot cannot reach all waypoints in three of five trials, an emergency stop due to collision with an obstacle occurred in one trial, and, hence, the success rate reaches only 1 / 5. Failures occur after 38 m of traveled distance, on average.

With the proposed localization feedback, the robot completes the ∼60 m long trajectory successfully in all trials. The external camera-based pose estimation compensates for the difficulties of the robot-internal navigation to localize in the highly cluttered, dynamic environment where only few distinctive features, such as walls or columns, are visible in the LiDAR, significantly increasing the system's robustness.

5.4.5 *Human-Aware Anticipatory Navigation*

In an additional experiment, we use the real-time 3D human pose tracking by the external smart edge sensors (cf. Chapters 2 and 3) to generate a semantic feedback message informing the robot about people in its vicinity but out of sight of its internal

Table 5.4: Robustness during 5+5 lab-scale experiments.

	WP not reached	Emergency Stop	Success Rate
w/o correction fb	3 / 5	1 / 5	1 / 5
w/ correction fb	**0 / 5**	**0 / 5**	**5 / 5**

sensors, e.g. due to occlusions or limited FoV. This enables anticipatory human-aware robot navigation where the robot can foresightedly adapt its navigation path e.g. to persons appearing from behind occlusions. This scenario is common in many household or office environments, e.g. at corridor intersections, as well as for warehouses with narrow aisles between high shelves, where people suddenly emerging from behind occlusions can be at risk of collision with autonomously operating robots. Fusing the root internal sensor views with semantic feedback from the instrumented environment, i.e. the network of external smart edge sensors, alleviates these issues and can enable the robot to "see around corners".

In our experiments, we implement the anticipatory human-aware robot navigation on a low planning level, using the local dynamic obstacle map of the robot navigation stack. This 2D grid map in robot coordinates can include multiple sensor sources (i.e. robot-internal 2D LiDAR and RGB-D camera, as well as semantic feedback from the sensor network) that provide information about occupied areas in the vicinity of the robot and is used to dynamically update the navigation path to avoid them.

We employ the 3D person skeletons and their respective linear root joint velocity estimated on the central backend to generate the semantic feedback message for the robot. For this, the joints of persons in the vicinity of the robot are transformed from allocentric to robot-centric coordinates, using the tracked robot pose and projected onto the ground plane. The projected joints are inflated by a safety margin and extrapolated 2 s into the future using the root velocity estimate. The calculated regions that are and will be occupied by persons are sent to the robot for integration into its dynamic obstacle avoidance map.

The anticipatory human-aware robot navigation experiment is illustrated in Fig. 5.10 for a scenario where a person emerges from behind an occluding wall and crosses the robot's path. Without feedback (Column (a)), the robot can react to the person only after they emerge from behind the wall and are visible in its own sensors. Robot and person come dangerously close (bottom row). With semantic feedback from the external sensors about tracked persons (Column (b)), the person with its linear velocity estimate is included in the robot's dynamic obstacle map, where regions they (prospectively) walk through are marked as occupied (dark gray color) even before they appear from behind the occlusion. The robot adapts its navigation path foresightedly and keeps a safe distance from the person crossing its originally planned path.

Table 5.5 further gives a quantitative evaluation of the minimum safety distance towards the human maintained by the robot during five iterations of the experiment with and without human pose feedback, respectively. We indicate the distance of the person's root joint projected to the floor towards the outer surface of the robot base. With semantic feedback about tracked persons, human and robot never come closer than 80 cm, keeping a safety distance of 86 cm on average over the iterations of the

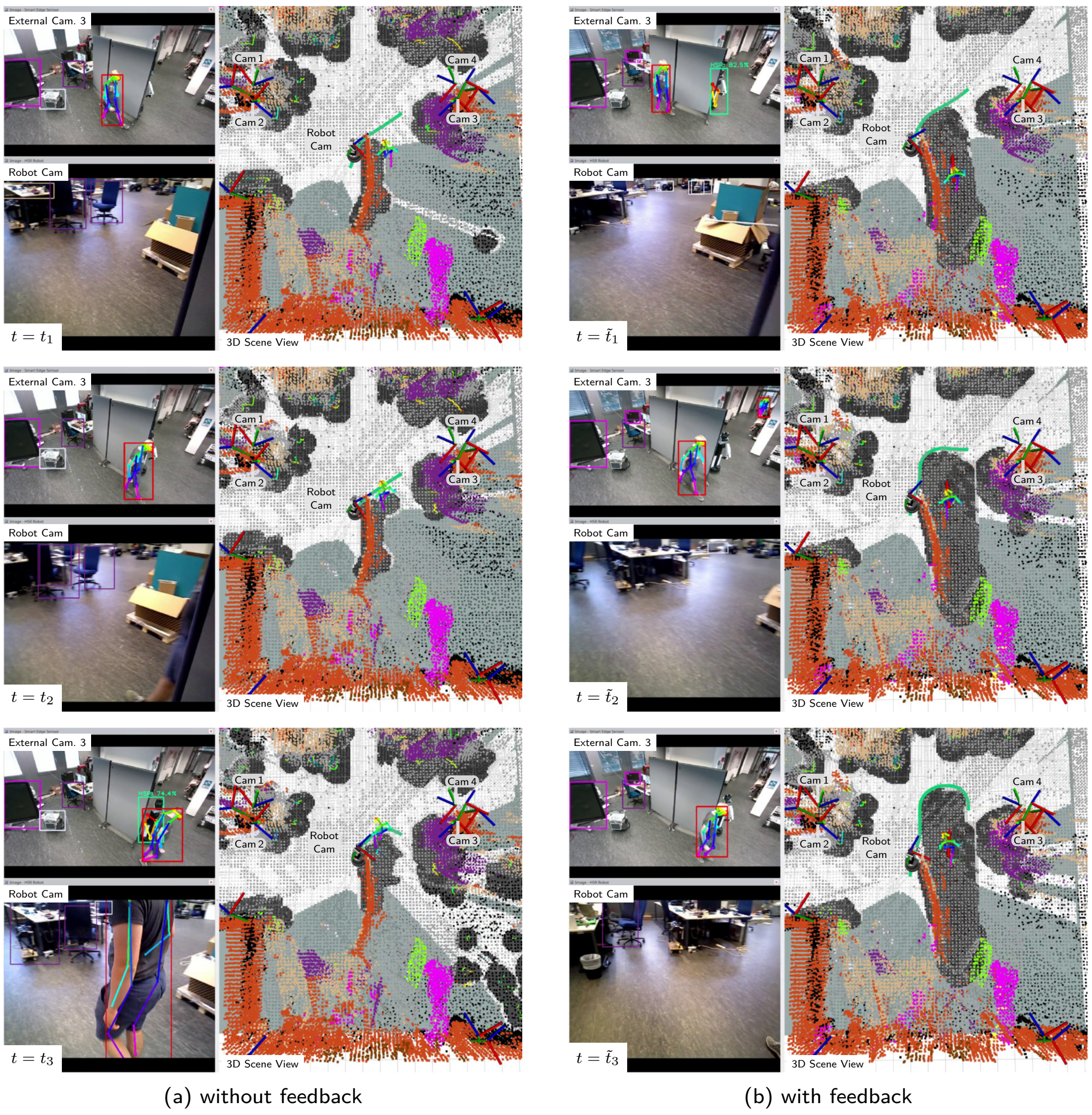

Figure 5.10: Human-aware anticipatory navigation: External and robot camera view, and 3D scene view for three time steps (from top to bottom) of a person walking out from behind an occluding wall and crossing the robot's path. The 3D scene view includes semantic map, tracked person and robot, planned robot path (green line), and obstacle avoidance map with *free* (white), *unknown* (light gray), and *occupied* (dark gray) areas. Semantic feedback from external smart edge sensors about tracked persons and their velocity (red arrow) enables the robot to anticipatorily adapt its navigation path to people emerging from behind occlusions.

experiment. Without feedback, the minimum safety distance between person and robot decreases to 10 cm or less in some iterations, posing a serious risk of collision. The safety distance only amounts to 29 cm on average.

Through the semantic human pose feedback, the HSR robot can anticipate a person emerging from behind an occlusion significantly earlier and anticipatorily adjust its navigation path to always maintain a sufficient safety distance.

