DARKO

Dynamic Agile Production Robots That Learn and Optimise Knowledge and Operations

H2020-ICT-2020-2 Grant agreement no: 101017274

DELIVERABLE 2.2
Initial report on perception in DARKO

Dissemination Level: PUBLIC

Due date: month 30 (June 2023)

Deliverable type: Report

Lead beneficiary: Robert Bosch GmbH (Bosch)

Contents

1 Introduction **4**
1.1 Scope of this deliverable 4
1.2 Relation to other work packages 4
1.3 Key highlights and improvements over prior work 5
1.4 Structure of this deliverable 6

2 T2.1: Object-level semantics **6**
2.1 DARKO in-the-wild and benchmark datasets 7
2.2 RGB-D methods for object-level semantics 12
2.3 Towards more data-efficient learning of 3D object-level semantics 17
2.4 Applications to downstream tasks & further semantic scene representations 23
2.5 Remaining challenges and outlook 25

3 T2.2: Perception for manipulation **28**
3.1 A novel method for object pose estimation 29
3.2 A novel method for grasp pose estimation 32
3.3 Extended testing with the proposed grasp pose estimation method 34
3.4 Parameterized manipulation primitives for flat object grasping 35
3.5 Deployment and real-world demonstrations of grasp pose estimation 39
3.6 Remaining challenges and outlook 40

4 T2.3: In-hand grasp perception **41**
4.1 Object pose estimation for novel objects with slight occlusion 41
4.2 Learning to prevent grasp failure with soft hands: from on-line prediction to dual-arm grasp recovery 43
4.3 Simultaneous perception and manipulation 47
4.4 Remaining challenges and outlook 50

5 T2.4: Detecting successful throws **51**
5.1 A trajectory-centric approach to detection of throw success 51
5.2 Experiments and results on detection of throw success 52
5.3 Summary of achievements and outlook 53

6 T2.5: Perception of humans and their poses **54**
6.1 A cross-modal comparison of human detection approaches for industrial robotics applications 56
6.2 Efficient deployment of human detection on embedded hardware 58
6.3 A novel multi-modal multi-view dataset for 3D human pose estimation . . . 61
6.4 Methods for wide-FOV 3D human pose estimation 75
6.5 Temporal filtering of articulated 3D human poses 75
6.6 Human activity recognition 77
6.7 Summary of achievements and outlook 78

7 Conclusion **81**

8 References **82**

List of Abbreviations

Abbreviation	Meaning
9-DoF	9 degrees of freedom (3D position, 3D orientation, 3D extents)
API	Application Programming Interface, the public interface provided by a library for use by software developers
ARENA 2036	A large research campus in form of a modern factory hall in Stuttgart-Vaihingen, Germany. Provides an innovation platform for mobility & production of the future and hosts DARKO project demonstrations.
CAD	Computer Aided Design
CNN	Convolutional Neural Network
DLA	Dynamic Layer Aggregation backbone
CUDA	Compute Unified Device Architecture for hardware acceleration of neural network computations on Nvidia GPUs
DeTR	Detection Transformer, transformer-based NN for object detection
FOV	Field of view of a sensor
GigE	Gigabit Ethernet
GPU	Graphics Processing Unit, used as a deep learning accelerator to train and run inference on neural networks
GRU	Gated Recurrent Unit, a gating mechanism in RNNs
ILIAD	EU Horizon 2020 project (2016–2020) which deployed a heterogeneous fleet of mobile service robots in intralogistics environments.
KLT	Kleinladungsträger, a standardized plastic box in different sizes often used in intralogistics.
LiDAR	Light Detection And Ranging, a time-of-flight-based sensor that produces point clouds. Also spelled “lidar”.
NN	Neural Network
NIC	Network Interface Controller
ONNX	Open Neural Network Exchange
ORU	Örebro University, member of the DARKO consortium
PTP	Precision Time Protocol, a more accurate clock synchronization protocol than Network Time Protocol (NTP)
RGB	Red, Green, Blue
RGB-D	Red, Green, Blue, Depth
RNN	Recurrent Neural network
ROS	Robot Operating System, see www.ros.org
SDK	Software Development Kit
SLAM	Simultaneous Localization and Mapping
SPENCER	EU FP7 project (2013–2016) which deployed a mildly humanized service robot in a busy airport terminal at Amsterdam Schiphol Airport.
SVM	Support Vector Machine, a machine learning classifier
ToE	Trigger-over-Ethernet, method for camera triggering
UNIPI	Università di Pisa, member of the DARKO consortium
VRAM	Video memory of a GPU that can be used to deploy neural networks
WP	Work package
YOLO	A series of 2D object detectors developed by J. Redmon

1 Introduction

This deliverable reports on the initial perception system developed in the EU H2020 project DARKO and covers the entire work package WP2: 3D Perception and Scene Understanding with all its five sub-tasks.

In contrast to the preceding *software* deliverable D2.1, which focused only on the broader-level scene understanding tasks T2.1 and T2.5, this detailed report also covers perception for manipulation-related tasks (T2.2, T2.3 and T2.4). Furthermore, this deliverable introduces novel methodologies that significantly go beyond the existing state-of-the-art in various aspects. It provides deeper theoretical insights as well as in-depth experimental results related to the research done by Bosch, Örebro University, Pisa, EPFL and University of Lincoln within this work package.

1.1 Scope of this deliverable

This deliverable reports on the accumulated research and development efforts and results obtained in the following tasks during periods 1 and 2 of the DARKO project:

- Task T2.1: Object-level semantics
- Task T2.2: Perception for manipulation
- Task T2.3: In-hand grasp perception
- Task T2.4: Detecting successful throws
- Task T2.5: Perception of humans and their poses

Project partners Robert Bosch GmbH (BOSCH, lead responsible), Örebro University (ORU), Università di Pisa (UNIPI), École polytechnique fédérale de Lausanne (EPFL), University of Lincoln (UoL), and Technical University of Munich (TUM) contributed to this deliverable.

1.2 Relation to other work packages

Figure 1 illustrates the relation of work package WP2, which this deliverable reports on, to the other technical work packages in DARKO. Either through direct communication, or indirectly through the mapping and localization work package (WP3), WP2 provides essential cues to dynamic manipulation (WP4), human-robot spatial interaction (WP5), motion planning for the mobile base (WP6) and risk management and scheduling (WP7).

With perception being located at the beginning at the *sense – plan – act* chain, the functionalities presented in this deliverable provide functionalities that are crucial for the DARKO robot platform to successfully achieve its overall objectives, by enabling the robot to robustly perceive and understand its 3D surroundings in an intralogistics and agile production environment of the factory of the future.

At the technical level, all inter-component communication in DARKO happens via the Robot Operating System (ROS). More details on the communication between different DARKO components at a task level, including the used ROS message types, can be found in the system architecture deliverable D8.2.

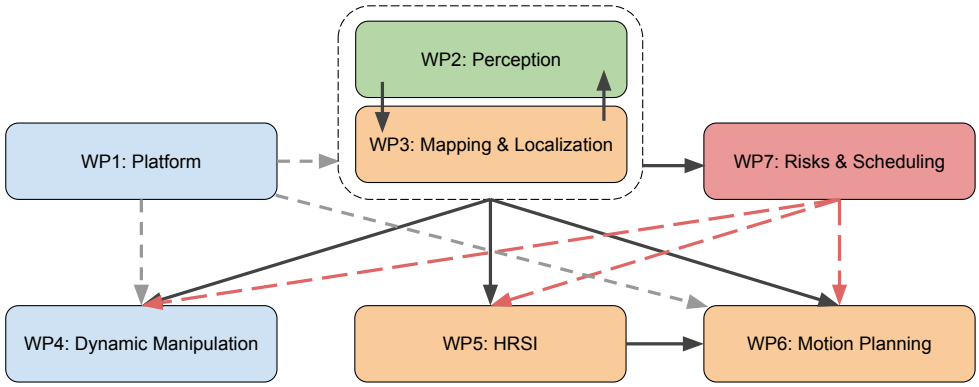

Figure 1: Relation of WP2, which this deliverable reports on, to other work packages in DARKO. WP2 is shown in green on top. It provides input both to work packages related to mobile manipulation (blue), as well as navigation in shared environments (yellow); here it is closely intertwined with WP3 (mapping and localization), which temporally integrates static and dynamic observations from tasks T2.1 and T2.5. Black arrows denote data flow during operation, dashed red arrows indicate constraints and orchestration, and dashed grey arrows indicate hardware dependencies.

1.3 Key highlights and improvements over prior work

In the first two periods of DARKO, the consortium has come up with a number of novel methods, experiments and datasets that directly address key objectives of DARKO. Combined together, they have led to the initial DARKO perception system presented in this deliverable. Key highlights and improvements over prior work include:

- Novel, comprehensive domain-specific datasets and efficient annotation strategies e. g. for 3D object detection and wide-FOV 3D human pose estimation targeted specifically at DARKO use-cases in intralogistics scenarios, which are so far underrepresented in publicly available datasets.
- Novel data-efficient methods for object-level semantics and 3D scene graph prediction, along with an extensive evaluation on DARKO-specific datasets and integration with downstream applications like semantic SLAM and semantic-aware navigation.
- Data-efficient learning-based methods for skeleton-based human activity recognition; for full-body 3D pose prediction with the aim of bridging temporary occlusions; and an extensive cross-modal comparison of human detection methods in intralogistics.
- Novel methods for perception for manipulation based upon direct grasp pose estimation and a learning-based combination with manipulation primitives, with real-world experiments using authentic objects from the DARKO target use-case.
- Novel approaches for in-hand perception to estimate in-hand object pose, to predict grasp failures with soft robotic hands and avoid object slippage, and to generate optimal manipulation patterns to collect information about grasped object's inertia.
- A trajectory-centric perception approach that detects whether grasped objects have successfully been thrown into target trays.
- Integration with downstream components, robot deployment and real-time optimization for runtime speed and memory consumption of several of the components described in this deliverable. Most components have been demonstrated live during the milestone MS2 demonstration at ARENA 2036 and the Automatica 2023 fair.

1.4 Structure of this deliverable

In the following sections, we discuss in detail the results obtained by the five tasks of work package WP2, starting with T2.1 and ending with T2.5. In each task-specific section, we at the end also outline remaining unsolved changes that may still be addressed in the final project period.

2 T2.1: Object-level semantics

Figure 2: A typical warehouse environment in DARKO's lead use-case with uncommon object types, cluttered and dynamic scenes illustrating the challenges for object-level semantics.

The goal of task T2.1 is to develop a module providing real-time semantic and spatial information of the surroundings using robot's on-board sensors while considering the DARKO objective O3 on efficient deployment. This goal encompasses several challenges and research questions in the context of DARKO's target scenario (Figure 2): i) How to reduce the manual labeling effort to enable efficient deployment in environments underrepresented in large-scale public data sets?, ii) How to leverage multi-modal and multi-view data from the robot's sensors to obtain more robust semantics?, iii) Which deep neural network architecture to use under data and hardware constraints present in DARKO?, iv) Which semantic representations are useful for the downstream robot tasks? In the following sections we discuss the progress made since the beginning of the project towards answering these questions.

Key contributions in T2.1

Our contributions for task T2.1, discussed in the next subsections, include:

- Several data recording campaigns in a warehouse from our target use-case and in the Bosch robotics lab with a replica of the DARKO scenario, using a hand-held sensor suite resembling the DARKO robot's sensor setup.
- Development of an efficient 3D labeling strategy for static objects using SLAM to enable training of 3D object detectors with reduced manual annotation effort.
- Creation of an annotated 3D intralogistics dataset using the developed efficient labeling strategy for training and benchmarking of 9DoF 3D object detectors in scenarios relevant to DARKO use-cases.
- Extension of the RGB-D YOLO [14] deep neural network to support estimation of multiple keypoints, multiple object classes, 9DoF object bounding box estimation and RGB + 3D lidar input data.
- Deployment and integration of the object-level semantics module for real-time inference on the DARKO robot.

- Extension of modern 3D object detectors for 9DoF object detection including their training and evaluation on the DARKO intralogistics dataset with 18 object classes.
- Research on self-supervised learning for label-efficient 3D object detection, and for prediction of 3D scene graphs – a higher-level scene representation which models relationships between objects.
- Research on alternative approaches for object-level semantics, such as a 2D-to-3D uplifting approach, which can potentially generalize better to a fisheye+lidar setup.
- Research on leveraging mid-level visual representations for more efficient semantics-aware navigation in collaboration with WP6.

2.1 DARKO in-the-wild and benchmark datasets

To start the development of the DARKO perception components, we conducted several data recordings with a hand-held device equipped with an Azure Kinect RGB-D camera, an Ouster 3D lidar, and 2x Basler wide-angle cameras during the initial phase of the project. This sensor set closely resembles the sensor configuration on the DARKO robot platform. The portable sensor box is shwon in Figure 3. It includes sensors with different sensing modalities and characteristics. The Azure Kinect RGB-D camera provides color and dense depth information at a close range, but is limited in its field of view, the Ouster 3D lidar provides precise, but sparse, 360-degree point cloud data with a range exceeding 100m. The Basler cameras with their fisheye lenses capture a larger portion of the scene, but they suffer from strong image distortions. The multi-sensor setup is highly valuable for comparing and combining different sensor modalities and understanding their respective strengths and weaknesses within the challenging DARKO use-case.

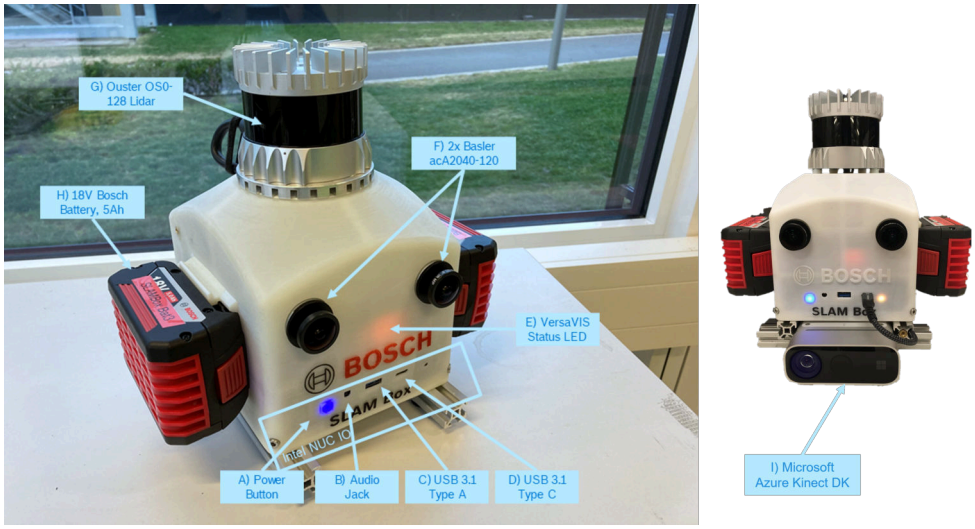

Figure 3: Portable sensor box by Bosch, utilizing an Ouster OS0-128 3D LiDAR and two Basler acA2040-120uc color cameras in stereo vision configuration. The sensor box can be optionally equipped with an MS Azure Kinect as shown on the right to further extend sensing capabilities.