Table 5.5: Minimum person–robot safety distance during anticipatory navigation experiment.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Avg.
w/o fb	54 cm	11 cm	7 cm	37 cm	35 cm	29 cm
w/ fb	89 cm	91 cm	82 cm	85 cm	83 cm	**86 cm**

5.5 Discussion

We presented a novel method for marker-less mobile robot pose estimation using multi-view keypoint detections from a network of external smart cameras. We use this to initialize and continuously update a mobile robot's localization in the allocentric scene model of the smart edge sensor network and build a system for collaborative perception between mobile robot and static smart edge sensors. The typical position error of our proposed method for mobile robot pose estimation is below 2.8 cm when detected in at least two cameras, and below 4.3 cm when detected in a single camera, while the robot localization typically deviates more than 19 cm after only 5 m of traveled distance.

Precisely initializing and tracking the robot's localization w.r.t the camera network allows to fuse its semantic observations in a globally consistent way into the allocentric scene model. The robot as a mobile sensor node provides changing viewpoints and can explore areas not covered by the static sensors due to occlusions and limited measurement range. We demonstrate a real-world application where a mobile robot and distributed smart edge sensors collaboratively build a 3D semantic map of a large room and the robot shows robust long-term operation capabilities in the challenging, cluttered environment. Furthermore, in an additional experiment on anticipatory human-aware robot navigation, we show that the robot can anticipate persons emerging from behind occlusions and anticipatorily adjust its navigation path to maintain a safe distance by incorporating semantic feedback of human pose observations from the external sensors.

In future work, we plan to use our system for collaborative perception to enable human-robot interaction in a shared workspace. The anticipatory human-aware navigation could be implemented on a higher planning level, instead of using the local obstacle avoidance map, taking long-term goals and intents of the persons into account.

6 Conclusions and Outlook

In this thesis, we developed a system for multi-view 3D semantic scene perception using a network of distributed smart edge sensors and mobile robots with on-board compute accelerators for local image processing. The smart edge sensors and robots communicate with a central backend that fuses all available sensor perspectives into an allocentric semantic scene model. The instrumentalization of the lab-scale workspace environment by installing numerous smart edge sensors with on-board semantic perception allows to efficiently and robustly build a comprehensive and detailed 3D semantic scene model online, in real time. Since the sensor data is semantically interpreted on-device, directly at its source, the communication between the sensor nodes and the central backend runs on a purely semantic level, and the raw images remain on the sensor boards. This significantly reduces the required network bandwidth and mitigates privacy issues for the observed persons.

We introduced the concept of semantic feedback to couple static and mobile smart edge sensors with the central backend through bidirectional communication at the semantic level, enabling information sharing among distributed sensors for collaborative perception. Semantic feedback allows global context information, such as fused multi-view human and robot pose estimates, to be incorporated into the local semantic model of each smart edge sensor. This improves pose estimation results, e.g., by resolving occlusions or detections that are ambiguous from a single sensor view, and enables mobile robots to preemptively adjust their navigation path, e.g., when a person emerges from an occluded area.

Since there are many different sensor perspectives, the failure of individual sensor nodes can be tolerated and compensated for by other overlapping views through collaborative information sharing. Mobile robots need less on-board perception when collaborating with the external sensor network as they can incorporate context information from the allocentric scene model via semantic feedback. On the other hand, all sensor nodes, including the mobile robots, maintain local short-term autonomy through on-board interpretation of their local sensor views, even in the case of communication failure. Semantic feedback sends relevant parts of the fused, allocentric scene model back to the individual sensor nodes, where the feedback information, if available, is fused with the local detections. Thus, semantic feedback expands the local perception fields of the sensors with global context information whenever an up-to-date feedback message is received, but the local view interpretation does not depend upon its availability.

Through collaborative information sharing via semantic feedback, mobile robots and static smart edge sensors deployed in challenging, cluttered, and dynamic real-world environments were demonstrated in this thesis to be able to build complete 3D semantic scene models including scene geometry, persons, and objects. We presented methods for real-time multi-view 3D human pose estimation and 3D semantic scene perception to build an allocentric 3D semantic scene model using static viewpoints of distributed smart edge sensors and changing viewpoints of mobile robots. The scene model contains dynamic 3D human poses estimated in real time and semantically annotated 3D scene geometry as a volumetric map with additional object-level pose and shape information. For real-time sensor data interpretation onboard the embedded inference accelerators, we adapted efficient CNN architectures for image and point cloud semantic segmentation, person and object detection, and human, robot, and object pose estimation, and retrained or fine-tuned them on task-specific datasets.

For real-time multi-view 3D multi-person pose estimation, each camera view is processed locally, onboard the respective embedded sensor, and only semantic person keypoint detections are transmitted to a central backend where they are fused into 3D skeleton models. Sets of corresponding keypoint detection messages are synchronized based on the detection timestamps, associated across camera views to person hypotheses based on the epipolar distance of their joints, and raw 3D poses are recovered via triangulation. The 3D poses are further refined using a skeleton model that incorporates prior information on the typical bone lengths of the human skeleton. Semantic feedback is implemented by reprojecting the allocentric multi-view human pose estimate into individual sensor views, where it is fused with the local detections to resolve occluded or ambiguous joint detections. The pipeline was evaluated on the Human 3.6M, Campus, and Shelf datasets where it achieves state-of-the-art results, as well as on own data in challenging real-world scenarios with up to 16 cameras and six persons. Up to three persons can be tracked at the full sensor frame rate of 30 Hz and up to six persons at 15 Hz, achieving real-time performance.

As the multi-view fusion of local joint detections to 3D pose estimates requires the relative camera poses to be known, we developed methods for online marker-free extrinsic camera calibration requiring only a rough, tape-measure-based initialization. Factor graph optimization problems are repeatedly solved to estimate the camera poses constrained by the synchronized sets of person keypoint observations. The calibration method was designed to be robust against occlusions and false or sparse sets of detections, and is free of many typical assumptions of similar methods: It does not require a specific calibration target, can cope with and exploit detections of multiple persons simultaneously, and handles arbitrary person poses. We showed that our results are more accurate than the reference calibration obtained by an offline method based on traditional calibration targets. Our calibration reliably achieved lower reprojection errors in the 3D multi-person pose estimation pipeline used as an evaluation scenario.

To build a complete 3D semantic scene model containing 3D geometry and object instances in addition to dynamic human poses, we extended the smart edge sensor network with additional sensor nodes with enhanced computational capabilities and RGB-D and thermal cameras. Semantically annotated point clouds and pose and shape information of detected furniture objects are streamed from the sensors to a central backend in addition to the 2D human keypoint estimates, which are augmented with the RGB-D depth.

The semantic point clouds from multiple views are aggregated into an allocentric volumetric semantic map through Bayesian fusion. Individual objects are represented in the semantic map via an a-priori known 3D mesh model or a volumetric sub-map that is learned online. Only a few semantic object properties, such as estimated pose and point measurement distribution variances are transmitted from the sensors to the backend, and object candidates from multiple views are associated and fused.

A two-stage approach of keypoint detection and PnP pose estimation was implemented for object pose estimation. Keypoints were defined on prominent geometric features of the respective object model and synthetic data for training keypoint detection CNNs was generated using a photo-realistic rendering framework. The usage of only synthetic training data allows to extend the method easily to different object classes. From the keypoint detections, object poses are calculated via the PnP-RANSAC algorithm using 2D-3D correspondences between detected and model keypoints and refined via ICP alignment when depth data is available. We evaluated the method for object pose estimation on the single-view YCB-V dataset and the multi-view Behave dataset where it achieved pose errors below 9 cm and 9° with online input data processing using lightweight CNN architectures deployed on the embedded sensor hardware.

The extended sensor network consists of 20 smart edge sensors, thereof 4 based on the novel Jetson NX board, covering an area of about 12×22 m in a real-world lab environment. We evaluated the performance and robustness of the proposed system for multi-view 3D semantic scene perception in challenging, cluttered real-world scenes with up to 8 persons and different furniture objects. Dynamic human motions are estimated in real time and the semantically annotated 3D geometry provides a complete scene view that explains interactions between persons and objects in a physically plausible manner. Multiple chairs and a table are tracked through the scene online, in real time, even under high occlusions.

The concept of *smart edge sensors* with local on-device semantic image processing was further extended from static sensor boards to mobile aerial and ground robots. We proposed a UAV platform with on-board real-time multi-modal semantic perception as a mobile smart edge sensor operating outdoors. We evaluated the computational efficiency of different embedded inference accelerators connected to the UAV computer. The Edge TPU performs inference in 8-bit quantized mode and showed more efficient CPU usage. The iGPU is more flexible, e. g. to directly run pre-trained models, as it uses 16- or 32-bit floating-point precision and does not require model quantization. While the previous scenarios used multiple, static smart edge sensors, here we used a single, but moving, sensor node. A LiDAR sensor was used in addition to the RGB-D cameras, providing precise range measures also at large distance. We addressed domain adaptation issues between sensors with different FoVs for the LiDAR semantic segmentation CNN, retraining it with data captured on our UAV using pseudo-labels automatically obtained from the aggregated image-based semantic map, providing cross-domain supervision. This takes up the idea of semantic feedback, reprojecting parts of the fused allocentric semantic model into individual sensor views to provide a reliable source of semantic information. With label propagation, the 3D segmentation accuracy of the proposed system significantly improved for the full 360° LiDAR FoV. We evaluated the system in real-world experiments in an urban environment and at a disaster test site, showing coherent semantic perception of diverse and challenging scenes.

We combined the previous approaches to the final extent of the proposed system for collaborative semantic scene perception: A mobile service robot was incorporated into the network of static smart edge sensors to collaboratively build a more complete semantic scene model. The external smart edge sensors estimate the mobile robot's pose using multi-view keypoint detections to initialize and continuously update its localization in the allocentric scene model. We evaluated the accuracy of the external pose estimation using an additional marker-based tracking system as a reference. The typical position error is below 2.8 cm when the robot is detected in at least two cameras, and below 4.3 cm when detected in a single camera.

Using the robot as a mobile sensor node, we built a system for collaborative perception between mobile robot and static smart edge sensors. The robot provides changing viewpoints and actively perceives areas not observed by the static sensors. Fusing its semantic observations in a globally consistent way into the allocentric scene model extends the coverage and level of detail of the semantic map. Through semantic feedback of human pose observations from the external sensors, the robot can anticipate people emerging from behind occlusions and preemptively adjust its navigation path to maintain a sufficient safety distance.

The proposed approaches using semantic feedback for collaborative semantic scene perception lay the foundations for many potential human-robot interactive and collaborative application tasks that go beyond the collaborative semantic mapping and human-aware navigation demonstrated in this thesis.

Future applications of the developed system for collaborative semantic perception include service robot tasks with human–robot interaction based on the semantic scene model built from the distributed smart edge sensors. In a collaborative assembly task, a mobile service robot could anticipatorily bring parts or tools needed by a person for the upcoming assembly steps. These parts and tools could be localized throughout the capture space by the distributed sensors. Also, the transformation of room layouts by autonomously or collaboratively moving chairs and tables in a shared office–teaching space, or household tasks like laying the table could be realized. The human-aware anticipatory navigation could be implemented on a higher planning level instead of the local obstacle avoidance map, taking longer-term intentions of persons into account.

Further directions for future work are related to extending and improving the semantic scene model. The data association in multi-person scenes could be improved by using visual re-id descriptors on top of geometric cues and modeling multiple hypotheses for data association could help to resolve ambiguities. Predictions locally in sensor coordinates could provide additional input complementary to local detections and received semantic feedback. Additionally, the semantic feedback of dynamic human poses could be improved by using a more elaborate, deep-learning-based predictive motion model on the central backend. Methods from federated learning (Brecko et al., 2022) could be applied to collaboratively improve the local CNN models without sharing privacy-sensitive raw data. Last but not least, a more unified scene representation could be found, based on semantic neural radiance fields (Suhani Vora et al., 2022; Zhi et al., 2021) that implicitly model the scene geometry, semantics, and person, object, and robot models which currently are explicitly instantiated in the scene model.

Bibliography

Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. (2016). "TensorFlow: A system for large-scale machine learning." In: *USENIX symposium on operating systems design and implementation (OSDI)*, pp. 265–283.

Agarwal, Sameer, Keir Mierle, and The Ceres Solver Team (Mar. 2022). *Ceres Solver*. Version 2.1. URL: https://github.com/ceres-solver/ceres-solver.

Ahmed, Imran, Sadia Din, Gwanggil Jeon, Francesco Piccialli, and Giancarlo Fortino (2021). "Towards collaborative robotics in top view surveillance: A framework for multiple object tracking by detection using deep learning." In: *IEEE/CAA Journal of Automatica Sinica* 8.7, pp. 1253–1270. DOI: 10.1109/JAS.2020.1003453.

Alonso, Iñigo, Luis Riazuelo, Luis Montesano, and Ana Murillo (2021). "Domain adaptation in LiDAR semantic segmentation by aligning class distributions." In: *International Conference on Informatics in Control, Automation and Robotics - ICINCO,* pp. 330–337. DOI: 10.5220/0010610703300337.

Amanatides, John and Andrew Woo (1987). "A fast voxel traversal algorithm for ray tracing." In: *8th European Computer Graphics Conference and Exhibition (EuroGraphics)*. DOI: 10.2312/egtp.19871000.

Amini, Arash, Hafez Farazi, and Sven Behnke (2022). "Real-time pose estimation from images for multiple humanoid robots." In: *RoboCup International Symposium*, pp. 91–102. DOI: 10.1007/978-3-030-98682-7_8.

Amini, Arash, Arul Selvam Periyasamy, and Sven Behnke (2021). "T6D-Direc: Transformers for multi-object 6D pose direct regression." In: *43th DAGM German Conference on Pattern Recognition (GCPR)*, pp. 530–544. DOI: 10.1007/978-3-030-92659-5_34.

– (2022). "YOLOPose: Transformer-based multi-object 6D pose estimation using keypoint regression." In: *International Conference on Intelligent Autonomous Systems (IAS)*. DOI: doi.org/10.1007/978-3-031-22216-0_27.

Ashraf, Muhammad Waseem, Waqas Sultani, and Mubarak Shah (2021). "Dogfight: Detecting drones from drones videos." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7067–7076. DOI: 10.1109/CVPR46437.2021.00699.

Bartol, Kristijan, David Bojanić, and Tomislav Petković (2022). "Generalizable human pose triangulation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11018–11027. DOI: 10.1109/CVPR52688.2022.01075.

Bartolomei, Luca, Lucas Teixeira, and Margarita Chli (2020). "Perception-aware path planning for UAVs using semantic segmentation." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5808–5815. DOI: 10.1109/IROS45743.2020.9341347.

Bauer, Peter, Werner Lienhart, and Samuel Jost (2021). "Accuracy investigation of the pose determination of a VR system." In: *Sensors* 21.5. DOI: 10.3390/s21051622.

Behley, J., M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall (2019). "SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences." In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 9296–9306. DOI: 10.1109/ICCV.2019.00939.

Belagiannis, Vasileios, Sikandar Amin, Mykhaylo Andriluka, Bernt Schiele, Nassir Navab, and Slobodan Ilic (2014). "3D pictorial structures for multiple human pose estimation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1669–1676. DOI: 10.1109/CVPR.2014.216.

Belagiannis, Vasileios, Xinchao Wang, Bernt Schiele, Pascal Fua, Slobodan Ilic, and Nassir Navab (2015). "Multiple human pose estimation with temporally consistent 3D pictorial structures." In: *Computer Vision - ECCV 2014 Workshops*. Springer, pp. 742–754. DOI: 10.1007/978-3-319-16178-5_52.

Bergstra, J., D. Yamins, and D. D. Cox (2013). "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures." In: *International Conference on Machine Learning (ICML)*, pp. 115–123. DOI: 10.5555/3042817.3042832.

Berrio, Julie Stephany, Mao Shan, Stewart Worrall, and Eduardo Nebot (2022). "Camera-LIDAR integration: Probabilistic sensor fusion for semantic mapping." In: *IEEE Transactions on Intelligent Transportation Systems* 23.7, pp. 7637–7652. DOI: 10.1109/TITS.2021.3071647.

Beul, Marius, Simon Bultmann, Andre Rochow, Radu Alexandru Rosu, Daniel Schleich, Malte Splietker, and Sven Behnke (2020). "Visually guided balloon popping with an autonomous MAV at MBZIRC 2020." In: *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 34–41. DOI: 10.1109/SSRR50563.2020.9292612.