2.1.1 Warehouse data recordings

To better understand the domain-specific challenges, we kicked off the project with a data collection campaign at a warehouse that represents the project's lead use-case. Using our

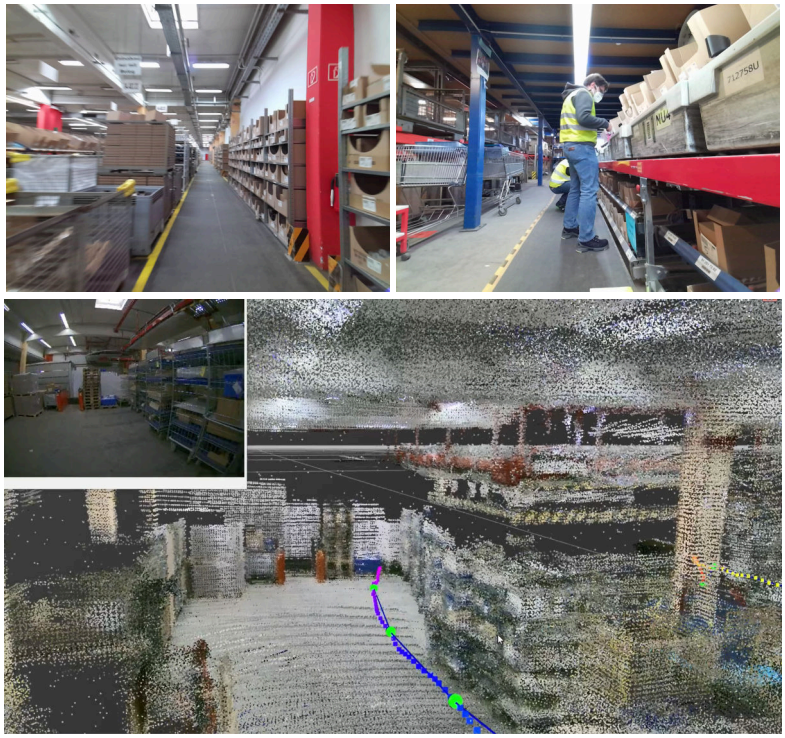

Figure 4: Warehouse from the DARKO lead use-case observed from different perspectives and sensors. The top left image provides a view from a hand-held MS Azure Kinect DK exhibiting significant motion blur. A view from the Azure Kinect placed on a cart at a lower height is shown in the top right corner. Such low placement of the sensor limits visibility of relevant objects, e.g. plastic trays on the conveyor, suggesting that the sensor should be mounted higher on the robot. The bottom picture includes a significantly distorted view from a fisheye camera as well as a 3D point cloud reconstruction from lidar and camera images of a mapping run in a cluttered scene. All these examples illustrate the challenges for broader-level 3D scene understanding in DARKO.

portable sensor box, we recorded over 2 TB of data, capturing various areas and halls of the warehouse, as well as human volunteers performing routine activities. Figure 4 showcases the impressions obtained from the ego-centric perspective of different sensors.

These recorded sequences allowed us to gain a deeper understanding of the requirements and challenges of the DARKO use-case for broader-level 3D scene understanding. Additionally, they have provided us with valuable insights for optimizing the sensor placement on the robot. Towards the end of the DARKO project, this data will allow us to assess the WP2 perception modules in realistic and highly challenging in-the-wild settings.

Bosch is considering to partially label these sequences for quantitative evaluation. However, the required effort needs to be first assessed, since due to the highly cluttered scenery with a lot of occlusions it might require resources beyond what is available in DARKO. Nevertheless, even without annotations, this dataset is already proving to be very valuable for qualitative assessment of the methods developed in DARKO.

As the dataset includes proprietary and sensitive information from an industrial facility, it is not planned to make it publicly available.

2.1.2 DARKO intralogistics dataset: a novel dataset for 9DoF 3D object detection in intralogistics scenarios

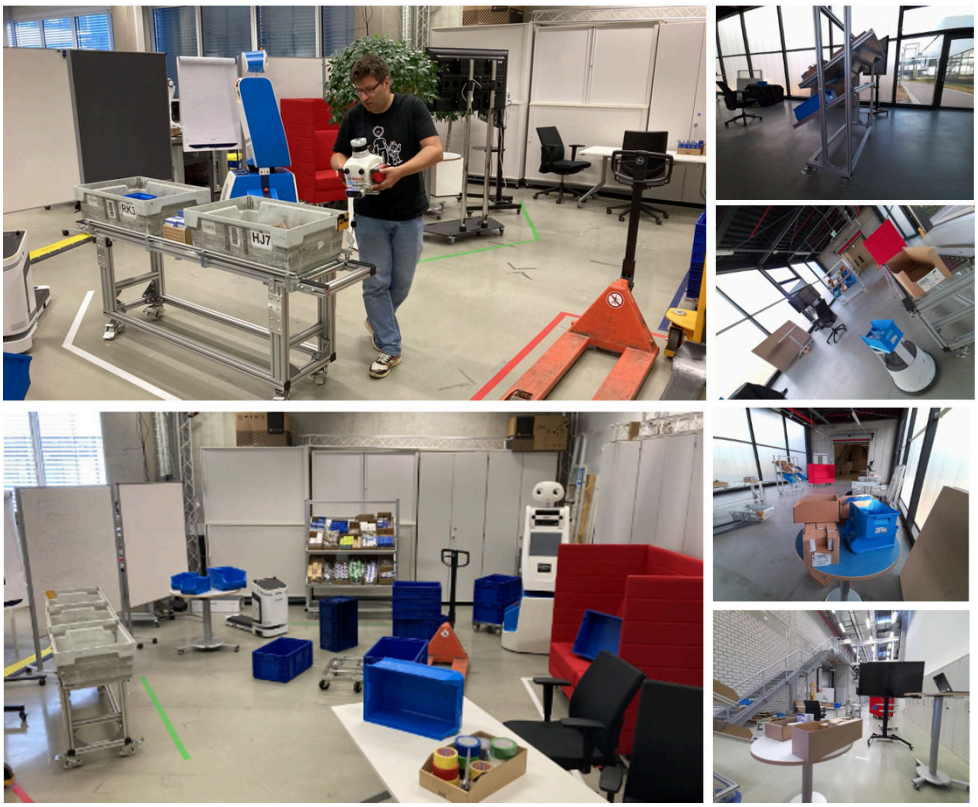

Figure 5: Recording of a comprehensive intralogistics dataset for 9DoF 3D object detection, capturing diverse environments, backgrounds, and lighting conditions. Objects were deliberately arranged in various configurations including stacked KLTs, tightly placed trays on the conveyor and tilted storage bins in the shelf. The objects were observed from multiple angles and distances using a portable sensor box.

DARKO's target scenario is characterized by objects usually not present in publicly available 3D scene understanding benchmarks. Therefore, to facilitate the development, the evaluation and to enable training of the object-level semantics component on DARKO-relevant objects, we recorded additional data sequences with the same sensor setup in a more controlled setting at the Bosch robotics lab. The recorded sequences contain objects relevant to intralogistics (e.g. storage boxes, dolleys, pallet trucks, etc.) and include a shelf and conveyor replica built specifically for DARKO. Impressions from the data collection and a few example scenes can be seen in Figure 5.

The recording process involved capturing data in multiple sessions, where different object types, arrangements and backgrounds were utilized. The first session focused on a single *KLT (Kleinladungsträger)* object category – a standard transportation and storage box used in industrial environments. The number of categories was extended to 8 in the second session and included further objects such as a pallet truck, a pallet stacker and a dolley. In the third and final recording session, we further increased the complexity of the dataset by introducing authentic DARKO use-case objects as well as the DARKO replica shelf and conveyor, increasing the number of object categories to 18. A complete list of

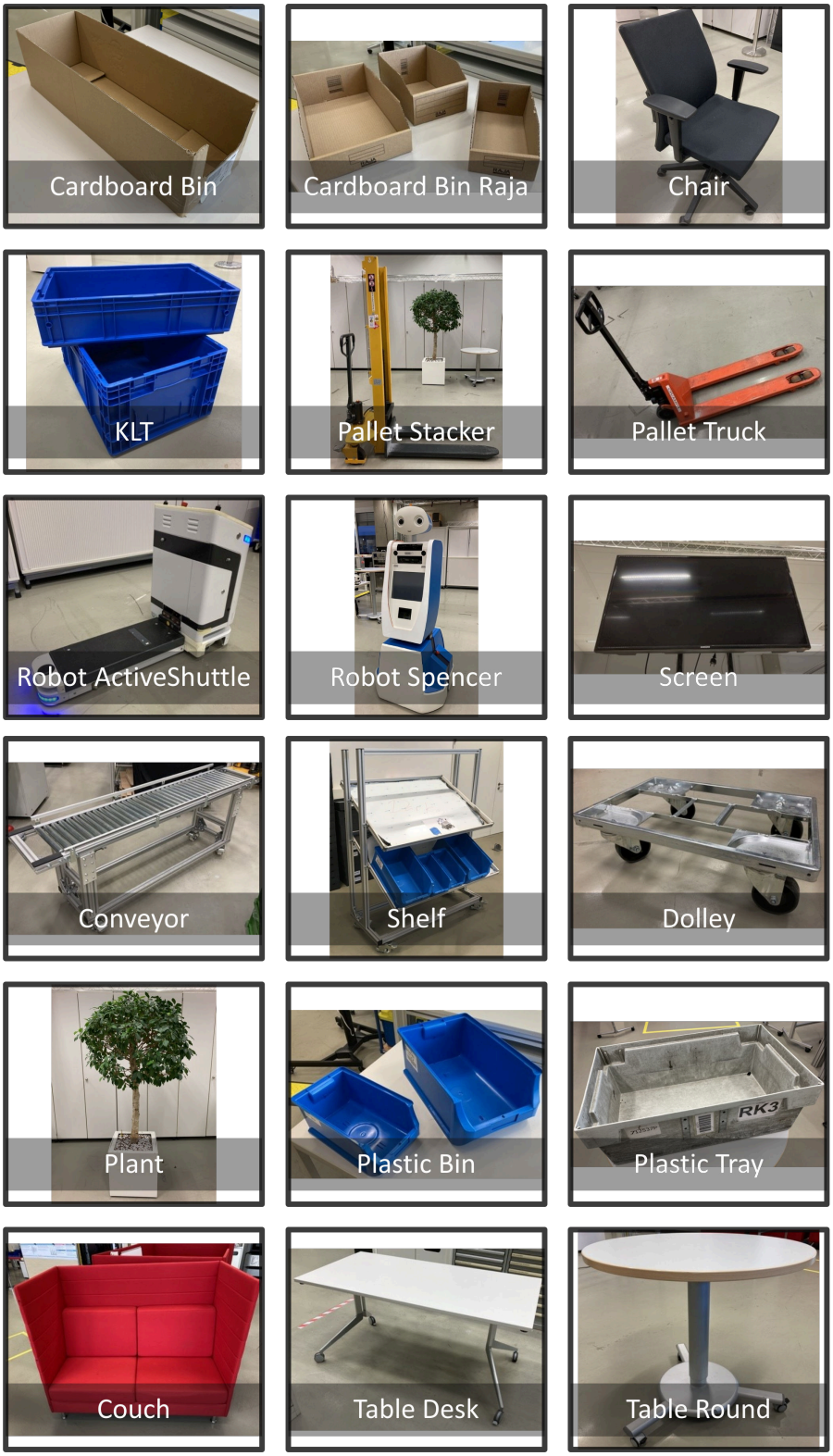

Figure 6: Objects from the novel DARKO intralogistics dataset covering different object categories varying in shape, material and size. Most of these objects are rarely encountered in publicly available benchmarks, but are common in industrial environments. The cardboard bins and the plastic tray are the authentic objects from the target use-case.

Object Category	Rotational Symmetry Type	Axis	Angles
KLT	discrete	**z**	180°
Dolley	discrete	**z**	180°
Conveyor	discrete	**z**	180°
Desk	discrete	**z**	180°
Plant	discrete	**z**	90°, 180°, 270°
Round Table	continuous	**z**	-

Table 1: List of object categories from the intralogistics dataset with rotational symmetries.

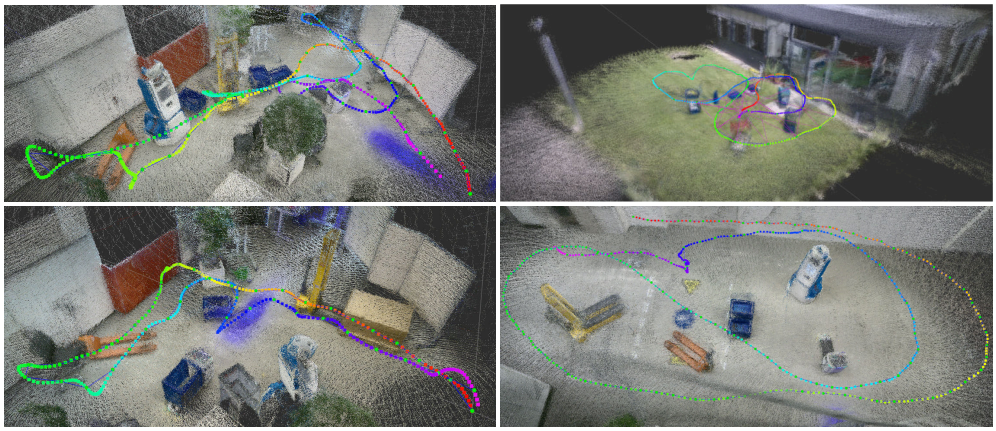

Figure 7: Examples of reconstructed scenes from the DARKO intralogistics dataset, visualized as colored point clouds along with the tracked trajectory of the portable sensor box.

object categories and example object instances are provided in Figure 6. Notably, some of these object categories exhibit rotational symmetries, as outlined in Table 1. The presence of symmetric objects poses additional challenges for 3D object detectors due to ambiguous poses with respect to the sensor.

In total, we recorded more than 50 sequences, each lasting from 1 to 5 minutes. These sequences encompass a wide range of object placements, backgrounds, and lighting conditions, ensuring diversity in the dataset. We deliberately varied the scene complexity, capturing scenarios that ranged from well-separated objects to realistic configurations like stacked KLTs, KLTs placed on dolleys, cluttered shelves, tightly packed storage bins, trays, and more. This comprehensive approach allowed us to capture a rich and varied dataset that accurately represents the intricacies of the DARKO use-case. Figure 7 showcases a few examples of the trajectories of the hand-held sensor box across different scenes from the intralogistics dataset.

In parallel to the dataset collection efforts, we have actively been developing 3D annotation and visualization tools. These tools are designed to support the annotation process and facilitate the analysis of the collected data. Additionally, we implemented a label-efficient annotation strategy based on simultaneous localization and mapping (SLAM) and used it to efficiently annotate multiple sequences with 9DoF (3DoF orientation, positiion, and extents) bounding boxes for all 18 categories. We will discuss this multi-view labelling approach in more detail in Section 2.3.1.

The final dataset consists of annotated RGB-D frames composed of time-synchronized RGB images and 3D lidar scans. The lidar point clouds were first projected to the RGB camera frame and points within the field of view of the camera were used to create a sparse depth map. A few examples of annotated bounding boxes are shown in Figure 8. In total,

	Train	Test
Sequences	33	8
RGB-D Frames	25,827	7,255
Annotations	327,265	109,614
Categories	18	18

Table 2: Statistics of the DARKO intralogistics dataset

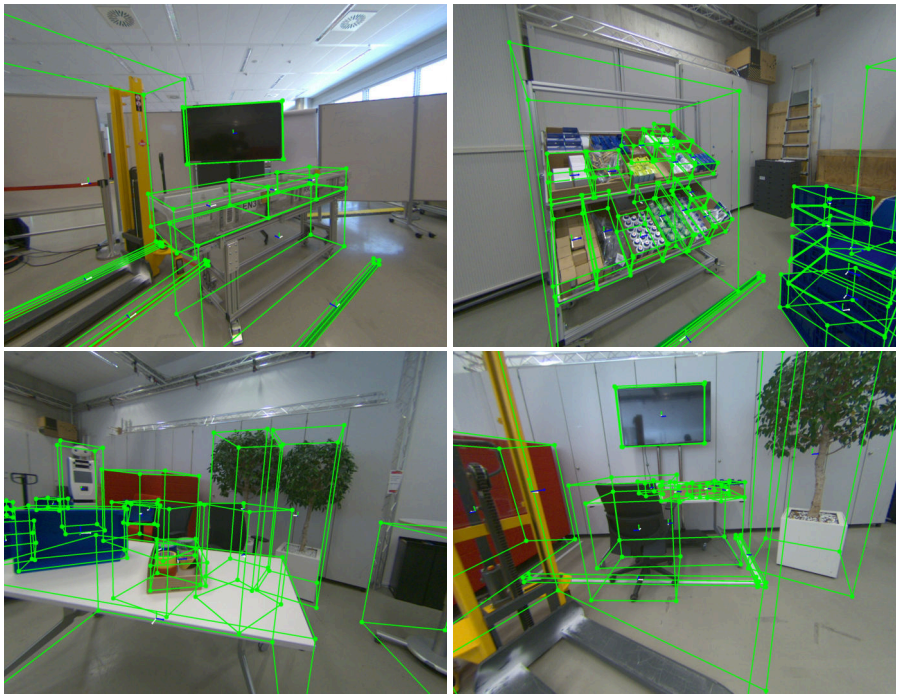

Figure 8: Multiple annotated scenes from DARKO intralogistics dataset. Notably, our labeling appraoch is able to generate amodal 3D bounding boxes. They are useful to assess object detectors' robustness to occlusions, as they provide a comprehensive representation of the object's full extent, even when partially occluded.

we annotated 33 sequences for the train split and 8 for the test split containing respectively ca. 26k and 7k RGB-D frames, leading to ca. 400k auto-labelled object instances in total. Each annotation includes a category label, 2D/3D instance segmentation masks and 2D/3D bounding boxes. For a comprehensive overview of the dataset characteristics, please refer to Table 2. Figure 9 provides category distributions for the test and train splits.