Beul, Marius, Max Schwarz, Jan Quenzel, Malte Splietker, Simon Bultmann, Daniel Schleich, Andre Rochow, Dmytro Pavlichenko, Radu Rosu, Patrick Lowin, Bruno Scheider, Michael Schreiber, Finn Süberkrüb, and Sven Behnke (2022). "Target chase, wall building, and fire fighting: Autonomous UAVs of team NimbRo at MBZIRC 2020." In: *Field Robotics* 2, pp. 807–842. DOI: 10.55417/fr.2022027.

Bhardwaj, Romil, Gopi Krishna Tummala, Ganesan Ramalingam, Ramachandran Ramjee, and Prasun Sinha (2018). "AutoCalib: Automatic traffic camera calibration at scale." In: *ACM Transactions on Sensor Networks (TOSN)* 14.3-4, pp. 1–27. DOI: 10.1145/3199667.

Bhatnagar, Bharat Lal, Xianghui Xie, Ilya Petrov, Cristian Sminchisescu, Christian Theobalt, and Gerard Pons-Moll (2022). "BEHAVE: Dataset and method for tracking human object interactions." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15914–15925. DOI: 10.1109/CVPR52688.2022.01547.

Boltres, Andreas, Angel Villar-Corrales, Jan Nogga, and Peer Schütt (2022). *SL-Cutscenes*. URL: https://github.com/AIS-Bonn/sl-scenes.

Brachmann, Eric, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, and Carsten Rother (2016). "Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3364–3372. DOI: 10.1109/CVPR.2016.366.

Bradski, G. (2000). "The OpenCV library." In: *Dr. Dobb's Journal of Software Tools*.

Brecko, Alexander, Erik Kajati, Jiri Koziorek, and Iveta Zolotova (2022). "Federated Learning for Edge Computing: A Survey." In: *Applied Sciences* 12.18. DOI: 10.3390/app12189124.

Bultmann, Simon and Sven Behnke (2021). "Real-time multi-view 3D human pose estimation using semantic feedback to smart edge sensors." In: *Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2021.XVII.040.

– (2022). "3D semantic scene perception using distributed smart edge sensors." In: *International Conference on Intelligent Autonomous Systems (IAS)*, pp. 313–329. DOI: 10.1007/978-3-031-22216-0_22.

Bultmann, Simon, Raphael Memmesheimer, and Sven Behnke (2023). "External camera-based mobile robot pose estimation for collaborative perception with smart edge sensors." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8194–8200. DOI: 10.1109/ICRA48891.2023.10160892.

Bultmann, Simon, Jan Quenzel, and Sven Behnke (2021). "Real-time multi-modal semantic fusion on unmanned aerial vehicles." In: *European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR50962.2021.9568812.

– (2023). "Real-time multi-modal semantic fusion on unmanned aerial vehicles with label propagation for cross-domain adaptation." In: *Robotics and Autonomous Systems* 159, p. 104286. DOI: 10.1016/j.robot.2022.104286.

Burenius, Magnus, Josephine Sullivan, and Stefan Carlsson (2013). "3D pictorial structures for multiple view articulated pose estimation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3618–3625. DOI: 10.1109/CVPR.2013.464.

Cao, Zhe, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh (2021). "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.1, pp. 172–186. DOI: 10.1109/TPAMI.2019.2929257.

Capellen, Catherine, Max Schwarz, and Sven Behnke (2020). "ConvPoseCNN: Dense convolutional 6D object pose estimation." In: *International Conference on Computer Vision Theory and Application (VISSAPP)*, pp. 162–172.

Chakravarty, Punarjay and Ray Jarvis (2009). "External cameras and a mobile robot: A collaborative surveillance system." In: *Australasian Conference on Robotics and Automation (ACRA)*.

Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam (2018). "Encoder-decoder with atrous separable convolution for semantic image segmentation." In: *European Conference on Computer Vision (ECCV)*, pp. 833–841. DOI: 10.1007/978-3-030-01234-2_49.

Chen, Long, Haizhou Ai, Rui Chen, Zijie Zhuang, and Shuang Liu (2020). "Cross-view tracking for multi-human 3D pose estimation at over 100 FPS." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3276–3285. DOI: 10.1109/CVPR42600.2020.00334.

Chen, Xieyuanli, Andres Milioto, Emanuele Palazzolo, Philippe Giguere, Jens Behley, and Cyrill Stachniss (2019). "SuMa++: Efficient LiDAR-based semantic SLAM." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4530–4537. DOI: 10.1109/IROS40897.2019.8967704.

Clapés, Albert, Julio C. S. Jacques Junior, Carla Morral, and Sergio Escalera (2020). "ChaLearn LAP 2020 challenge on identity-preserved human detection: Dataset and results." In: *IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, pp. 801–808. DOI: 10.1109/FG47880.2020.00135.

Cortinhal, Tiago, George Tzelepis, and Eren Erdal Aksoy (2020). “SalsaNext: Fast, uncertainty-aware semantic segmentation of LiDAR point clouds.” In: *International Symposium on Visual Computing*, pp. 207–222. DOI: 10.1007/978-3-030-64559-5_16.

Dalal, Navneet and Bill Triggs (2005). “Histograms of oriented gradients for human detection.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 886–893. DOI: 10.1109/CVPR.2005.177.

Dellaert, Frank and Michael Kaess (2017). “Factor graphs for robot perception.” In: *Foundations and Trends in Robotics (FNT)* 6.1-2, pp. 1–139. DOI: 10.1561/2300000043.

Dengler, Nils, Tobias Zaenker, Francesco Verdoja, and Maren Bennewitz (2021). “Online object-oriented semantic mapping and map updating.” In: *European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR50962.2021.9568817.

Deschaud, Jean-Emmanuel, David Duque, Jean Pierre Richa, Santiago Velasco-Forero, Beatriz Marcotegui, and François Goulette (2021). “Paris-CARLA-3D: A real and synthetic outdoor point cloud dataset for challenging tasks in 3D mapping.” In: *Remote Sensing* 13.22, p. 4713. DOI: 10.3390/rs13224713.

Dong, Junting, Wen Jiang, Qixing Huang, Hujun Bao, and Xiaowei Zhou (2019). “Fast and robust multi-person 3D pose estimation from multiple views.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7784–7793. DOI: 10.1109/CVPR.2019.00798.

Dong, Siyan, Kai Xu, Qiang Zhou, Andrea Tagliasacchi, Shiqing Xin, Matthias Nießner, and Baoquan Chen (2019). “Multi-robot collaborative dense scene reconstruction.” In: *ACM Transactions on Graphics (TOG)* 38.4, pp. 1–16. DOI: 10.1145/3306346.3322942.

Felzenszwalb, Pedro F. and Daniel P. Huttenlocher (2005). “Pictorial Structures for object recognition.” In: *International Journal of Computer Vision (IJCV)* 61.1, pp. 55–79. DOI: 10.1023/B:VISI.0000042934.15159.49.

Fischler, Martin A. and Robert C. Bolles (1981). “Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography.” In: *Communications of the ACM* 24.6, pp. 381–395. DOI: 10.1145/358669.358692.

Fischler, Martin A. and Robert A. Elschlager (1973). “The Representation and matching of Pictorial Structures.” In: *IEEE Transactions on Computers* C-22.1, pp. 67–92. DOI: 10.1109/T-C.1973.223602.

Gawel, Abel, Carlo Del Don, Roland Siegwart, Juan Nieto, and Cesar Cadena (2018). “X-view: Graph-based semantic multi-view localization.” In: *IEEE Robotics and Automation Letters (RA-L)* 3.3, pp. 1687–1694. DOI: 10.1109/LRA.2018.2801879.

Geiger, Andreas, Philip Lenz, and Raquel Urtasun (2012). “Are we ready for autonomous driving? The KITTI vision benchmark suite.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

Grinvald, Margarita, Fadri Furrer, Tonci Novkovic, Jen Jen Chung, Cesar Cadena, Roland Siegwart, and Juan Nieto (2019). “Volumetric instance-aware semantic mapping and 3D object discovery.” In: *IEEE Robotics and Automation Letters (RA-L)* 4.3, pp. 3037–3044. DOI: 10.1109/LRA.2019.2923960.

Grinvald, Margarita, Federico Tombari, Roland Siegwart, and Juan Nieto (2021). “TSDF++: A multi-object formulation for dynamic object tracking and reconstruction.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14192–14198. DOI: 10.1109/ICRA48506.2021.9560923.

Guan, Junzhi, Francis Deboeverie, Maarten Slembrouck, Dirk Van Haerenborgh, Dimitri Van Cauwelaert, Peter Veelaert, and Wilfried Philips (2016). “Extrinsic calibration of camera networks based on pedestrians.” In: *Sensors* 16.5. DOI: 10.3390/s16050654.

Hartley, Richard and Andrew Zisserman (2003). *Multiple view geometry in computer vision.* 2nd ed. Cambridge University Press. ISBN: 0521540518.

Hau, Julian, Simon Bultmann, and Sven Behnke (2022). “Object-level 3D semantic mapping using a network of smart edge sensors.” In: *6th IEEE International Conference on Robotic Computing (IRC)*, pp. 198–206. DOI: 10.1109/IRC55401.2022.00041.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick (2017). “Mask R-CNN.” In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep residual learning for image recognition.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

He, Yihui, Rui Yan, Katerina Fragkiadaki, and Shoou-I Yu (2020). “Epipolar transformers.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7776–7785. DOI: 10.1109/CVPR42600.2020.00780.

Henning, D, T Laidlow, and S Leutenegger (2022). “BodySLAM: Joint camera localisation, mapping, and human motion tracking.” In: *European Conference on Computer Vision (ECCV)*, pp. 656–673. DOI: 10.1007/978-3-031-19842-7_38.

Hinterstoisser, Stefan, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab (2012). “Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes.” In: *Asian Conference on Computer Vision (ACCV)*, pp. 548–562. DOI: 10.1007/978-3-642-37331-2_42.

Hodaň, Tomáš, Dániel Baráth, and Jiří Matas (2020). “EPOS: Estimating 6D pose of objects with symmetries.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11700–11709. DOI: 10.1109/CVPR42600.2020.01172.

Hodaň, Tomáš, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas (2020). “BOP challenge 2020 on 6D object localization.” In: *European Conference on Computer Vision (ECCV) Workshops*, pp. 577–594. DOI: 10.1007/978-3-030-66096-3_39.

Hodaň, Tomáš et al. (2018). “BOP: Benchmark for 6D object pose estimation.” In: *European Conference on Computer Vision (ECCV)*, pp. 19–35. DOI: 10.1007/978-3-030-01249-6_2.

Hödlmoser, Michael and Martin Kampel (2010). “Multiple camera self-calibration and 3D reconstruction using pedestrians.” In: *International Symposium on Visual Computing (ISVC)*. DOI: 10.1007/978-3-642-17274-8_1.

Hornung, Armin, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard (2013). “OctoMap: An efficient probabilistic 3D mapping framework based on octrees.” In: *Autonomous Robots* 34.3, pp. 189–206. DOI: 10.1007/s10514-012-9321-0.

Howard, Andrew, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam (2019). “Searching for MobileNetV3.” In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1314–1324. DOI: 10.1109/ICCV.2019.00140.

Howard, Andrew, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam (2017). "Mobilenets: Efficient convolutional neural networks for mobile vision applications." In: *preprint arXiv:1704.04861*.

Huang, Yinghao, Omid Taheri, Michael J. Black, and Dimitrios Tzionas (2022). "InterCap: Joint markerless 3D tracking of humans and objects in interaction." In: *44th DAGM German Conference on Pattern Recognition (GCPR)*, pp. 281–299. DOI: 10.1007/978-3-031-16788-1_18.

Huynh, Du Q. (2009). "Metrics for 3D rotations: Comparison and analysis." In: *Journal of Mathematical Imaging and Vision* 35.2, pp. 155–164.

Ionescu, Catalin, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu (2014). "Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7, pp. 1325–1339. DOI: 10.1109/TPAMI.2013.248.

Iskakov, Karim, Egor Burkov, Victor Lempitsky, and Yury Malkov (2019). "Learnable triangulation of human pose." In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 7717–7726. DOI: 10.1109/ICCV.2019.00781.

Jacob, Benoit, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko (2018). "Quantization and training of neural networks for efficient integer-arithmetic-only inference." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2704–2713. DOI: 10.1109/CVPR.2018.00286.

Jaritz, Maximilian, Tuan-Hung Vu, Raoul de Charette, Emilie Wirbel, and Patrick Pérez (2020). "xMUDA: Cross-modal unsupervised domain adaptation for 3D semantic segmentation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12602–12611. DOI: 10.1109/CVPR42600.2020.01262.

Joshi, Bharat, Md Modasshir, Travis Manderson, Hunter Damron, Marios Xanthidis, Alberto Quattrini Li, Ioannis Rekleitis, and Gregory Dudek (2020). "DeepURL: Deep pose estimation framework for underwater relative localization." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1777–1784. DOI: 10.1109/IROS45743.2020.9341201.

Julier, Simon J and Jeffrey K Uhlmann (2004). "Unscented filtering and nonlinear estimation." In: *Proceedings of the IEEE* 92.3, pp. 401–422. DOI: 10.1109/JPROC.2003.823141.

Kaess, Michael, Hordur Johannsson, Richard Roberts, Viorela Ila, John J Leonard, and Frank Dellaert (2012). "iSAM2: Incremental smoothing and mapping using the Bayes tree." In: *The International Journal of Robotics Research (IJRR)* 31.2, pp. 216–235. DOI: 10.1177/0278364911430419.

Karrer, Marco, Patrik Schmuck, and Margarita Chli (2018). "CVI-SLAM–Collaborative visual-inertial SLAM." In: *IEEE Robotics and Automation Letters (RA-L)* 3.4, pp. 2762–2769. DOI: 10.1109/LRA.2018.2837226.

Kendall, Alex, Matthew Grimes, and Roberto Cipolla (2015). "PoseNet: A convolutional network for real-time 6-DOF camera relocalization." In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2938–2946. DOI: 10.1109/ICCV.2015.336.

Komorowski Jacekand Rokita, Przemysław (2012). "Extrinsic camera calibration method and its performance evaluation." In: *Computer Vision and Graphics*, pp. 129–138. DOI: 10.1007/978-3-642-33564-8_16.

Kruijff-Korbayová, Ivana et al. (2021). "German Rescue Robotics Center (DRZ): A holistic approach for robotic systems assisting in emergency response." In: *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, pp. 138–145. DOI: 10.1109/SSRR53300.2021.9597869.

Labbé, Yann, Justin Carpentier, Mathieu Aubry, and Josef Sivic (2020). "CosyPose: Consistent multi-view multi-object 6D pose estimation." In: *European Conference on Computer Vision (ECCV)*, pp. 574–591. DOI: 10.1007/978-3-030-58520-4_34.

Landgraf, Zoe, Fabian Falck, Michael Bloesch, Stefan Leutenegger, and Andrew J. Davison (2020). "Comparing view-based and map-based semantic labelling in real-time SLAM." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6884–6890. DOI: 10.1109/ICRA40945.2020.9196843.

Langer, Ferdinand, Andres Milioto, Alexandre Haag, Jens Behley, and Cyrill Stachniss (2020). "Domain transfer for semantic segmentation of LiDAR data using deep neural networks." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8263–8270. DOI: 10.1109/IROS45743.2020.9341508.

Lee, Timothy E, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Oliver Kroemer, Dieter Fox, and Stan Birchfield (2020). "Camera-to-robot pose estimation from a single image." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9426–9432. DOI: 10.1109/ICRA40945.2020.9196596.

Lepetit, Vincent, Francesc Moreno-Noguer, and Pascal Fua (2008). "EPnP: An accurate O(n) solution to the PnP problem." In: *International Journal of Computer Vision (IJCV)* 81.2, p. 155. DOI: 10.1007/s11263-008-0152-6.

Li, Jiefeng, Can Wang, Hao Zhu, Yihuan Mao, Hao-Shu Fang, and Cewu Lu (2019). "CrowdPose: Efficient crowded scenes pose estimation and a new benchmark." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10855–10864. DOI: 10.1109/CVPR.2019.01112.