In the final project period, we will assess whether we can release this dataset or parts of it to the public for research purposes, e. g. as part of an upcoming publication.

2.2 RGB-D methods for object-level semantics

During the initial phase of the project, the DARKO consortium made the decision to utilize object bounding boxes with 9 degrees of freedom (3DoF centroid + 3DoF orientation + 3DoF extents) as the primary means of representing object-level semantics. Although this 9DoF bounding box representation may not provide a highly detailed description of object shapes and appearance, it still offers a concise representation that can be beneficial for various downstream tasks relying on object-level semantics. Additionally, it enables the

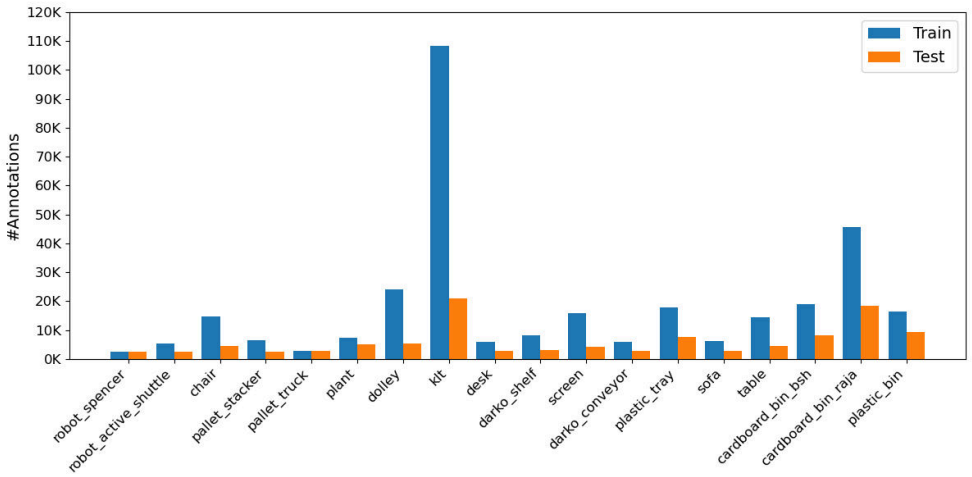

Figure 9: Class distribution for train and test splits in the DARKO intralogistics dataset. The initial dataset recordings and experiments focused on KLTs, resulting in this dominance as the most prevalent category. Therefore, in the future we might consider to remove some of the KLT-focused sequences to reduce class imbalance in the dataset.

use of more efficient deep neural network architectures and reduces the effort required for labeling during training compared to e. g. pixel- or point-wise instance segmentation masks. It is worth mentioning that while 9DoF bounding boxes were selected as the initial representation, the consortium remains open to investigating and integrating alternative representations as the project advances.

In the next section we describe a novel RGB-D method for 9DoF 3D object detection which we developed at the beginning of the project. We then proceed to discuss benchmark results on the DARKO intralogistics dataset in Section 2.2.2, where we compare additional methods with more recent neural architectures than our initial approach.

2.2.1 A novel RGB-D method for efficient 9DoF object detection

Our initial method for providing object-level semantics, called *9DoF RGB-D YOLO++*, is based on the RGB-D YOLO deep neural network architecture originally developed for human detection [14] during the EU H2020 project ILIAD. The perception pipeline, including the extended network architecture, is illustrated in Figure 10. Further implementation and software details can be found in DARKO deliverable D2.1. Here, we provide a brief overview of the extensions introduced during the DARKO project.

RGB+Lidar support: The original method from [14] was developed for dense RGB-D images. The DARKO robot and our portable sensor box used for data recordings are equipped with lidar and fisheye cameras to cover a larger field of view than can be obtained with an MS Azure Kinect RGB-D camera. Therefore, we extended RGB-D YOLO to support RGB+Lidar input data. To that end, we first had to extrinsically calibrate the sensors and synchronize their data streams. Given a time-synchronized pair of RGB image and a 3D lidar scan, our pipeline projects the points into the image plane using a z-buffer to account for occlusions that arise due to the displacement of the sensors. The projected points which fall within the RGB sensor field of view form a sparse dense map, which is densified in the following step using morphological image operations [15]. The densification is optional, but our experiments showed that it can improve the performance

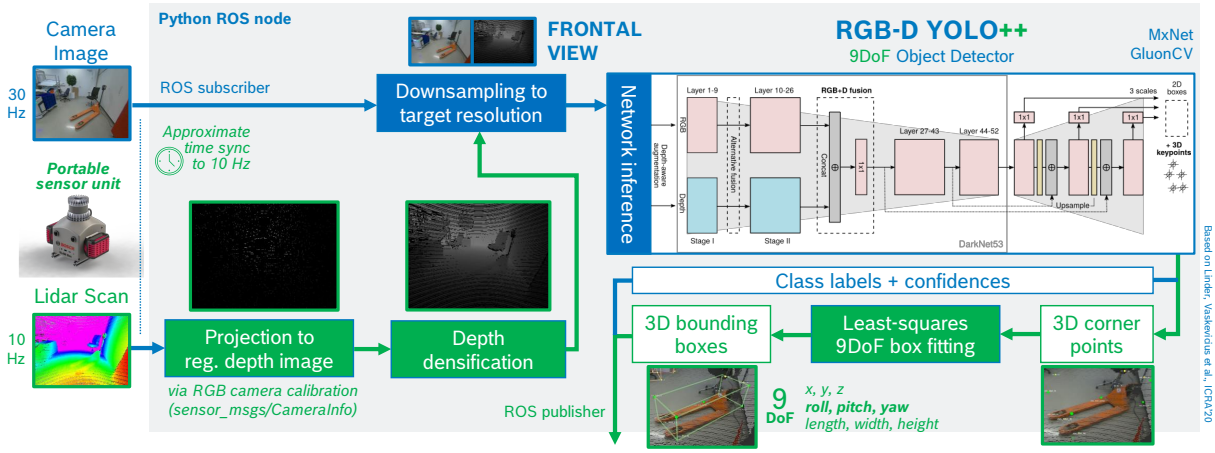

Figure 10: The extended 9DoF RGB-D YOLO++ detector for 3D object detection in T2.1. Highlighted in green are new contributions developed in DARKO that extend our prior work.

of the network significantly. Therefore, we utilize it during both training and inference.

Extensions to the architecture: We have extended the RGB-D YOLO method from a simple 3D centroid regression to the estimation of k 3D keypoints. The number of keypoints can be chosen based on the downstream application and the keypoints can have an arbitrary meaning e.g. they could represent human body joints or particular points of interest on an object. In the case of 9DoF 3D object detection, we chose them to represent ordered corner points of an oriented 3D bounding box of an object. In addition, we extended [14] to support multi-class training and inference.

Least-squares 9DoF box fitting: While predicted corner keypoints have a semantic meaning, e.g. of being the top-left-front or the bottom-left-rear corner, they do not necessarily represent an exact 9DoF box (i. e. a cuboid) due to the noise in the estimated 3D coordinates. To address this problem, we add a refinement step. We first estimate the 3DoF scale of the bounding box by computing mean extents from the predicted corner keypoints[1]. Given the estimated scale, we can formulate the refinement step as least-squares fitting of two 3D point sets – the noisy predicted keypoints, and the corners of an ideal box with the estimated scale. There are multiple ways of solving this task. We use an approach based on singular value decomposition (SVD) similar to [16].

In Figure 11 we provide a few examples of *9DoF RGB-D YOLO++* predictions on the DARKO intralogistics dataset and an example of the predictions from the integrated module running on the DARKO robot deployed at ARENA 2036 in Stuttgart during the milestone MS2 demonstration. As described later in T2.5 and in deliverable D2.1, the object-level semantics module is part of a reusable, modular Docker-based software stack and has been optimized for real-time performance on the DARKO robot using the Microsoft ONNX Runtime and Nvidia's TensorRT backend.

[1]Note that there are 4 estimates per x, y, z extent from in total 8 corner points.

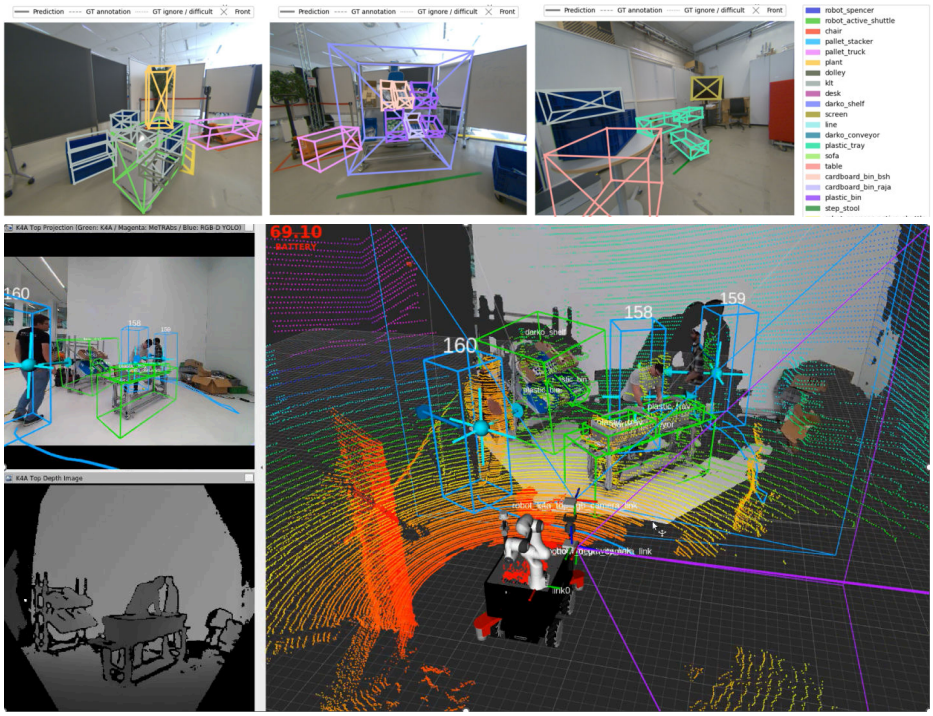

Figure 11: Example detections by *9DoF RGB-D YOLO++* on the DARKO intralogistics dataset are shown on the top. The bottom image provides an example of 3D object detections (green boxes) from a model deployed on the DARKO platform at ARENA 2036 in Stuttgart.

2.2.2 Intralogistics benchmark for 9DoF object detection

In the second period of the project, we also investigated more recent neural architectures for the task of 9DoF 3D object detection than the YOLO v3-based RGB-D YOLO method. We looked into 3D object detection methods operating on RGB-only, Pointcloud-only, RGB-D and RGB + Pointcloud input data. The majority of these methods, with an exception of Cube R-CNN [17], predict only a yaw angle in addition to 3D scale and position (i. e. 7DoF bounding boxes). Therefore, we extended these method to predict a full 3DoF orientation. Below, we give a brief overview of the methods chosen for benchmarking.

VoteNet [18] is a point cloud-based method, which takes raw point cloud input and processes it using a PointNet++ [19] backbone to output a set of seed points and seed features which in turn are used to generate vote points that approximate potential object centers. Features around these vote points are aggregated to later regress the objectness score, semantic score, object pose (i.e. center location, yaw angle), and size (extents). As mentioned earlier, in the original VoteNet, only the yaw angle is estimated (7DoF bounding box). It is computed in a bin-based classification and regression fashion where the entire 360° angle is divided into 12 bins/classes (trained with classification loss) and the residual angle within each bin is regressed (trained with regression loss). We extended this angle regression to estimate roll and pitch angles in a similar fashion.

ImVoteNet [20] is a VoteNet-based RGB + point cloud fusion method. The point cloud is processed by vanilla VoteNet, while the RGB image is processed by an image backbone, which is usually fine-tuned on the current dataset for 2D object detection and frozen

thereafter. During the training of ImVoteNet, the RGB image is passed through the frozen RGB backbone to generate 2D box proposals. Features from these 2D proposals (involving geometric, texture, and semantic cues) are fused with corresponding seed point features computed by the VoteNet branch. Then, the original VoteNet pipeline is followed to estimate category, pose and size of objects. In our experiments, we use our extended 9DoF version of VoteNet as described above. For the image branch we use Faster R-CNN with ResNet-50 backbone.

DeMF [21], short for deformable attention-based multimodal fusion, is a recent strategy proposed for combining RGB and point cloud data. It is based on a modified version of the deformable attention module, originally designed for faster computations in dense vision transformers. In this strategy, the RGB image and point cloud data are first processed separately using dedicated backbones. However, before regressing the pose and size parameters, the features of the seed points are combined with features from the image backbone using the proposed DeMF module. For the image branch, we use the frozen encoder of Deformable DETR trained for 2D object detection on the current dataset. For the point cloud branch we use our extended 9DoF VoteNet.

Cube R-CNN [17] is a recently proposed RGB-only method for 9DoF 3D object detection. We directly use the method as one of the baselines. Cube R-CNN takes an RGB image as input and processes it with a DLA34 [22] backbone followed by a Region Proposal Network (RPN) head which outputs several Regions of Interest (RoI). Each RoI is expected to contain an object of interest. Further, these RoIs are passed through a 2D Head which outputs class probability scores and 2D bounding box estimates for a potential object in an RoI. Until this point, the above pipeline is similar to that of a Faster R-CNN 2D object detector. Cube R-CNN introduces an additional head (Cube head) which takes in an RoI and estimates the full 9DoF pose and size of the potential object in 3D.

Cube R-CNN (RGB-D) is a custom version of Cube R-CNN thas has been extended for RGB-D data and is being developed in an academic collaboration of Bosch outside of the DARKO project. The current version of the method is included in the benchmark to evaluate its future potential for DARKO-related use cases.

We have also investigated **CenterPose [23]**, a method for category-level pose estimation from RGB images. As usual for categorical pose estimation methods, CenterPose supports only a single class per trained model. Our attempts to extend the method for the multi-class 9DoF object detection task however led to poor performance. Therefore, we omit this baseline in our benchmark.

For the evaluation of the above listed methods including *9DoF RGB-D YOLO++* on the test split of the DARKO intralogistics dataset, we extended the usual mean average precision (mAP) metric used for 3D object detection to support 9DoF bounding boxes within the 3D IoU (intersection over union) computation[2]. The results of our benchmark for two different IoU thresholds (0.25 and 0.5) are shown in Figure 12. We can observe that almost all of the methods improve upon *RGB-D YOLO*, our initial model for object-level semantics in DARKO. The best performing method on mAP@25 metric is the extended version of Cube R-CNN (RGB-D). However, all point cloud-based methods have significantly better performance on the mAP@50 metric indicating that they produce more accurate localization estimates. Further analysis suggests that point cloud-based methods underperform on tightly placed

[2]Commonly used mAP metrics for 3D object detection assume that bounding boxes are aligned with the ground plane and consider only yaw angle in the IoU computation.

smaller objects such as plastic and cardboard bins in the shelf. In Figure 13 we show a few examples of Cube R-CNN (RGB-D) predictions on the test scenes.

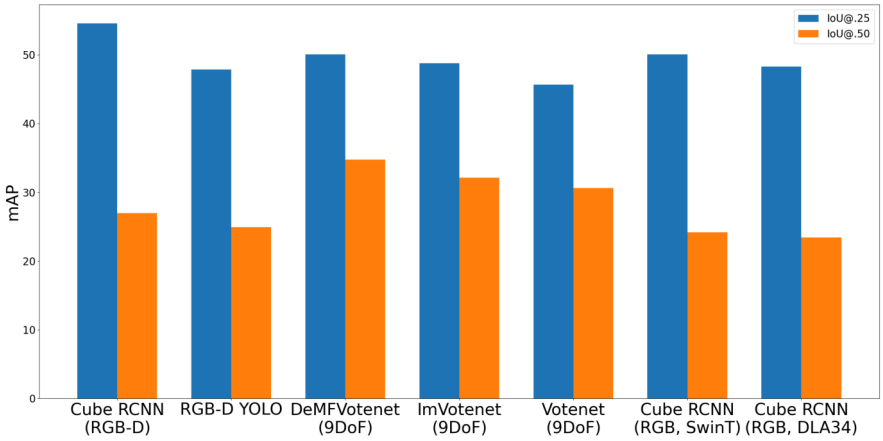

Figure 12: Benchmark results on the DARKO intralogistics dataset. The blue bars correspond to mAP with 9DoF 3D IoU@0.25 and the orange bars represent the stricter metric of mAP with 9DoF 3D IoU@0.5. A more permissive metric can be interpreted as an indicator for method's recall power while the stricter indicates better method's accuracy in object localization.

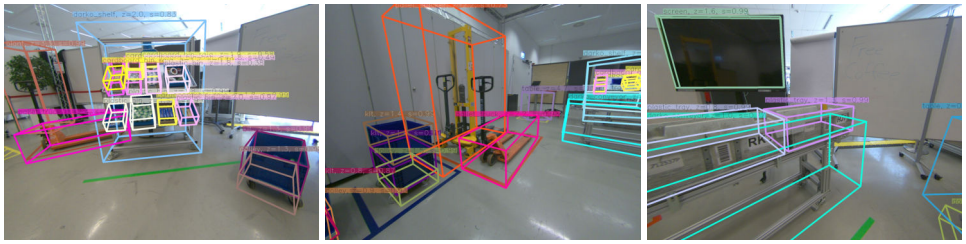

Figure 13: Example object detection results of the best performing method, Cube R-CNN (RGB-D), on the DARKO intralogistics benchmark. Notably, in comparison to *9DoF RGB-D YOLO++*, it can better handle objects which extend beyond the field of view of the sensor.

Since the DARKO intralogistics dataset was recorded with multiple sensors, we are currently investigating cross-sensor generalization of these methods, e.g. training on wide-FOV cameras + 3D lidar, but testing on MS Azure Kinect RGB-D images. Preliminary results indicate that performance drops significantly when the sensor is changed during test time. However, the point cloud-based methods handle the change significantly better. The benchmark results and the insights gained in cross-sensor generalization experiments will play a crucial role in the design of the final module for object-level semantics.

2.3 Towards more data-efficient learning of 3D object-level semantics

In line with DARKO's objective O3, we will now describe our research and development results for label-efficent learning of 3D object-level semantics in the context of the challenging DARKO use-case. In Section 2.3.1, we first describe our label-effiicient annotation strategy, which we used to create DARKO intralogistics dataset. In Section 2.3.2 and Section 2.3.3 we briefly discuss our research on self-supervised learning for downstream tasks of 3D object detection and 3D scene graph prediction respectively.

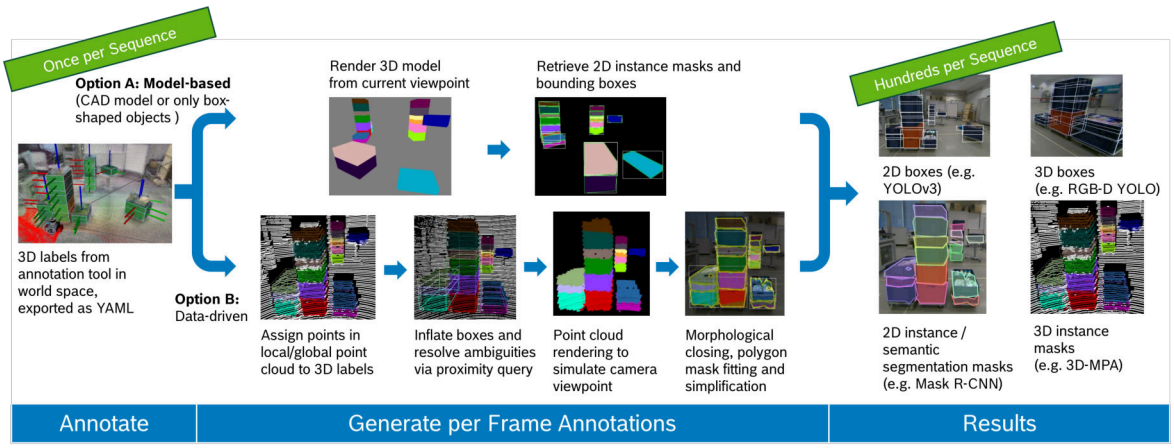

Figure 14: Our label-efficient annotation strategy requires objects to be labeled only once per scene in a global world frame. Using SLAM, scene labels are then propagated into all frames of the recorded sequence. Our automatic label generation pipeline supports two pathways, namely a (CAD-) model-based and a data-driven approach to derive object masks. This way, the method can generate 4 types of annotations: 2D bounding boxes, 2D instance segmentation masks, 9DoF 3D bounding boxes and 3D instance masks.

2.3.1 Label-Efficient Annotation Strategy

The collected datasets and the environments from DARKO target use-case consist predominantly of objects underrepresented in publicly available 3D scene understanding benchmarks, and thus present very interesting challenges for object-level perception. However, to train and evaluate methods on these use-cases, the data needs to be labelled with 3D bounding boxes. Unfortunately, obtaining accurate labels in 3D space is much more costly, tiresome and time-consuming than in 2D space.

Therefore, in line with DARKO's objective O3 on efficient deployment, we developed an efficient 3D labeling strategy that significantly reduces the amount of manual labels required. It is inspired by the approach commonly used in the 6DoF object pose estimation community, where a static scene is observed from multiple viewpoints while tracking the sensor pose with respect to the global reference frame. Manual annotations are then required only once per scene as global labels are projected to different viewpoints along the trajectory to generate abundant per-frame annotations.

Commonly, the sensor tracking is aided by augmenting the environment with fiducials (e.g AR markers). However, their setup in larger scenes with extended objects such as pallet trucks, shelves can quickly become a tedious and time-consuming task. Therefore, we opted for an infrastructure-free, SLAM-based approach where we rely on the multimodal sensor setup available on our portable sensor box and the DARKO robot. We use 3D lidar-based localization in combination with visual inertial odometry to track the sensors across the scene. 3D bounding box labelling is performed only once per scene in a global coordinate system. The resulting annotations are then propagated into each individual frame, as shown in Figure 14. The resulting per-frame labels are then used for training and validation of the developed object-level semantics module.

The overall labeling process consists of several steps. We detail each step in the following paragraphs.

1) Data acquisition: The sensor data is recorded by traversing a static scene and observing the objects from various angles from a mobile sensor. This can be done for example by an

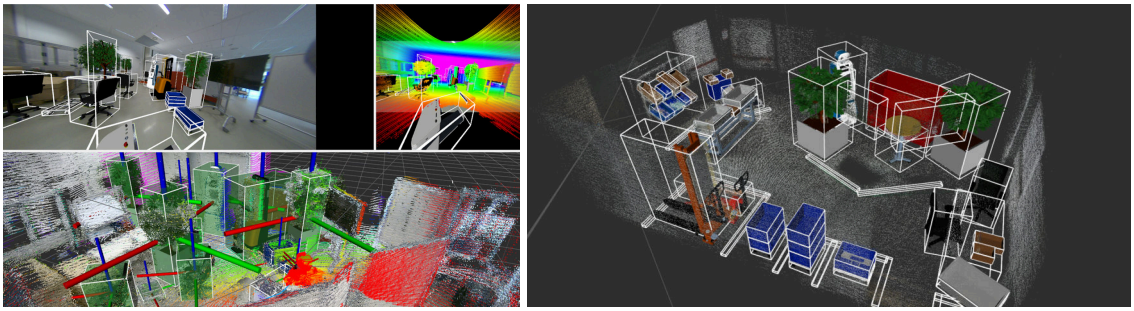

Figure 15: A view from the rviz-based 3D annotation tool is provided on the left. The green boxes are interactive markers and can be dragged, rotated or resized to create 3D groundtruth annotations. Top windows provide a live view of the alignment of the projected groundtruth boxes with RGB and lidar data, thus greatly aiding the annotation process. On the right, a final annotated scene is shown with groundtruth bounding boxes and corresponding meshes. Meshes are used for visualization and are extremely helpful for quality assurance during labelling, e. g. to assert if object orientations have been labelled correctly. Each mesh represents a different category, but not the concrete object instances in the scene.

autonomous robot or by a human with a hand-held sensor. We chose the latter option for the DARKO intralogistics dataset, since it allowed us to observe the objects from more diverse perspectives. An example of such a data collection effort with Bosch's portable sensor box is depicted in Figure 5. Usually we recorded multiple trajectories per static scene to cover different viewing angles and aspects of the environment.

2) 3D SLAM run: The second step involves building 3D map of the environment from the recorded data by running simultaneous localization and mapping on the recorded sequences. The outcome of the SLAM run is a consistent colored 3D point cloud representing the environment and a globally optimized trajectory of the mobile sensor. Examples of such scenes as well as trajectories are shown in Figure 7. For creation of the DARKO intralogistics dataset, we have used an existing Bosch-internal SLAM system.

3) Manual 3D labelling: The global 3D point cloud together with auxiliary egocentric views along the optimized trajectory from the previous step are then utilized in the manual annotation step. For that, we extended an existing open-source and rviz-based labeling tool [24]. We added additional functionalities (e.g. copy & paste and deletion of annotations) and enhanced 3D visualization for improved user experience during labeling of the DARKO intralogistics dataset. Figure 15 provides a glimpse into the annotation tool and an example of a fully annotated scene, which is then processed in subsequent steps.

4) Global label transfer: As discussed earlier, we usually record sensor data along multiple trajectories in the same scene to ensure a good coverage of objects and diversity of views. Assuming that all labeled objects remain static between recordings, only one of the global maps needs to be manually annotated. For the remaining sequences, the labels can be transferred automatically via 3D scan matching of the global 3D point cloud.

As illustrated in Figure 16 we align two global maps using RANSAC-based global registration based on Fast Point Feature Histograms (FPFH) descriptors. We then refine the coarse global registration with point-to-plane ICP. We use the Open3D [25] implementation for both of these methods. The estimated transformation is then used to transfer the labels from one trajectory to another without any additional manual labeling effort.

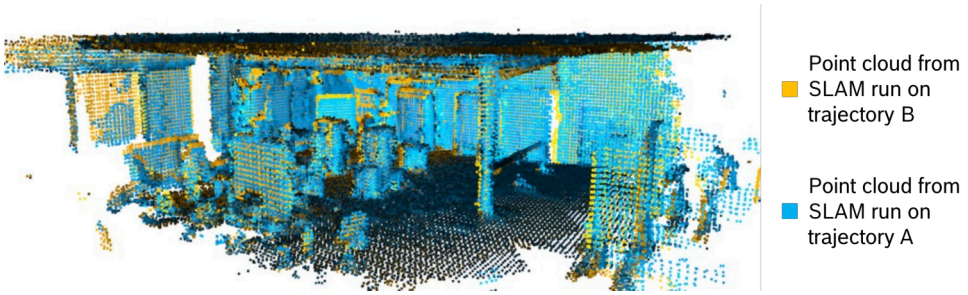

Figure 16: Registered 3D point clouds obtained from SLAM runs on two different trajectories. The computed rigid transform can be applied to transfer labels from one sequence to another.

5) Per-frame label generation: Once the global map of the scene is annotated we then automatically generate per-frame labels as illustrated in Figure 14. The initial phase of this step involves time synchronization of the sensor poses and its data along the trajectory. Then, for each synchronized data frame, our auto-labeling pipeline is applied to obtain foreground object masks and determine object visibility for the current view. Our pipeline supports two alternative pathways – model-based or data-driven. The model-based approach requires to have CAD/mesh models for each of the objects in the scene. The meshes then can be used to render a 2D view from the pose associated with the current data frame. Such an approach automatically handles occlusions and can provide more accurate 2D masks of the objects, but is limited due to its dependency on the availability of object meshes. Therefore, for the generation of our DARKO intralogistics dataset, we have opted for a data-driven approach. Similarly to the model-based approach we use a virtual sensor to render global annotations from the current pose/view-point. However, instead of relying on meshes, here we utilize the depth measurements (from the current data frame) available either from time-synchronized lidar or an RGB-D sensor. With the help of a z-buffer, they are used to check the visibility and occlusion of the global annotations in the current view. The detailed substeps of the data-driven approach are provided in Figure 14 (Option B). Both options are able to generate 2D bounding boxes, 2D instance segmentation masks, 9DoF 3D bounding boxes and 3D instance masks.

6) Dataset Export: Once all of the data frames along a trajectory are processed the auto-generated labels are serialized into a JSON file and can be used for training and testing. We mainly follow a commonly used MS COCO format with custom extensions for 9DoF 3D bounding boxes.

All described steps reduce the manual labeling effort significantly and provide a way to quickly teach-in custom objects e. g. during deployment of a robot in a novel environment. Concretely, in the DARKO intralogistics dataset, we have auto-generated two orders of magnitude more per-frame labels than we had to manually annotate in global 3D maps.

2.3.2 Self-supervised pre-training for 3D object detection

In a MSc thesis conducted within the context of DARKO, we explored self-supervised pre-training techniques for label-efficient learning of 3D object detectors. Our research primarily centered around two methods: VoteNet [18], as previously described a widely adopted and efficient network for 3D object detection, and 3DETR [26], a more recent transformer-based architecture. Both of these methods operate on 3D point cloud data. For these two network architectures, we extensively evaluated three different approaches

for self-supervised pre-training – OcCo [27], STRL [28], and Barlow Twins [29]. We investigated the response of VoteNet and 3DETR to pre-training using quantitative and qualitative methods. Our overall findings suggest the following:

- Self-supervised pre-training methods effectively help to learn 3D object detection from few annotations. We show this for both 3DETR and VoteNet by evaluating them on various simulated low-data settings of SUN RGB-D benchmark [30].
- The choice of pre-training technique impacts the effectiveness of learning with few annotations. As we found, STRL was the most effective out of the three approaches.
- The choice of 3D object detection approach is crucial when learning with few annotations. As we found, VoteNet has a pre-disposition to learning with few annotations as it proved quite effective even without any pre-training.

In addition to our empirical findings, we discovered that 3DETR does not respond to STRL and Barlow Twins pre-training. We remedy this issue by proposing an auxiliary loss as described in our publication [1]. With the proposed pre-training strategy (see

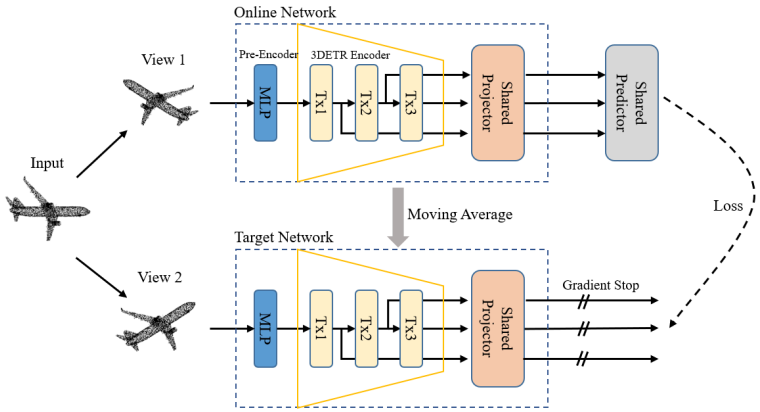

Figure 17: In [1], we propose a simple idea of bringing close intermediate layer representations between the online and target network in the STRL framework. Tx1, Tx2 and Tx3 denotes the first, second and third Transformer layer respectively.