Li, Zhigang, Gu Wang, and Xiangyang Ji (2019). "CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation." In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 7677–7686. DOI: 10.1109/ICCV.2019.00777.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick (2014). "Microsoft COCO: Common objects in context." In: *European Conference on Computer Vision (ECCV)*, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

Liu, Bin, Zhirong Wu, Han Hu, and Stephen Lin (2019). "Deep metric transfer for label propagation with limited annotated data." In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 1317–1326. DOI: 10.1109/ICCVW.2019.00167.

Liu, Jingchen, Robert T. Collins, and Yanxi Liu (2011). "Surveillance camera autocalibration based on pedestrian height distributions." In: *British Machine Vision Conference (BMVC)*. DOI: 10.5244/C.25.

– (2013). "Robust autocalibration for a surveillance camera network." In: *IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 433–440. DOI: 10.1109/WACV.2013.6475051.

Liu, Jinhui, Zhikang Zou, Xiaoqing Ye, Xiao Tan, Errui Ding, Feng Xu, and Xin Yu (2020). "Leaping from 2D detection to efficient 6DoF object pose estimation." In:

European Conference on Computer Vision (ECCV) Workshops, pp. 707–714. DOI: 10.1007/978-3-030-66096-3_47.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg (2016). “SSD: Single shot multibox detector.” In: *European Conference on Computer Vision (ECCV)*, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2.

Liu, Zhengzhe, Xiaojuan Qi, and Chi-Wing Fu (2021). “One thing one click: A self-training approach for weakly supervised 3D semantic segmentation.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1726–1736. DOI: 10.1109/CVPR46437.2021.00177.

Loper, Matthew, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black (2015). “SMPL: A skinned multi-person linear model.” In: *ACM Transactions on Graphics* 34.6. DOI: 10.1145/2816795.2818013.

Lowe, D.G. (1999). “Object recognition from local scale-invariant features.” In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 2, pp. 1150–1157. DOI: 10.1109/ICCV.1999.790410.

Lu, Jingpei, Florian Richter, and Michael C Yip (2022). “Pose estimation for robot manipulators via keypoint optimization and sim-to-real transfer.” In: *IEEE Robotics and Automation Letters (RA-L)* 7.2, pp. 4622–4629. DOI: 10.1109/LRA.2022.3151981.

Lv, Fengjun, Tao Zhao, and R. Nevatia (2006). “Camera calibration from video of a walking human.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.9, pp. 1513–1518. DOI: 10.1109/TPAMI.2006.178.

Mascaro, Ruben, Lucas Teixeira, and Margarita Chli (2021). “Diffuser: Multi-view 2D-to-3D label diffusion for semantic scene segmentation.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13589–13595. DOI: 10.1109/ICRA48506.2021.9561801.

Mathis, Alexander, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge (2018). “DeepLabCut: Markerless pose estimation of user-defined body parts with deep learning.” In: *Nature Neuroscience* 21.9, pp. 1281–1289.

Maturana, Daniel, Sankalp Arora, and Sebastian Scherer (2017). “Looking forward: A semantic mapping system for scouting with micro-aerial vehicles.” In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6691–6698. DOI: 10.1109/IROS.2017.8206585.

Maye, Jérôme, Paul Furgale, and Roland Siegwart (2013). “Self-supervised calibration for robotic systems.” In: *IEEE Intelligent Vehicles Symposium (IV)*, pp. 473–480. DOI: 10.1109/IVS.2013.6629513.

McCormac, John, Ankur Handa, Andrew Davison, and Stefan Leutenegger (2017). “SemanticFusion: Dense 3D semantic mapping with convolutional neural networks.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4628–4635. DOI: 10.1109/ICRA.2017.7989538.

Meger, David, Dimitri Marinakis, Ioannis Rekleitis, and Gregory Dudek (2009). “Inferring a probability distribution function for the pose of a sensor network using a mobile robot.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 756–762. DOI: 10.1109/ROBOT.2009.5152800.

Meyer, Gregory P., Jake Charland, Darshan Hegde, Ankit Laddha, and Carlos Vallespi-Gonzalez (2019). “Sensor fusion for joint 3D object detection and semantic segmentation.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 1230–1237. DOI: 10.1109/CVPRW.2019.00162.

Milioto, Andres, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss (2019). “RangeNet++: Fast and accurate LiDAR semantic segmentation.” In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4213–4220. DOI: 10.1109/IROS40897.2019.8967762.

Naikal, Nikhil, Pedram Lajevardi, and Shankar. S. Sastry (2014). “Joint detection and recognition of human actions in wireless surveillance camera networks.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4747–4754. DOI: 10.1109/ICRA.2014.6907554.

Neuhold, Gerhard, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder (2017). “The Mapillary Vistas dataset for semantic understanding of street scenes.” In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 5000–5009. DOI: 10.1109/ICCV.2017.534.

Nguyen, Ty, Shreyas S. Shivakumar, Ian D. Miller, James Keller, Elijah S. Lee, Alex Zhou, Tolga Özaslan, Giuseppe Loianno, Joseph H. Harwood, Jennifer Wozencraft, Camillo J. Taylor, and Vijay Kumar (2019). “MAVNet: An effective semantic segmentation micro-network for MAV-based tasks.” In: *IEEE Robotics and Automation Letters (RA-L)* 4.4, pp. 3908–3915. DOI: 10.1109/LRA.2019.2928734.

Oleynikova, Helen, Zachary Taylor, Marius Fehr, Roland Siegwart, and Juan Nieto (Sept. 2017). “Voxblox: Incremental 3D Euclidean signed distance fields for on-board MAV planning.” In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/iros.2017.8202315.

Olson, Edwin (2011). “AprilTag: A robust and flexible visual fiducial system.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3400–3407. DOI: 10.1109/ICRA.2011.5979561.

Pasqualetto Cassinis, Lorenzo, Robert Fonod, Eberhard Gill, Ingo Ahrns, and Jesus Gil Fernandez (2020). “CNN-based pose estimation system for close-proximity operations around uncooperative spacecraft.” In: *AIAA Scitech 2020 Forum*, p. 1457. DOI: 10.2514/6.2020-1457.

Patel, Manthan, Marco Karrer, Philipp Bänninger, and Margarita Chli (2023). “COVINS-G: A generic back-end for collaborative visual-inertial SLAM.” In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2076–2082. DOI: 10.1109/ICRA48891.2023.10160938.

Pätzold, Bastian, Simon Bultmann, and Sven Behnke (2022). “Online marker-free extrinsic camera calibration using person keypoint detections.” In: *44th DAGM German Conference on Pattern Recognition (GCPR)*, pp. 300–316. DOI: 10.1007/978-3-031-16788-1_19.

Pavlakos, Georgios, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. Osman, Dimitrios Tzionas, and Michael J. Black (2019). “Expressive body capture: 3D hands, face, and body from a single image.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10967–10977. DOI: 10.1109/CVPR.2019.01123.

Pavlakos, Georgios, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis (2017a). “6-DoF object pose from semantic keypoints.” In: *IEEE In-*

ternational Conference on Robotics and Automation (ICRA), pp. 2011–2018. DOI: 10.1109/ICRA.2017.7989233.

Pavlakos, Georgios, Xiaowei Zhou, Konstantinos G. Derpanis, and Kostas Daniilidis (2017b). "Harvesting multiple views for marker-less 3D human pose annotations." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1253–1262. DOI: 10.1109/CVPR.2017.138.

Peng, Sida, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao (2019). "PVNet: Pixel-wise voting network for 6DoF pose estimation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4556–4565. DOI: 10.1109/CVPR.2019.00469.

Periyasamy, Arul Selvam, Luis Denninger, and Sven Behnke (2022). "Learning implicit probability distribution functions for symmetric orientation estimation from RGB images without pose labels." In: *6th IEEE International Conference on Robotic Computing (IRC)*, pp. 221–228. DOI: 10.1109/IRC55401.2022.00044.

Piewak, Florian, Peter Pinggera, Manuel Schäfer, David Peter, Beate Schwarz, Nick Schneider, Markus Enzweiler, David Pfeiffer, and Marius Zöllner (2018). "Boosting LiDAR-based semantic labeling by cross-modal training data generation." In: *European Conference on Computer Vision (ECCV) Workshops*, pp. 497–513. DOI: 10.1007/978-3-030-11024-6_39.

Pizarro, Daniel, Manuel Mazo, Enrique Santiso, and Hideki Hashimoto (2008). "Mobile robot geometry initialization from single camera." In: *Field and Service Robotics (FSR)*, pp. 93–102. DOI: 10.1007/978-3-540-75404-6_9.