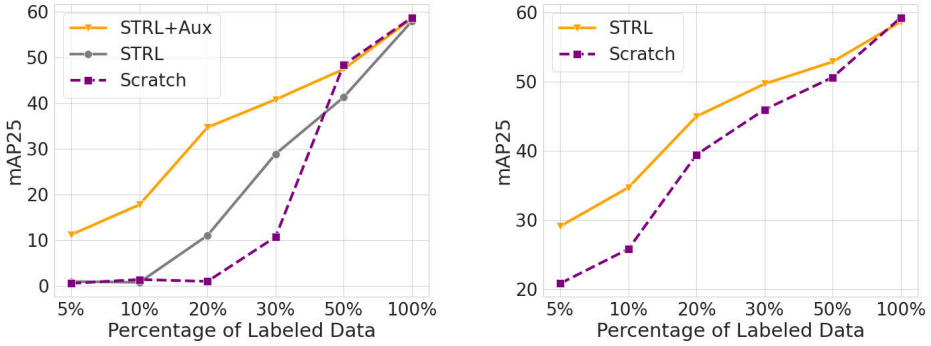

Figure 18: A comparison of STRL pre-trained models against training from scratch at different levels of labeled data on the SUN RGB-D benchmark. The left plot shows results of 3DETR and the right one for VoteNet. STRL+Aux denotes our proposed pre-training method in [1].

Figure 17), 3DETR achieves comparable performance to VoteNet in learning with a low number of annotations. It also improves the learning with full SUN RBG-D data by 0.2 mAP. The improvements gained over 3DETR and VoteNet models trained from scratch at different levels of labeled data are shown in Figure 18.

The insights gained in this line of research potentially can improve the robustness and generalization of the object-level semantics module developed in DARKO, by e.g. pre-training it on larger-scale unlabelled data collected in DARKO-related environments.

2.3.3 Label-efficient Scene Graph Prediction

Bosch pursues a line of research on label-efficient learning for 3D scene graph prediction. DARKO funds this research only partially[3], but the results are of high interest in the context of the DARKO project. Scene graphs provide a compact scene representation which models not only objects, but also their relationships. Such a higher-level representation of the environment is especially useful for a variety of downstream applications such as task and motion planning, context-aware object localization, etc. For example, in DARKO, this could enable task specifications such as *"Pick up an item from a cardboard box next to a blue bin and throw it to the plastic tray on the conveyor"*. Note that here we investigate scene graphs which represent a *local* scene with *semantic* relationships between objects, in contrast to e.g. Rosinol et al. [31] which construct large-scale *global*, hierarchical scene graphs with mostly *spatial* relationships between objects.

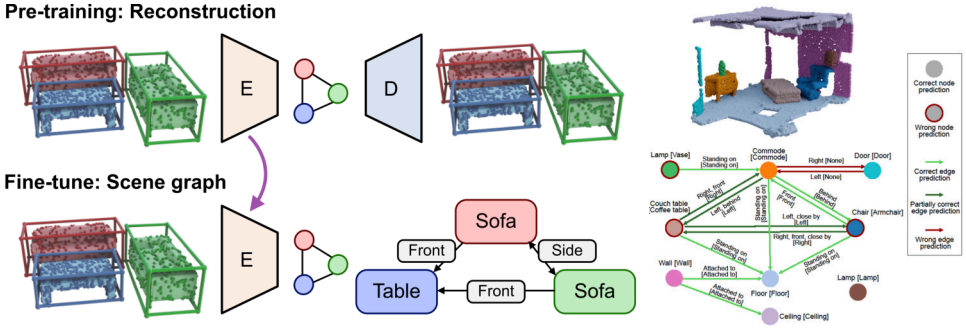

Figure 19: High-level overview of SGRec3D is provided on the left side. An example predicted scene graph and the input point cloud is shown on the right side.

Our research here focuses on how to predict 3D scene graphs from raw sensor data while minimizing annotation effort. In [2, 3] we present SGRec3D, a novel self-supervised pre-training method for 3D scene graph prediction. We propose to reconstruct the 3D input scene (i. e. point cloud) from a *graph bottleneck* as a pretext task. Pre-training SGRec3D does not require object relationship labels, making it possible to exploit large-scale 3D scene understanding datasets, which were off-limits for 3D scene graph learning before. Our experiments demonstrate that in contrast to recent point cloud-based pre-training approaches tailored to other downstream tasks, our proposed method improves the 3D scene graph prediction considerably, which results in state-of-the-art performance, outperforming other 3D scene graph models by +10% on object prediction and +4% on relationship prediction. Additionally, we show that using only a small subset of 10% of the labeled data during fine-tuning is sufficient to outperform the same model without pre-training. Figure 19 provides a high-level overview of the approach and an example of a predicted scene graph.

[3]DARKO funds one of the co-supervisors of the PhD student working on this topic.

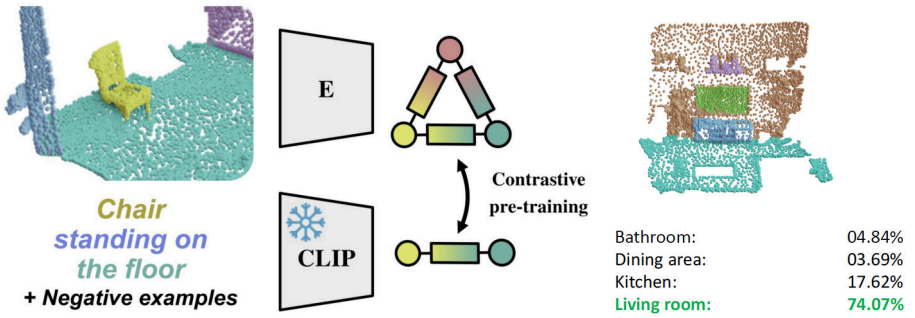

Figure 20: A high level overview of Lang3DSG method proposed in [4] is illustrated on the left next to an example application of zero-shot room classification shown on the right side.

In [4], we further propose Lang3DSG, a *language-based* pre-training approach where we leverage the language encoder of CLIP [32], a popular vision-language model, to distill its knowledge into a graph-based network. As shown in Figure 20, we formulate a contrastive pre-training, which aligns text embeddings of relationships and predicted 3D graph features. Our method achieves state-of-the-art results. Since our scene graph features are now language-aligned, it allows us to query the language space of the features in a zero-shot manner. In this paper, we show an example of utilizing this property of the features to predict the room type of a scene without further training.

Both of the proposed methods are domain-agnostic, but so far they were only evaluated in household environments. Therefore, their potential for exploitation in the DARKO use-case still needs to be assessed.

2.4 Applications to downstream tasks & further semantic scene representations

As part of T2.1 we also want to investigate different semantic representations of the observed scene, and their suitability and accessibility for downstream tasks such as mapping, navigation or task planning.

2.4.1 Mid-level visual priors for semantics-aware navigation

In collaboration with T6.3, we investigated semantic representations for robot navigation. In particular, we looked into learning to predict dense and context-aware cost maps from partially observed top-down semantic maps and mid-level visual representations, where the aim is to perform object goal navigation tasks in unknown environments with the target being specified by a semantic label (e.g. find a couch). Such a navigation task is challenging as it requires understanding of semantic context in diverse settings (e.g. TVs are often nearby couches). Most of the prior work tackles this problem under the assumption of a discrete action policy, whereas in our works [5, 6], we present an approach with continuous control which brings it closer to real world applications.

Our approach is illustrated in Figure 21. We use information-theoretic model predictive control on dense cost maps to bring object goal navigation closer to real robots with kinodynamic constraints. We propose a deep neural network framework to learn cost maps that encode semantic context and guide the robot towards the target object. We also present a novel way of fusing mid-level visual representations in our neural architecture to provide additional semantic cues for cost map prediction. The experiments show that our method leads to more efficient and accurate goal navigation with higher quality paths than the reported baselines. The results also indicate the importance of mid-level representations

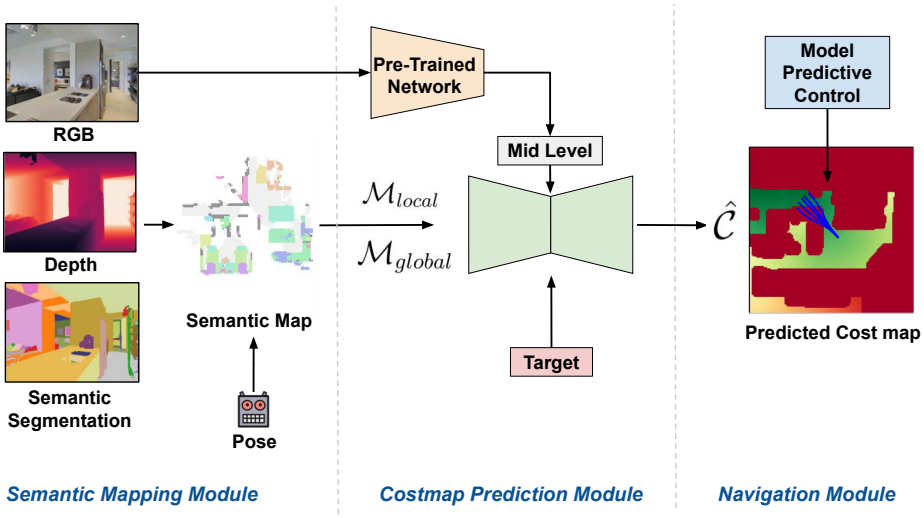

Figure 21: This figure illustrates the framework of our approach for semantics-aware navigation using semantic maps and mid-level visual priors. Our framework is composed of three modules: A *semantic mapping module* which constructs a semantic map of the environment as the robot explores. The *cost map prediction module* which predicts the cost map for navigation based on the input semantic map, mid-level representation and target object. Finally, the *navigation module* generates the optimal control using sampling-based MPC.

for navigation by improving the success rate by 8 percentage points. Mid-level visual priors are an interesting alternative to the explicit semantic representations, since they can be learned in a self-supervised manner from unlabeled data. This observation has been supported by the recent surge of large pre-trained vision-language models and their applications to navigation.

2.4.2 Semantic SLAM

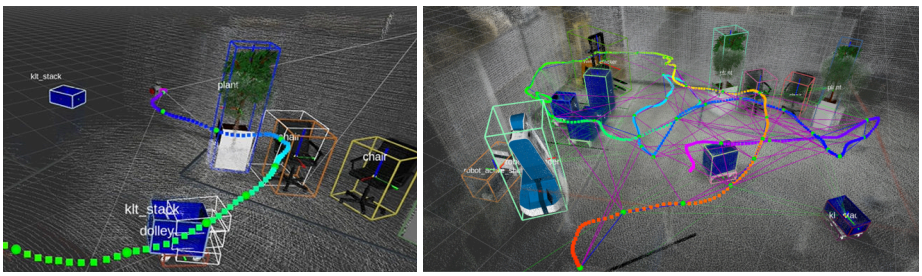

Figure 22: The left subfigure illustrates an interim state of a semantic SLAM run. The white bounding boxes represent the object detections by *9DoF RGB-D YOLO++* from the current view-point, whereas the colored boxes with meshes visualize optimized 3D landmarks. The final globally optimized semantic map with the full trajectory of the sensor is shown on the right. Note that even with noisy per-frame object detection the final map is accurate.

Object-level semantics have tight synergies with semantic SLAM. On the one hand, semantics can inform SLAM for more accurate localization, on the other hand object-level semantics can be improved using multi-view observations, thus potentially requiring less accurate object detectors. We investigated the latter aspect during the first two periods of

DARKO. We explored this topic with two different representations of semantics, namely, 9DoF 3D bounding boxes and 2D panoptic segmentation uplifted to 3D. For easier future exploitation, we used an existing Bosch-internal SLAM system developed outside of the DARKO project for these experiments, which is different from the system being developed by ORU in WP3 for deployment on the DARKO robot. Nevertheless, these experiments provide valuable insights also for the design of semantic mapping in DARKO.

For the first variant, we have integrated our *9DoF RGB-D YOLO++* detector with Bosch's internal pose graph-based SLAM. Detected 9DoF bounding boxes are used as landmarks and are continuously updated when new observations arrive. We found that multi-view observations can greatly improve the accuracy of the noisy 9DoF 3D object detections from a single view-point. This can be seen in Figure 22, which provides a glimpse into one of our semantic SLAM runs in the robotics lab.

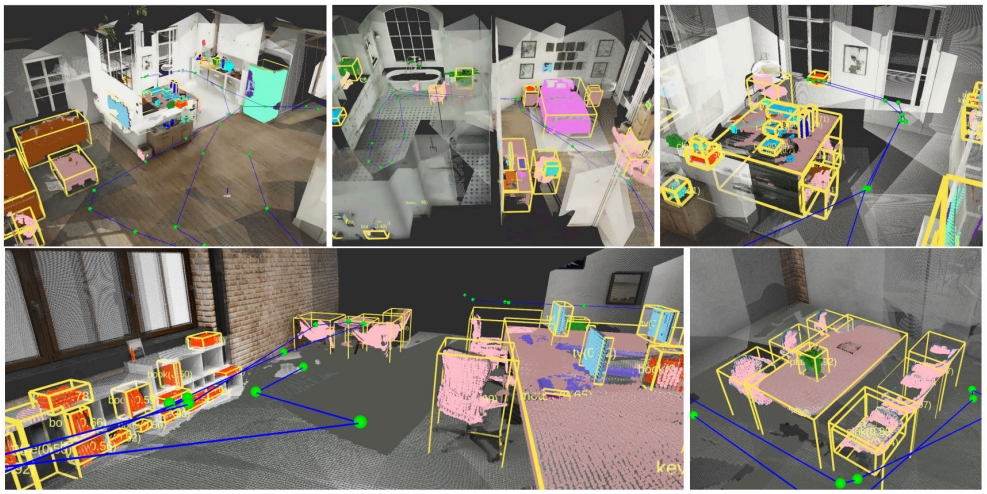

Figure 23: Example results of panoptic hierarchical semantic SLAM on the scenes from the semantic SLAM challenge as part of Embodied AI workshop at CVPR 2022.

In addition, during the first period of the DARKO project, Bosch successfully participated in the Semantic SLAM challenge at CVPR 2022, where we tried out an alternative approach [33] based on 2D-to-3D uplifting for predicting 3D object-level semantics. Our method, panoptic hierarchical semantic SLAM, utilizes a 2D panoptic/semantic segmentation frontend together with a custom foreground-background segmentation to create panoptic point clouds per frame. These point clouds are then voxelized in a hierarchical manner, and a label voting for voxel centers is performed, resulting in a global panoptic point cloud. Bounding boxes are extracted from this global panoptic point cloud by either top-down projection or clustering. Finally, a filtering step is performed to obtain the final object detections. One of the advantages of this approach is that it can leverage existing strong 2D methods. However, 3D bounding boxes are extracted as a post-processing step making the approach less efficient. Also, in the CVPR challenge only axis-aligned bounding boxes were used. To fulfill DARKO requirements, this approach would need to be extended to support 9DoF bounding boxes, which is not a trivial task. The example scenes and results from the competition are shown in Figure 23.

2.5 Remaining challenges and outlook for T2.1: Object-level semantics

The initial object-level semantics module based upon 9DoF RGB-D YOLO++ has been successfully trained on the DARKO intralogistics dataset and has been integrated and

deployed on the DARKO platform for real-time inference. However, on the robot, its performance has dropped significantly in comparison to results on the original test split of the dataset. This might partially be attributed to the domain shift caused by differences between the training and inference environments (robotics lab at Bosch vs. ARENA 2036), as well as to the limited diversity of backgrounds in our dataset. However, our further investigation reveals that the main reason is likely the lack of cross-sensor generalization of the trained model. The training dataset is created using RGB+lidar data from the portable sensor box, but the currently deployed model is run on Azure Kinect RGB-D data.

The test split of the DARKO intralogistics dataset was recorded with multiple sensors (including also the Azure Kinect DK). Therefore, we are currently performing quantitative experiments to evaluate the extent of this issue for the different 9DoF 3D object detectors from our benchmark described in Section 2.2.2. Our initial results indicate that while it affects all of the methods, the point cloud-based approaches seem to generalize better across sensors. This issue could potentially be remedied by introducing additional augmentations during training. Also, more robust and generalizable models could be obtained by using self-supervised pre-training techniques as discussed in Section 2.3.2. We will consider these options in the upcoming final phase of the project.

As one of the main challenges for the remainder of the project, we identify the need to properly deal with fisheye optics (see also T2.5). For broad-level scene understanding, the DARKO robot is equipped with two fisheye cameras and lidar covering 360° field of view. As it is impossible to map an uncropped, 180° fisheye image to a rectilinear image without extreme distortions, we are currently investigating different data augmentation and pre-processing techniques which could enable training of 3D object detectors robust to fisheye distortions on existing (non-fisheye) datasets. As an intermediate step, we are considering to work with rectified, partial fisheye images with the drawback of a reduced field of view and discontinuities at the boundaries of the cropped image.

Figure 24: Egocentric view from the camera with fisheye lens mounted on the DARKO robot at ARENA 2036. A 2D segmentation result from SAM [34] is shown on the right, demonstrating that the method can successfully generalize to images with fisheye distortions.

In addition, we are closely observing the recent developments in foundation and large vision-language models (VLMs) and assessing their suitability to the DARKO use-case. The strong generalization capabilities of these models could potentially address many of the aforementioned issues. For example, as shown in Figure 24, Segment Anything (SAM) [34] can successfully segment fisheye images without any additional training. While SAM is class agnostic and does not provide labels, projects like Grounded SAM [35] demonstrate how to convert it into an open-vocabulary method.

How to uplift these methods to 3D is another research question. As discussed in Section 2.4.2, Bosch participated in the Semantic SLAM challenge at CVPR 2022, where our method relied on 2D segmentation methods which were uplifted to 3D via depth

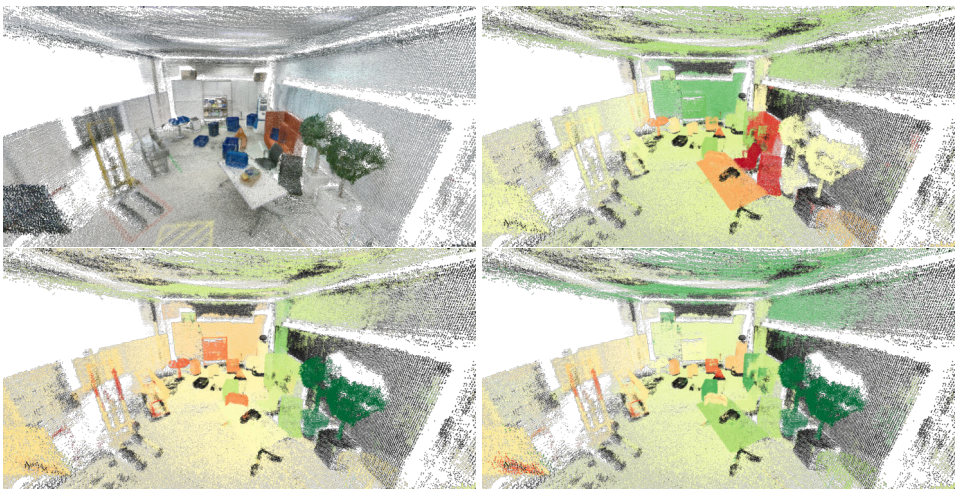

Figure 25: OpenMask3D [36] results on one of the scenes from our DARKO intralogistics dataset. The top left subfigure shows a colored point cloud of the scene. The rest of the images visualize 3D response maps to different language queries. Red color depicts a high-level response while green corresponds to a low-response. The following queries were used: *"something to sit on" (top right)*, *"conveyor" (bottom left)*, *"forklift" (bottom right)*.

measurements and multi-view observations [33]. A similar approach could be applied in the DARKO context. Methods like OpenMask3D [36] offer open-vocabulary instance segmentation directly in 3D. Example results from different natural language queries on a scene from our DARKO intralogistics dataset are shown in Figure 25. While OpenMask3D was able to identify objects to sit on, it was not able to locate neither forklifts nor the conveyor. In general, our initial insights suggest that open-vocabulary methods as well as VLMs are still susceptible to domain shifts and do not generalize well to intralogistics scenarios prevalent in DARKO. However, the potential of these methods is huge. Therefore, we are constantly looking into how to leverage these models for DARKO perception tasks.

Finally, in the last stages of the DARKO project, we would also like to assess the developed methods for object-level semantics on in-the-wild data collected at the warehouse site of the DARKO target use-case.

3 T2.2: Perception for manipulation

Figure 26: Challenging objects for T2.2: Perception for manipulation from DARKO's lead use-case. Objects are often tightly spaced, come in various shapes, sizes & colors, and might be wrapped into transparent plastic bags. Each source bin contains only objects of the same kind.

Task T2.2 deals with the challenge of determining suitable strategies for object picking using sensor observations. As can be seen in Figure 26, the wide variety of objects and the manner in which they are placed in the DARKO scenario pose a challenge for object picking. The goal of this task T2.2 is to develop a module that provides stable and task-relevant grasps to inform object picking (T4.3). This module primarily focuses on addressing these challenges:

i) Picking objects that do not afford direct grasp acquisition,

ii) Picking objects with degradation of both appearance and geometric data, and

As per the DARKO work plan, there are two interfaces envisioned to enable object picking in tandem with T4.3. First, task T2.2 is to develop a method for direct grasp pose estimation. This interface would provide a set of possible grasp poses to T4.3, which will then select a suitable collision-free grasp to execute. The second interface is to be used in the cases when a collision-free grasp is hard to acquire: e.g., tightly packed goods, or objects close to a boundary wall. To handle such scenarios, T2.2 will develop a grasping method that relies on executing a sequence of parametrized manipulation primitives. The primary goal of T2.2 in this case is to detemine what primitives should be used, based on the current workspace observation. The grasp pose interface is discussed in detail in sections 3.1—3.3, while the manipulation primitives-based interfaces is covered in section 3.4.

Key contributions in T2.2

Our contributions for task T2.2, discussed in the next subsections, include:

- Initial research focused on *grasp pose estimation*, which involves directly regressing grasp poses from RGB-D sensor data. This effort resulted in the publication of one RAL'22 journal paper [7] and one ICRA'22 conference paper [8] in period 1. Additionally, we defined interfaces connecting to T2.1 and T4.3.
- To better align the requirements of the DARKO setup, we also assessed our method using a multi-finger soft hand, subjecting it to various visual disturbances, including plastic wrapping.

- Certain (flat) objects may not readily allow for direct grasp acquisition. To further address the task, we therefore propose a novel method that focuses on learning and combination with *manipulation primitives* to address the challenge. This effort resulted in the publication of one IROS'23 workshop paper [9] and one under-review conference paper [10].
- Subsequently, in period 2, we focused on *deployment of the module on the DARKO robot platform* and experimentally demonstrated it in practice with a DH-3 gripper at ARENA 2036 and at the Automatica 2023 fair.

3.1 A novel method for object pose estimation

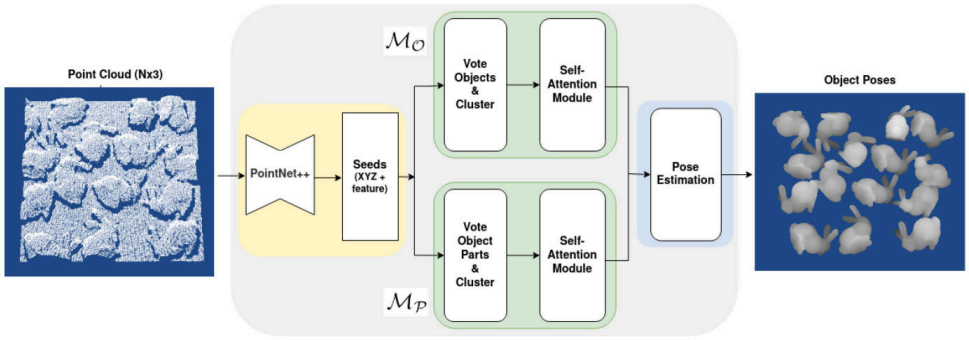

Figure 27: The architecture of the proposed network for 6DOF object pose estimation in point cloud data. Based on a deep Hough voting neural network [18], we introduce self-attention modules for encoding the dependency of object parts and object instances into features to boost the performance of object pose estimation.

The first interface we developed in DARKO is based on estimating the pose of each target object placed in a container. This method was proposed in a RAL'22 journal paper [7], with the key idea being to estimates object 6DOF pose from 3D point cloud data. This method is robust to noise and occlusion, can be trained in simulation and deployed directly to real-world scenes, and is able to deal with multiple objects in clutter. We summarize the main ideas of the method and obtained results below.

3.1.1 Task: 6DoF Object Pose Estimation

The 6DOF object pose estimation task aims to determine the spatial configuration of objects in three-dimensional space precisely. This task encompasses both three-dimensional translation and rotation components, providing a comprehensive understanding of an object's position and orientation. Based on the current sensor data, the objective is to learn a transformation matrix that represents the rotation and translation between the object model and the current view of the object. Using the object pose and a database of object-specific grasps, task T4.3 can then select and plan a motion to an appropriate grasping pose.

3.1.2 Method for object pose estimation

In [7], we introduced an end-to-end network for the precise estimation of 6DOF object poses from 3D point clouds. We consider object instances of the same class packed tightly in a box, and treat each instance as composed of several distinct *parts*. Our approach uses

	[37]	[9]	[15]	[17]	[18]	[19]	Ours
Brick	0.10	0.19	0.37	0.42	0.36	0.38	**0.48**
Bunny	0.38	0.27	0.44	0.54	0.46	0.52	**0.61**
C.stick	0.37	0.15	0.50	0.54	0.50	0.52	**0.60**
C.cup	0.08	0.27	0.40	0.45	0.41	0.44	**0.52**
Gear	0.21	0.28	0.40	0.51	0.42	0.50	**0.64**
Pepper	0.04	0.06	0.28	0.30	0.24	0.25	**0.39**
Tless 20	0.11	0.21	0.31	0.38	0.35	0.30	**0.44**
Tless 22	0.09	0.13	0.22	0.29	0.20	0.23	**0.37**
Tless 29	0.19	0.28	0.39	0.35	0.40	0.37	**0.46**
Gear shaft	0.18	0.32	0.38	0.48	0.41	0.51	**0.65**
Ring screw	0.27	0.40	0.54	0.56	0.45	0.54	**0.67**
MEAN	0.18	0.23	0.38	0.44	0.38	0.41	**0.53**

Method	Attempts	Success Rate	Time (s).
Linemod [37]	342	43.9%	1.60
Linemod+PP [38]	295	50.8%	3.21
PPF [9]	287	52.3%	1.52
PPF+PP [38]	270	55.6%	3.00
[15]	256	58.6%	0.30
G2L-net [17]	238	63.0%	0.21
[18]	252	59.6%	0.18
[19]	248	60.5%	0.20
Ours	**207**	**72.5%**	**0.15**

Table 3: (Left) AP values for different algorithms on objects for the SILÉANE dataset and the Fraunhofer IPA dataset. (Right) Results of real robot experiments with 150 objects for each method. Computation time (Time) is only for grasp generation per point cloud. (See the original paper [7] for references and further details.)

self-attention to capture both part-to-part and instance-to-instance correlations, enhancing the model's capability to overcome challenges inherent in cluttered scenes, such as noise and occlusion.

As shown in Figure 27, the initial component processes a 3D point cloud, extracting seeds as high-dimensional features. Following this, the second component, denoted as M_P, operates on these high-dimensional features. Its primary functions include learning part proposals and leveraging a self-attention module to facilitate higher-level interactions between part proposals. Similarly, the third component, M_O, takes seeds as input to learn object proposals. Subsequently, it encodes instance-to-instance correlation information through the application of a self-attention module. The voting mechanism employed in both M_P and M_O draws inspiration from VoteNet [18] — a prior method for 3D object detection, discussed for T2.1 in sec. 2.2.2. Notably, the addition of an attention mechanism allows local feature information to propagate globally, which results in better pose estimation. Intuitively, this results from the fact that neighboring objects physically support each other and impose constraints on interpenetration, which would otherwise be difficult to take into account without considering global information. Finally, a pose estimation module combines features from these two modules and generates the pose of objects.

3.1.3 Experimental results for object pose estimation

Dataset Experiments We use about 206,000 synthetic scenes from the Fraunhofer IPA Bin-Picking dataset [37] to train our method and use the real-world scenes from the same dataset and Siléane dataset [38] for evaluation. Because the coffee cup (C.cup) is not in the Fraunhofer IPA Bin-Picking dataset, we generated 10,000 synthetic scenes to add to our training set.

Table 3 (left) summarizes the comparison results between our method and current point cloud-based approaches on 11 objects, where 9 objects are from the Siléane dataset and the 2 novel objects (gear shaft and ring screw) from the Fraunhofer IPA dataset. Our proposed approach achieves state-of-the-art results and outperforms others in all objects, obtaining an average precision of 53%.

Figure 28 shows qualitative results. The first row comprises 3D point cloud inputs, followed by the ground truth poses in the second row. The third row displays the retrieved pose hypotheses generated by our baseline method, while the last row showcases the retrieved pose hypotheses produced by the proposed method. The visualization includes color-coded representations of point-wise distance errors, ranging from 0 (green) to values greater than 0.2 times the diameter of the object (red).

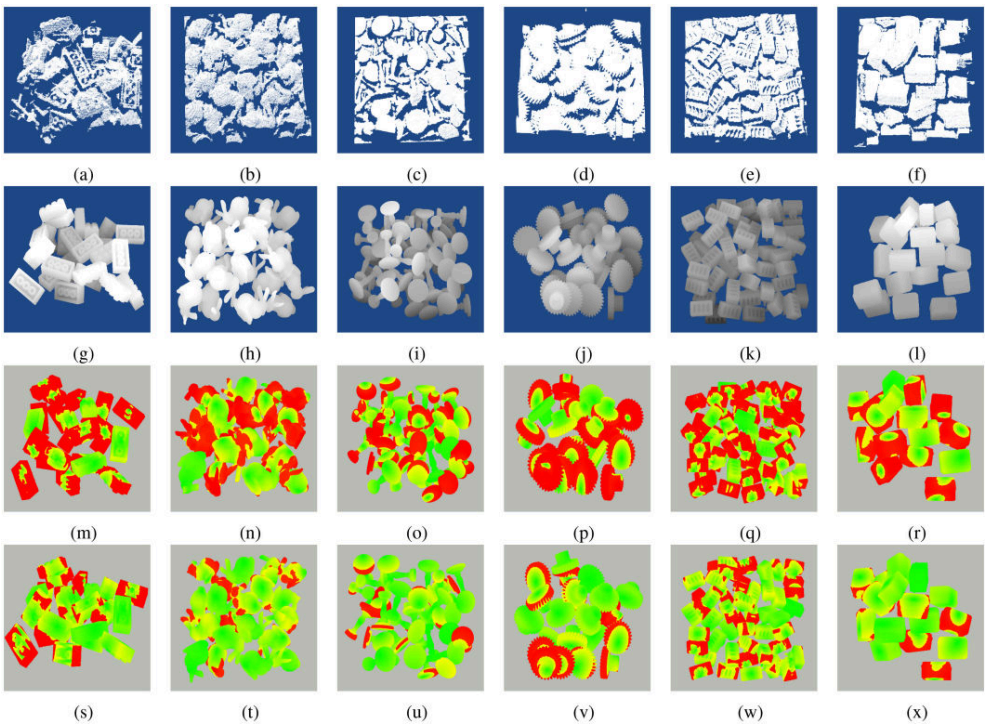

Figure 28: Visualization of pose estimation results includes rows representing, from top to bottom: 3D point cloud inputs, ground truth poses, baseline results, and the outcomes from our method. The colors represent the difference between the ground truth pose and the estimated pose. Green indicates less difference, while red indicates a larger difference.

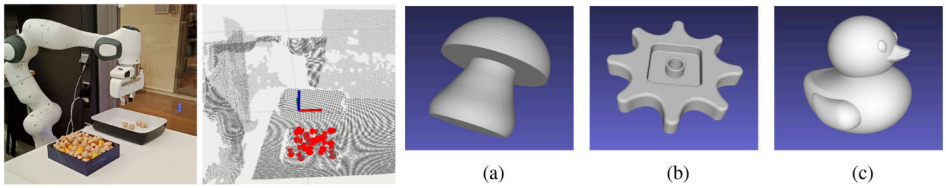

Figure 29: Real robot experiments on pick-and-place task. (Left) We evaluate our method in the real robot on a pick-and-place task. (Right) 3D models of objects in real robot experiments. (a) Wood sponge; (b) Plastic gear; (c) Rubber duck.