Pizarro, Daniel, Manuel Mazo, Enrique Santiso, Marta Marron, David Jimenez, Santiago Cobreces, and Cristina Losada (2010). "Localization of mobile robots using odometry and an external vision sensor." In: *Sensors* 10.4, pp. 3655–3680. DOI: 10.3390/s100403655.

Qi, Charles R., Hao Su, Kaichun Mo, and Leonidas J. Guibas (2017). "PointNet: Deep learning on point sets for 3D classification and segmentation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 77–85. DOI: 10.1109/CVPR.2017.16.

Qi, Charles R, Yin Zhou, Mahyar Najibi, Pei Sun, Khoa Vo, Boyang Deng, and Dragomir Anguelov (2021). "Offboard 3D object detection from point cloud sequences." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6134–6144. DOI: 10.1109/CVPR46437.2021.00607.

Qiu, Haibo, Chunyu Wang, Jingdong Wang, Naiyan Wang, and Wenjun Zeng (2019). "Cross view fusion for 3D human pose estimation." In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 4341–4350. DOI: 10.1109/ICCV.2019.00444.

Quenzel, Jan and Sven Behnke (2021). "Real-time multi-adaptive-resolution-surfel 6D LiDAR odometry using continuous-time trajectory optimization." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5499–5506. DOI: 10.1109/IROS51168.2021.9636763.

Quigley, Morgan, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng (2009). "ROS: An open-source robot operating system." In: *IEEE International Conference on Robotics and Automation (ICRA) Workshop on Open Source Software*. Vol. 3, p. 5.

Rad, Mahdi and Vincent Lepetit (2017). "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth." In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 3848–3856. DOI: 10.1109/ICCV.2017.413.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi (2016). "You Only Look Once: Unified, real-time object detection." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

Rehder, Joern, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart (2016). "Extending Kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4304–4311. DOI: 10.1109/ICRA.2016.7487628.

Reinke, Andrzej, Marco Camurri, and Claudio Semini (2019). "A factor graph approach to multi-camera extrinsic calibration on legged robots." In: *IEEE International Conference on Robotic Computing (IRC)*, pp. 391–394. DOI: 10.1109/IRC.2019.00071.

Rekleitis, Ioannis, Robert Sim, Gregory Dudek, and Evangelos Milios (2001). "Collaborative exploration for the construction of visual maps." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 3, pp. 1269–1274. DOI: 10.1109/IROS.2001.977157.

Remelli, Edoardo, Shangchen Han, Sina Honari, Pascal Fua, and Robert Wang (2020). "Lightweight multi-view 3D pose estimation through camera-disentangled representation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6039–6048. DOI: 10.1109/CVPR42600.2020.00608.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2017). "Faster R-CNN: Towards real-time object detection with region proposal networks." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6, pp. 1137–1149. DOI: 10.1109/TPAMI.2016.2577031.

Rosinol, Antoni, Andrew Violette, Marcus Abate, Nathan Hughes, Yun Chang, Jingnan Shi, Arjun Gupta, and Luca Carlone (2021). "Kimera: From SLAM to spatial perception with 3D dynamic scene graphs." In: *The International Journal of Robotics Research (IJRR)* 40.12-14, pp. 1510–1546. DOI: 10.1177/02783649211056674.

Rosu, Radu Alexandru, Jan Quenzel, and Sven Behnke (2020). "Semi-supervised semantic mapping through label propagation with semantic texture meshes." In: *International Journal of Computer Vision (IJCV)* 128.5, pp. 1220–1238. DOI: 10.1007/s11263-019-01187-z.

Rünz, Martin, Maud Buffier, and Lourdes Agapito (2018). "MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects." In: *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 10–20. DOI: 10.1109/ISMAR.2018.00024.

Rusu, Radu Bogdan and Steve Cousins (2011). "3D is here: Point Cloud Library (PCL)." In: *IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2011.5980567.

Sa, Inkyu, Zetao Chen, Marija Popović, Raghav Khanna, Frank Liebisch, Juan Nieto, and Roland Siegwart (2018). "weedNet: Dense semantic weed classification using multispectral images and MAV for smart farming." In: *IEEE Robotics and Automation Letters (RA-L)* 3.1, pp. 588–595. DOI: 10.1109/LRA.2017.2774979.

Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen (2018). "MobileNetV2: Inverted residuals and linear bottlenecks." In: *IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

Schleich, Daniel, Marius Beul, Jan Quenzel, and Sven Behnke (2021). "Autonomous flight in unknown GNSS-denied environments for disaster examination." In: *International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 950–957. DOI: 10.1109/ICUAS51884.2021.9476790.

Schmuck, Patrik and Margarita Chli (2019). "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams." In: *Journal of Field Robotics (JFR)* 36.4, pp. 763–781. DOI: 10.1002/rob.21854.

Schwarz, Max and Sven Behnke (2020). "Stillleben: Realistic scene synthesis for deep learning in robotics." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10502–10508. DOI: 10.1109/ICRA40945.2020.9197309.

Seshadri, Kiran, Berkin Akin, James Laudon, Ravi Narayanaswami, and Amir Yazdanbakhsh (2022). "An evaluation of Edge TPU accelerators for convolutional neural networks." In: *2022 IEEE International Symposium on Workload Characterization (IISWC)*. DOI: 10.1109/iiswc55918.2022.00017.

Severi, Stefano, J Härri, M Ulmschneider, B Denis, M Bartels, et al. (2018). "Beyond GNSS: Highly accurate localization for cooperative-intelligent transport systems." In: *IEEE Wireless Communications and Networking Conference (WCNC)*. DOI: 10.1109/WCNC.2018.8377457.

Shim, Jae-Hong and Young-Im Cho (2015). "A mobile robot localization using external surveillance cameras at indoor." In: *Procedia Computer Science* 56, pp. 502–507. DOI: 10.1016/j.procs.2015.07.242.

Shkurti, Florian, Wei-Di Chang, Peter Henderson, Md Jahidul Islam, Juan Camilo Gamboa Higuera, Jimmy Li, Travis Manderson, Anqi Xu, Gregory Dudek, and Junaed Sattar (2017). "Underwater multi-robot convoying using visual tracking by detection." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4189–4196. DOI: 10.1109/IROS.2017.8206280.

Strand, Leah, Jens Honer, and Alois Knoll (2022). "Systematic error source analysis of a real-world multi-camera traffic surveillance system." In: *25th International Conference on Information Fusion (FUSION)*.

Stückler, Jörg, Nenad Biresev, and Sven Behnke (2012). "Semantic mapping using object-class segmentation of RGB-D images." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3005–3010. DOI: 10.1109/IROS.2012.6385983.

Sturm, Jürgen, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers (2012). "A benchmark for the evaluation of RGB-D SLAM systems." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 573–580. DOI: 10.1109/IROS.2012.6385773.

Sun, Li, Zhi Yan, Anestis Zaganidis, Cheng Zhao, and Tom Duckett (2018). "Recurrent-OctoMap: Learning state-based map refinement for long-term semantic mapping with 3D-LiDAR data." In: *IEEE Robotics and Automation Letters (RA-L)* 3.4, pp. 3749–3756. DOI: 10.1109/LRA.2018.2856268.

Tan, Mingxing and Quoc Le (2019). "EfficientNet: Rethinking model scaling for convolutional neural networks." In: *International Conference on Machine Learning (ICML)*, pp. 6105–6114.

Tanke, Julian and Juergen Gall (2019). “Iterative greedy matching for 3D human pose tracking from multiple views.” In: *German Conference on Pattern Recognition (GCPR)*, pp. 537–550. DOI: 10.1007/978-3-030-33676-9_38.

Thrun, Sebastian, Dieter Fox, Wolfram Burgard, and Frank Dellaert (2001). “Robust Monte Carlo localization for mobile robots.” In: *Artificial Intelligence* 128.1-2, pp. 99–141. DOI: 10.1016/S0004-3702(01)00069-8.

Tian, Yulun, Yun Chang, Fernando Herrera Arias, Carlos Nieto-Granda, Jonathan P. How, and Luca Carlone (2022). “Kimera-Multi: robust, distributed, dense metric-semantic SLAM for multi-robot systems.” In: *IEEE Transactions on Robotics* 38.4, pp. 2022–2038. DOI: 10.1109/TRO.2021.3137751.