Real Robot Experiments In our robot experiments, we evaluate the performance of the proposed approach and compare it with related works in a pick-and-place task, as illustrated in Figure 29. The experiment is conducted using a Panda robot arm with 7 degrees of freedom (7-DOF), featuring a parallel-jaw gripper. The ASUS Xtion PRO LIVE sensor captures input point clouds, and the entire system is implemented using the ROS and MoveIt! frameworks.

In these experiments, we randomly arranged 150 object instances. Predicted object poses guide the selection of grasp configurations from a pre-computed database. The robot executes the picking process by selecting objects from the top and repeating the procedure until all objects are retrieved. The success rate of the integrated system is calculated as the number of objects successfully picked divided by the total number of picking attempts made. Table 3 (right) shows that our method achieves a 72.5% success rate, outperforming baseline methods by a considerable margin.

3.2 A novel method for grasp pose estimation

While the approach from section 3.1 relies on recovering the pose of a set of known objects, this strategy has a number of limitations. First, it is difficult to extend this approach to objects not previously seen during training: both due to issues with defining and regressing 6DOF poses for such objects, but also due to the difficulty of planning grasps on unknown objects. And second, when objects are cluttered together, relying on a discrete set of grasps stored in a database is suboptimal, as a large percentage of these grasps are obstructed by neighboring objects. To address these challenges, we proposed an alternative method [8] that directly extracts feasible grasps from the scene. This approach operates on RGB-D data and was trained on a large set of sample objects, resulting in a general grasp estimation approach that does not rely on a known set of objects and is able to propose collision-free grasps in cluttered scenes.

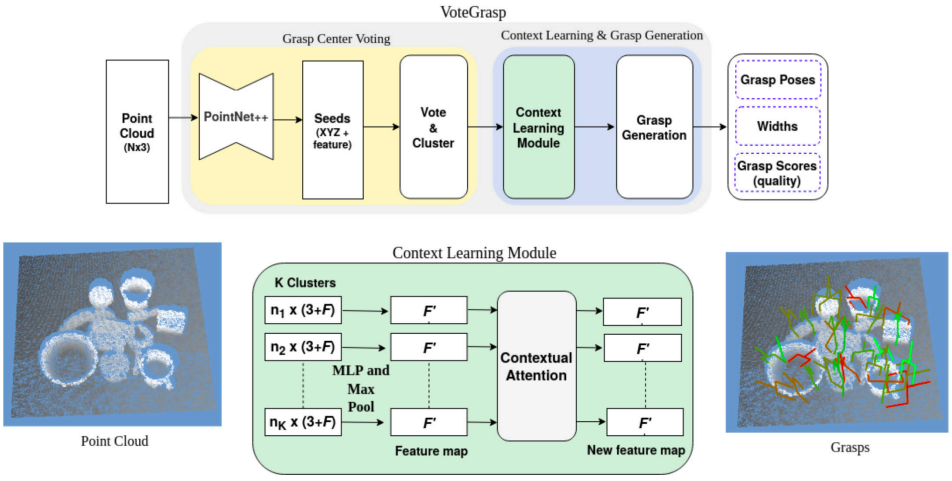

Figure 30: Architecture of our proposed method for grasp pose estimation

3.2.1 Task: Grasp Pose Estimation in Cluttered Environments

The grasp pose estimation task aims to determine optimal grasp poses for successful manipulation or grasping. The grasp pose includes information about the position and orientation of the gripper or hand in three-dimensional space. Given the current sensor data, the objective is to learn a transformation matrix that represents the rotation and translation between the gripper and the robot/camera frame. In cluttered scenes, estimating grasp poses becomes more challenging due to the presence of occlusions, varied object shapes, and complex spatial arrangements. In addition, neighboring objects often invalidate otherwise good grasps, which can result in a long planning time (each grasp needs to be tested), or worse, a complete depletion of feasible grasp poses.

3.2.2 Method for grasp pose estimation

In [8], we introduced a new framework for 6DOF grasp generation (VoteGrasp) for robustly estimating 6DOF collision-free grasp configurations in cluttered environments given a 3D scene point cloud. To overcome the challenge of occlusion, the proposed model generates candidates by casting votes and accumulating evidence for feasible grasp configurations. We exploit contextual information by encoding the dependency of objects in the scene into features to boost the performance of grasp generation.

	Seen			Unseen (but similar)			Novel		
	AP	$AP_{0.8}$	$AP_{0.4}$	AP	$AP_{0.8}$	$AP_{0.4}$	AP	$AP_{0.8}$	$AP_{0.4}$
GG-CNN [39]	15.5/16.9	21.8/22.5	10.3/11.2	13.3/15.1	18.4/19.8	4.6/6.2	5.5/7.4	5.9/8.8	1.9/1.3
Chu et al. [40]	16.0/17.6	23.7/24.7	10.8/12.7	15.4/17.4	20.2/21.6	7.1/8.9	7.6/8.0	8.7/9.3	2.5/1.8
GPD [8]	22.9/24.4	28.5/30.2	12.8/13.5	21.3/23.2	27.8/28.6	9.6/11.3	8.2/9.6	8.9/10.1	2.7/3.2
PointNetGPD [10]	26.0/27.6	33.0/34.2	15.4/17.8	22.7/24.4	29.2/30.8	10.8/12.8	9.2/10.7	9.9/11.2	2.7/3.2
Fang et al. [11]	27.6/29.9	33.4/36.2	17.0/19.3	26.1/27.8	34.2/33.2	14.2/16.6	10.6/11.5	11.3/12.9	4.0/3.6
Gou et al. [41]	28.0/32.1	33.5/39.5	17.8/20.9	27.2/30.4	36.3/37.9	15.6/18.7	12.3/13.1	12.5/13.8	5.6/6.0
$Ours^{-}$	29.2/33.8	34.7/41.0	19.1/22.2	28.3/31.7	37.2/39.1	16.8/19.9	13.6/15.0	13.7/15.1	6.8/7.2
Ours	**34.1/37.5**	**38.9/45.6**	**24.0/27.7**	**33.0/35.9**	**40.8/43.3**	**20.5/24.7**	**16.9/18.5**	**17.0/18.5**	**10.0/10.6**

Table 4: The table shows the results on the GraspNet-1Billion test set captured by RealSense/Kinect sensors respectively. $Ours^{-}$ denotes our proposed network without context learning module. (See the original paper [8] for references and further details.)

As shown in Figure 30, our model builds on a deep Hough voting neural network [18] to vote for grasps and we add a self-attention context learning module. VoteGrasp includes two modules: grasp center voting and context learning & grasp generation. The former estimates multiple votes (positions) and their features to represent multiple grasp centers from a 3D scene point cloud. In more detail, we apply the PointNet++ [19] architecture with multi-scale grouping to select M interest points (called seed points) and enrich them with high-dimensional features. The seed points are then fed into an MLP to compute votes per seed and then find neighboring votes as a subset by clustering. The output is K groups of vote subsets, where each group corresponds to a grasp center. The latter estimates grasp poses, gripper opening widths, and grasp scores from the votes and their features. Based on the self-attention-based models [39, 40], our context learning module computes a new feature map from all clusters (groups) to learn the context that considers the relationships between all clusters. It is this global context propagation step that allows our approach to propose collision-free grasps by considering not only local-shape information, but also context from neighboring surface patches. Last, a multi-layer perceptron network is applied to output a ranked list of grasps from the new feature maps.

3.2.3 Experimental results on grasp pose estimation

Dataset Experiments We evaluate our method and compare with other state-of-the-art methods on the publicly available GraspNet-1Billion [41] dataset. Table 4 shows the performance of our approach compared to state-of-the-art methods in the GraspNet-1Billion testing set. Our method performs better than state-of-the-art methods in both seen, unseen, and novel objects.

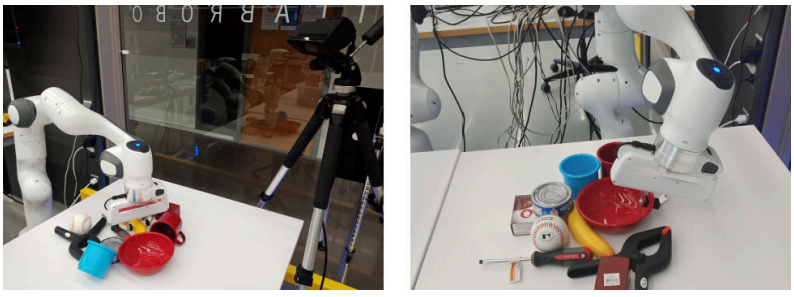

Figure 31: Real-world grasping experiment. A Franka Panda robot selects the highest ranked grasp for a target object, plans a trajectory using MoveIt and executes if feasible.

Real Robot Experiments The real robot experiments are conducted with a Franka Emika Panda robot arm with 7-DOF, equipped with a parallel-jaw gripper, as shown in Figure 31. To capture the input point clouds, we use a Microsoft Kinect v2. The whole system is

implemented using the ROS and MoveIt! frameworks. The objects are randomly placed within the workspace of the robot arm and the camera. A grasp was considered a success if the robot could grasp and lift the object within one attempt. In our experiments, the robot succeeds on 70% of the grasps using our proposed approach.

3.3 Extended testing with the proposed grasp pose estimation method

Figure 32: Picking up a DARKO object with the multi-finger soft hand

3.3.1 Testing with various DARKO objects in multi-finger soft hand

In the preceding sections, we discussed our real robot experiments conducted in cluttered scenes, as detailed in the mentioned papers. To further investigate the ability of our proposed method in the DARKO setup, this section extends our testing to include DARKO objects with a multi-finger soft hand, based on [8]. An example is shown in Figure 32.

Challenges with Transparent Plastic Bags To gauge the robustness of our grasp pose estimation method in the DARKO scenario, we expanded our testing to incorporate a range of DARKO objects. Our evaluation encompassed scenarios such as grasping objects enclosed within transparent plastic bags, which is common practice for some of the DARKO test objects. The transparency and reflective properties of these bags introduce challenges, degrading both appearance and geometric sensor data.

Figure 33: Six DARKO objects inside transparent plastic bags.

DARKO Objects We select six DARKO objects, as shown in Figure 33. Notably, these six DARKO objects, exhibiting diverse appearances and geometric structures, are all packed within transparent plastic bags.

Table 5: Real robot experiments with six plastic bag objects using multi-finger soft hand

Method	obj-0	obj-1	obj-2	obj-3	obj-4	obj-5	**Average**
VoteGrasp[8]+S	9/10	4/10	5/10	7/10	7/10	7/10	**65%**

3.3.2 Real robot experiments on grasp pose estimation

Experimental Setup We train our method on the GraspNet-1Billion dataset [41] and test it on the real robot directly without model fine-tuning. The target object is placed on a flat table. We perform ten grasping attempts and calculate the average success rate. The pick is considered a success if the robot can pick up the object in one try. To adapt the grasp pose from a finger gripper to the multi-finger soft hand, we manually calculate the transformation of the grasp point between the two hands (UNIPI's soft hand and the Franka 2-finger gripper).

Experimental Results In Table 5, the method VoteGrasp+S denotes our proposed method VoteGrasp [8] with a multi-finger soft hand. Table 5 shows that our method can achieve a 65% average success rate on DARKO objects with plastic bags. We note that *obj-1* and *obj-2* have significant performance drops. For *obj-1*, the robot usually failed when it grasped the plastic bag rather than the object itself. Because this object is much heavier than others, it is easy to slide after picking. For *obj-2*, the robot usually tries to grasp one of the ends, which results in the object being easily knocked out of place upon contact with the hand. However, despite similarities in shape, interestingly *obj-4* did not have similar issues. We think maybe the different material affects the degradation of both appearance and geometric sensor data differently. Besides, most of the failures happen when the object is placed too close to the robot. When the robot tries to reach the grasp pose that is too close, it often reaches joint limits and thus stops. Based on this, the workspace for a multi-finger soft hand should have more distance to the robot compared to the two-finger gripper.

3.4 Parameterized manipulation primitives for flat object grasping

In this section, we present a novel approach based on manipulation primitives for effectively grasping a large, horizontally lying object. As highlighted in the preceding section, identifying suitable grasp poses in this particular scenario poses a challenge for conventional collision-free grasp pose estimation methods. In response, we introduce a manipulation primitive-based method wherein these primitives dynamically alter the object's state through interactions with both the object itself and the surrounding environment. This approach facilitates a more streamlined process for estimating grasp configurations, enhancing the overall adaptability of the system.

3.4.1 Task: Flat object grasping

We apply our method to address a variant of the *occluded grasping* task [42]. In this scenario, a robot arm equipped with a simple parallel jaw gripper is tasked with picking up a flat object situated on a tabletop (see Figure 34), where the workspace is bounded on all four sides by immovable walls. Given that the object is only graspable along approach directions that collide with the table, solving this task necessitates non-prehensile manipulation and interaction with the environment. Specifically, the robot must strategically push the object against one of the four boundaries, pivot it to facilitate flipping, and ultimately execute a grasp from the top.

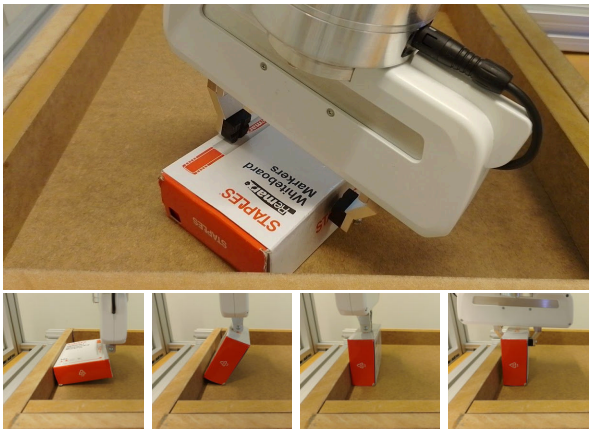

Figure 34: *Top:* In the initial pose, all feasible grasps on the target object are occluded by the environment. *Bottom (left to right):* We learn to push the object to a wall and exploit it as a pivot to flip the object up and finally grasp it from the top.

3.4.2 Method for flat object grasping

Overview. In [10, 9], we propose a novel method for solving the occluded grasping task using hierarchical reinforcement learning with manipulation primitives. However, employing manually designed parametric controllers necessitates expert input in designing, implementing, and tuning the primitive controllers. Instead, we propose to learn hierarchical control policies whose actions are a series of parametrized learned manipulation primitives. We address the occluded grasping task by employing a variation of hierarchical Deep Q Networks (DQN) [43]. As shown in Figure 35, a high-level agent is responsible for selecting a sequence of pose-parametrized manipulation primitives, each of which is in turn assigned to a low-level agent responsible for selecting appropriate primitive-specific actions. The goal of the high-level agent is to learn a policy that maps a sensor observation in the form of depth data to an appropriate pose-parametrized manipulation primitive. The low-level manipulation primitive constitutes a feedback controller that is learned through interaction.

Manipulation primitives. We use three manipulation primitives: a push primitive that achieves in-plane object motion; a flip primitive that uses contact with the environment to pivot an object; and a grasp primitive that picks directly graspable objects. These three primitives can be combined to solve the occluded grasping task and demonstrate extrinsic dexterity manipulation. In this paper, we employ a low-level DQN agent to learn the complex flip primitive, while we design the other two primitives manually.

High-level agent. In the high-level agent, the observation space is a depth map of the workspace and the action space is composed of a pixel coordinate that corresponds to samples of the state-action-value function Q. Each pixel represents a starting pose (x, y, θ_i) and a primitive id ϕ, where (x, y) encodes the (x, y)-th pixel of the depth image; $\theta_i = 2\pi i/K$ encodes the i-th discrete end-effector rotation of K possible directions around the Z-axis (K=16 in this paper); while ϕ is a categorical choice between a group of primitives Φ, with $\|\Phi\| = 3$. We use a sparse reward for each primitive. Successful actions with the *flip primitive* and the *grasp primitive* are rewarded by $r_t^{H_f} = 1$ and $r_t^{H_g} = 1$ respectively. The reward of successful *push primitive* actions is set to a value of $r_t^{H_p} = 0.2$ on success and $r_t^{H_p} = 0.1$ on a change in the workspace configuration.