Tome, Denis, Matteo Toso, Lourdes Agapito, and Chris Russell (2018). “Rethinking pose in 3D: Multi-stage refinement and recovery for markerless motion capture.” In: *Intl. Conference on 3D Vision (3DV)*, pp. 474–483. DOI: 10.1109/3dv.2018.00061.

Truong, Anh Minh, Wilfried Philips, Junzhi Guan, Nikos Deligiannis, and Lusine Abrahamyan (2019). “Automatic extrinsic calibration of camera networks based on pedestrians.” In: *IEEE International Conference on Distributed Smart Cameras (ICDSC)*. DOI: 10.1145/3349801.3349802.

Tu, Hanyue, Chunyu Wang, and Wenjun Zeng (2020). “VoxelPose: Towards multi-camera 3D human pose estimation in wild environment.” In: *European Conference on Computer Vision (ECCV)*, pp. 197–212. DOI: 10.1007/978-3-030-58452-8_12.

Umeyama, S. (1991). “Least-squares estimation of transformation parameters between two point patterns.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.04, pp. 376–380. DOI: 10.1109/34.88573.

Vora, Sourabh, Alex H. Lang, Bassam Helou, and Oscar Beijbom (2020). “PointPainting: Sequential fusion for 3D object detection.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4603–4611. DOI: 10.1109/CVPR42600.2020.00466.

Vora, Suhani, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi S. M. Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth (2022). “NeSF: Neural semantic fields for generalizable semantic segmentation of 3D scenes.” In: *Transactions on Machine Learning Research*. ISSN: 2835-8856. URL: https://openreview.net/forum?id=ggPhsYCsm9.

Wang, Chen, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese (2019). “DenseFusion: 6D object pose estimation by iterative dense fusion.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3338–3347. DOI: 10.1109/CVPR.2019.00346.

Wang, Tao, Jianfeng Zhang, Yujun Cai, Shuicheng Yan, and Jiashi Feng (2021). “Direct multi-view multi-person 3D pose estimation.” In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 34, pp. 13153–13164. URL: https://proceedings.neurips.cc/paper/2021/file/6da9003b743b65f4c0ccd295cc484e57-Paper.pdf.

Wang, Wei, Yan Yan, Luming Zhang, Richang Hong, and Nicu Sebe (2016). “Collaborative sparse coding for multiview action recognition.” In: *IEEE MultiMedia* 23.4, pp. 80–87. DOI: 10.1109/MMUL.2016.69.

Wang, Zejie, Zhen Zhao, Zhao Jin, Zhengping Che, Jian Tang, Chaomin Shen, and Yaxin Peng (2021). “Multi-stage fusion for multi-class 3D lidar detection.” In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 3113–3121. DOI: 10.1109/ICCVW54120.2021.00347.

Whelan, Thomas, Stefan Leutenegger, R Salas-Moreno, Ben Glocker, and Andrew Davison (2015). "ElasticFusion: Dense SLAM without a pose graph." In: *Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2015.XI.001.

Wirtz, Stefan and Dietrich Paulus (2016). "Evaluation of established line segment distance functions." In: *Pattern Recognition and Image Analysis* 26, pp. 354–359. DOI: 10.1134/S1054661816020267.

Xiang, Yu, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox (2018). "PoseCNN: A convolutional neural network for 6D object pose estimation in Cluttered Scenes." In: *Robotics: Science and Systems XIV*. DOI: 10.15607/RSS.2018.XIV.019.

Xiao, Bin, Haiping Wu, and Yichen Wei (2018). "Simple baselines for human pose estimation and tracking." In: *European Conference on Computer Vision (ECCV)*, pp. 466–481. DOI: 10.1007/978-3-030-01231-1_29.

Xiong, Yunyang, Hanxiao Liu, Suyog Gupta, Berkin Akin, Gabriel Bender, Yongzhe Wang, Pieter-Jan Kindermans, Mingxing Tan, Vikas Singh, and Bo Chen (2021). "MobileDets: Searching for object detection architectures for mobile accelerators." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3825–3834. DOI: 10.1109/CVPR46437.2021.00382.

Xu, Binbin, Wenbin Li, Dimos Tzoumanikas, Michael Bloesch, Andrew Davison, and Stefan Leutenegger (2019). "MID-Fusion: Octree-based object-level multi-instance dynamic SLAM." In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5231–5237. DOI: 10.1109/ICRA.2019.8794371.

Xu, Chenfeng, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka (2020). "SqueezeSegV3: Spatially-adaptive convolution for efficient point-cloud segmentation." In: *European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-030-58604-1_1.

Xu, Danfei, Dragomir Anguelov, and Ashesh Jain (2018). "PointFusion: Deep sensor fusion for 3D bounding box estimation." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 244–253. DOI: 10.1109/CVPR.2018.00033.

Ye, Hang, Wentao Zhu, Chunyu Wang, Rujie Wu, and Yizhou Wang (2022). "Faster VoxelPose: Real-time 3D human pose estimation by orthographic projection." In: *European Conference on Computer Vision (ECCV)*, pp. 142–159. DOI: 10.1007/978-3-031-20068-7_9.

Yi, Li, Boqing Gong, and Thomas Funkhouser (2021). "Complete & Label: A domain adaptation approach to semantic segmentation of LiDAR point clouds." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15358–15368. DOI: 10.1109/CVPR46437.2021.01511.

Yue, Yufeng, Chunyang Zhao, Zhenyu Wu, Chule Yang, Yuanzhe Wang, and Danwei Wang (2020). "Collaborative semantic understanding and mapping framework for autonomous systems." In: *IEEE/ASME Transactions on Mechatronics* 26.2, pp. 978–989. DOI: 10.1109/TMECH.2020.3015054.

Zappel, Moritz, Simon Bultmann, and Sven Behnke (2021). "6D object pose estimation using keypoints and part affinity fields." In: *RoboCup International Symposium*, pp. 78–90. DOI: 10.1007/978-3-030-98682-7_7.

Zhang, Jinrui, Deyu Zhang, Xiaohui Xu, Fucheng Jia, Yunxin Liu, Xuanzhe Liu, Ju Ren, and Yaoxue Zhang (2020). "MobiPose: Real-time multi-person pose estimation on mobile devices." In: *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pp. 136–149. DOI: 10.1145/3384419.3430726.

Zhang, Letian, Lixing Chen, and Jie Xu (2021). “Autodidactic neurosurgeon: Collaborative deep inference for mobile edge intelligence via online learning.” In: *Proceedings of the Web Conference 2021*, pp. 3111–3123. DOI: 10.1145/3442381.3450051. URL: https://doi.org/10.1145/3442381.3450051.

Zhang, Letian and Jie Xu (2023). “E3Pose: Energy-efficient edge-assisted multi-camera system for multi-human 3D pose estimation.” In: *8th ACM/IEEE Conference on Internet of Things Design and Implementation (IoTDI)*, pp. 52–65. DOI: 10.1145/3576842.3582370.

Zhang, Pengyi, Yunxin Zhong, and Xiaoqiong Li (2019). “SlimYOLOv3: Narrower, faster and better for real-time UAV applications.” In: *IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 37–45. DOI: 10.1109/ICCVW.2019.00011.

Zhang, Zhengyou (2000). “A flexible new technique for camera calibration.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11, pp. 1330–1334. DOI: 10.1109/34.888718.

Zhao, Lin, Hui Zhou, Xinge Zhu, Xiao Song, Hongsheng Li, and Wenbing Tao (2023). “LIF-Seg: LiDAR and camera image fusion for 3D LiDAR semantic segmentation.” In: *IEEE Transactions on Multimedia*. DOI: 10.1109/TMM.2023.3277281.

Zhi, Shuaifeng, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison (2021). “In-place scene labelling and understanding with implicit scene representation.” In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 15818–15827. DOI: 10.1109/ICCV48922.2021.01554.

Zhou, Bolei, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba (2019). “Semantic understanding of scenes through the ADE20K dataset.” In: *International Journal of Computer Vision* 127.3, pp. 302–321. DOI: 10.1007/s11263-018-1140-0.

Zhu, Xinge, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin (2021). “Cylindrical and asymmetrical 3D convolution networks for LiDAR segmentation.” In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9939–9948. DOI: 10.1109/CVPR46437.2021.00981.