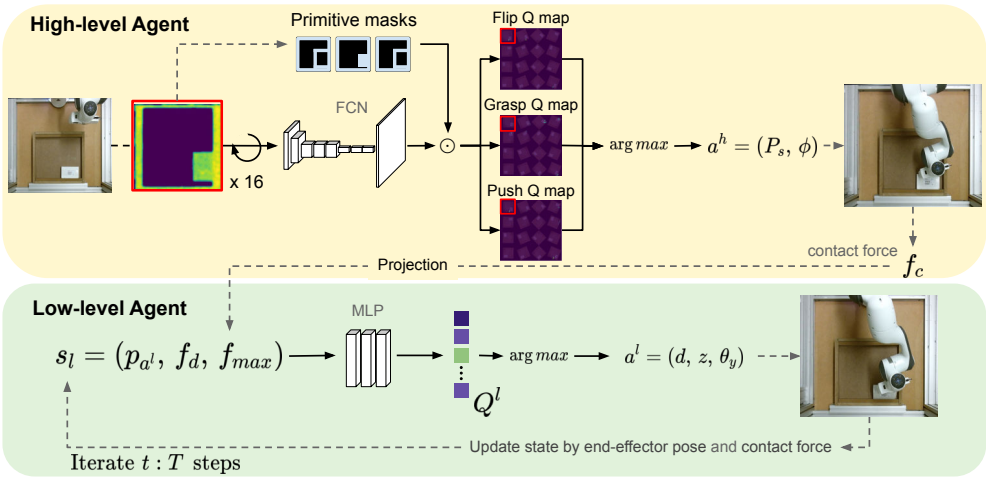

Figure 35: Overview. Our method aims to break down complex tasks into sub-tasks and reduce the need for manual primitive design. It comprises high-level and low-level agents. **High-Level Agent** (top): The high-level agent takes a height map as input to a DQN, implemented using an FCN model. It then outputs pixel-wise maps of Q values, where each pixel corresponds to a starting pose and a primitive. **Low-Level Agent** (bottom): The low-level agent combines the current end-effector pose and contact force as the state of a DQN model. It iteratively estimates a series of actions to accomplish the sub-task within a designated number of iterations T.

Low-level agent. In the low-level agent, the action space for the low-level agent is a vector of the state-action-value function Q. Within this vector, each value corresponds to a specific end-effector displacement (d, z, θ_y). In this context: $d \in \{0, a_d\}$ encodes the forward distance, $z \in \{0, a_z\}$ encodes the vertical movement, and $\theta_y \in \{-r_y, 0, r_y\}$ encodes the angular deviation along the Y-axis relative to the primitive's initial pose. The observation space of the low-level agent is a combination of the end-effector pose and the contact force. To facilitate efficient training, we develop an observation space designed to remain invariant to the starting pose of the low-level agent, ensuring state consistency across different initial poses. We design a reward function that considers the contact force and end-effector position:

$$r_\tau = r_t^{H_f} + r_\tau^L \tag{1}$$

$$r_\tau^L = \begin{cases} \min(\sigma, \frac{z_\tau \sigma}{w}) & \text{, if } f_c > 0 \\ -1 & \text{, if } f_c > f_{limit} \\ 0 & \text{, otherwise} \end{cases} \tag{2}$$

where f_c is the current contact force, f_{limit} is the maximum safety contact force, z_τ is the current end-effector height, and σ and w are hyper-parameters that normalize the z_τ and limit the upper bound of the reward. Based on this reward function, we encourage the agent not only to flip up the object but also to raise it up with contact and avoid applying too much contact force. We find that without the penalty for applying too much contact force, the robot may trigger emergency stops in the real world, making the transfer of policies learned in simulation more challenging.

3.4.3 Experimental results on flat object grasping

Evaluation in Simulation We train our method in simulation using flat objects of random friction, weight, and size. We randomly place an object in the workspace and evaluate whether the robot can pick it up, applying 10 primitives or less. We compare the training

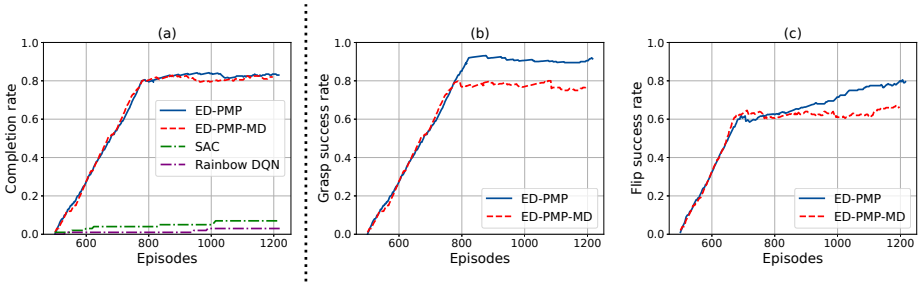

Figure 36: Testing curve of success rate versus training episodes of the high-level model in simulation. (a) The completion rate for full-task success. (b) The success rate for the grasp primitive. (c) The success rate for the flip primitive.

regime indicated above for our method with two baselines: SAC [44] and Rainbow DQN [45]. Additionally, we introduce an ablation — ED-PMP-MD — where the learned flip primitive is replaced by a manually designed version.

Figure 36 shows the learning curve of success rate versus training episodes. Our method performs better than SAC and Rainbow DQN. Compared to the ED-PMP-MD, despite the two agents using an *identical grasping primitive* implementation, our method achieves substantially higher grasping success rates. We speculate that this is due to the learned flipping primitive manipulating the object into configurations that better afford top-down grasps: a synergy that arises thanks to the simultaneous learning at different hierarchical levels.

Real-world Experiments To evaluate our method, we test it with zero-shot transfer from simulation to a real-world setup using the box-shaped objects in Table 6. We use four different boxes (Box-0 to Box-3) but consider two configurations of Box-0 (with and without additional contents to make it heavier), which means we have five object types.

Table 6: Target objects for the real-world experiments

				
Object-ID	Box-0	Box-1	Box-2	Box-3
Weight (g)	209 / 368	283	58	49
Length (cm)	14.2	19.9	10.4	15.1
Width (cm)	9.1	11.2	10.2	10.1
High (cm)	4.1	4.4	3.3	4.2

Table 7: Real robot experiments with 5 objects in different scenes across 10 episodes

Method	Scene	Box-0 (209g)	Box-0 (368g)	Box-1	Box-2	Box-3	**Average**
[42]	close	7/10	5/10	8/10	6/10	9/10	70%
[42]	random	4/10	1/10	3/10	3/10	5/10	32%
ED-PMP-MD	close	9/10	9/10	9/10	9/10	10/10	92%
ED-PMP-MD	random	9/10	7/10	10/10	9/10	9/10	88%
ED-PMP	close	9/10	9/10	10/10	10/10	10/10	**96%**
ED-PMP	random	10/10	9/10	10/10	10/10	10/10	**98%**

We consider two setups: *close*, where the object is placed next to one random wall of the box; and *random* where the object is placed randomly near the center of the workspace.

The results are shown in Table 7. We compare with Zhou and Held [42] as our baseline. Their method focuses on learning a controller that is able to flip an object and acquire an initially occluded grasping configuration. Our proposed method with the learned low-level flipping primitive clearly outperforms other baselines. Whether the object is placed close to a wall or randomly within the workspace, our method consistently attains high completion rates, ranging from 96% to 98%.

3.5 Deployment and real-world demonstrations of grasp pose estimation

For now, we have deployed the grasp pose estimation-based method on the perception manipulation module in the DARKO robot. We demonstrated this module at ARENA 2036 in Stuttgart (Figure 37) and at the Automatica 2023 fair (Figure 38).

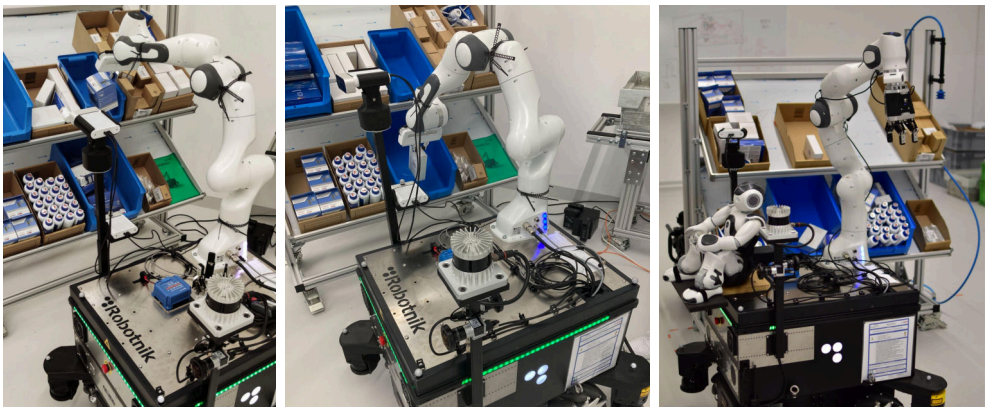

Figure 37: Demo of grasp pose estimation & picking at ARENA 2036 in Stuttgart.

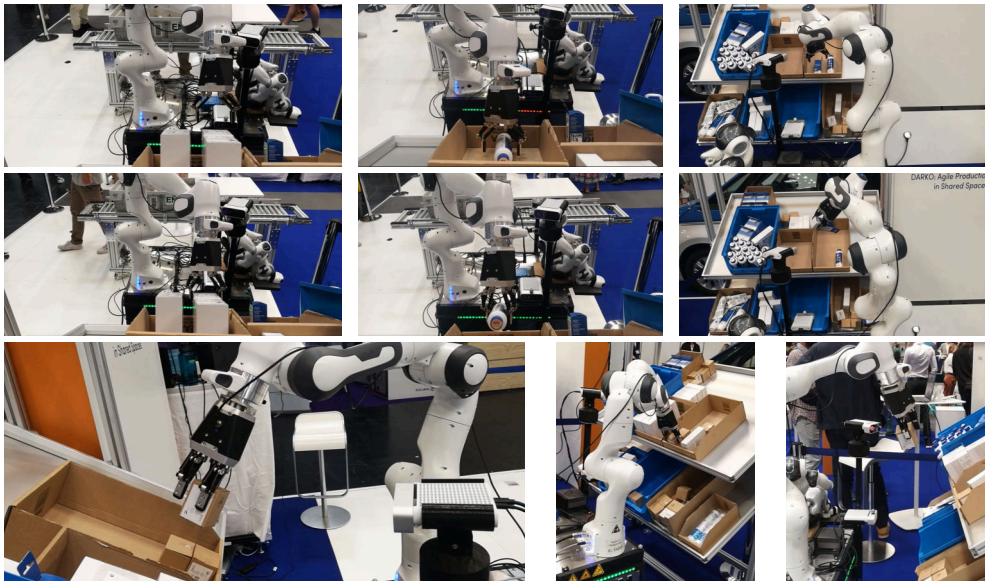

Figure 38: Demo of grasp pose estimation & picking at Automatica 2023 in Munich.

3.6 Remaining challenges and outlook for T2.2: Perception for manipulation

While we have achieved significant milestones in advancing task T2.2, encompassing the successful development of a direct grasp pose estimation method through point cloud regression and its effective application with a multi-finger soft hand on DARKO plastic bag objects, persistent challenges prompt further exploration.

In scenarios where securing a collision-free grasp is elusive, we have initiated an exploration of a manipulation primitives-based approach to address these complexities. Despite these efforts, several challenges visualized in Figure 26 (on page 28) remain open, including: handling more constrained picking conditions; managing cluttered object configurations within the container; and refining the success rate of picking transparent bag objects with a multi-finger soft hand. Therefore, work will continue to address these research questions. Furthermore, the imperative to test our method with a more diverse range of DARKO objects is evident.

As the next step, we will focus on enhancing the manipulation primitives-based approach by extending our proposed method [10] to ensure its versatility and efficacy in the face of diverse and intricate real-world scenarios.

4 T2.3: In-hand grasp perception

In this particular task, our focus lies on reconstructing the state of the gripper-object system post-grasping, with the primary aim of providing crucial insights into the throwing controller developed in T4.4. Given that T4.4 treats the grasped object as a singular point situated at the object's center of mass, understanding the intricate relationship between grasp configuration and the grasped object's center of mass, particularly the nuances of shifting between them, assumes paramount importance. This knowledge plays a pivotal role in facilitating the throwing controller's ability to make necessary adjustments, ensuring optimal performance and accuracy in the throwing process.

In the context of DARKO, we are targeting underactuated grippers, wherein all fingers are controlled by a limited number of joint controllers. The joints in each finger are determined through the intricate interplay between gripper, object, and the surrounding environment. This complexity makes predicting the gripper-object relationship prior to grasping a challenging task. To address this challenge, given the marked center position in object models, we first estimate the 6D pose of the grasped object and then calculate its current center position. A primary obstacle in this process is the accurate estimation of the grasped object's pose, a task complicated by occlusions caused by the robot's fingers and palm. This poses one of the main challenges we encounter within DARKO in this aspect.

Another crucial point related to the in-hand perception is the evaluation of the grasp stability. In fact, while the usage of a soft underactuated gripper permits handling the uncertainty related to the picking of an object in an easier way, the compliance of the end-effector fingers makes it difficult to predict the grasp configuration and consequently know in advance if the grasp could be successful. This problem could be partially tackled using a visual-based framework like the one we will present regarding the 6D in-hand object pose estimation. However, this type of solution is not feasible in scenarios similar to the one proposed in the DARKO project since it would require the gripper to always be in the camera's field of view while manipulating the object. To overcome this problem, the development of tactile-based solutions that exploit sensors directly embedded in the gripper becomes crucial to predict the failure of the grasp and consequently react by implementing some recovering routine able to prevent it.

Key contributions in T2.3

Our contributions for task T2.3, discussed in the next subsections, include:

- Testing of 6D object pose estimation involving novel objects with slight occlusion caused by a multi-finger hand.
- A framework able to: 1) predict grasp failure with soft robotic hands exploiting tactile information, and 2) implement a set of reaction strategies to avoid object slippage from the hand (published in "Advanced Intelligent Systems" journal [11]).
- An approach to generate the optimal manipulation patterns that maximize the information about a grasped object's inertial parameters and, as a consequence, to minimize their estimation uncertainty.

4.1 Object pose estimation for novel objects with slight occlusion

In this task, our objective is to determine the displacement between the object's center of mass and the grasp point, as shown in Figure 39. We achieve this by estimating the current object center position derived from its 6D pose. Utilizing the marked center position in the object model, once we ascertain the current object pose, we can accurately calculate

Figure 39: The objective in this task is to determine the displacement between the object's center of mass (red dot) and the grasp point (green dot) to make necessary adjustments for the throwing controller developed in T4.4.

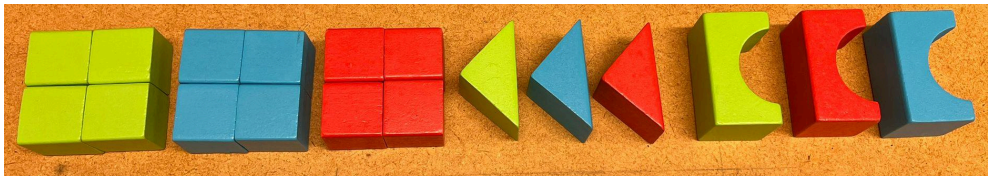

Figure 40: Simple geometry objects that we use in the experiments, including square, triangle, and semi-circle.

the position of its center of mass. This information allows us to estimate the precise shift between these key points. As our first step, we use simple geometry objects (Figure 40) with multi-finger hand in the experiments.

4.1.1 Method for object pose estimation of novel occluded objects

We employ Gen6D [46] to estimate grasped object pose. To do this, we first collect our object's reference images that include a series of different view RGB images. Then, we define object pose and recover camera pose in each image by COLMAP [47, 48]. An overview of this method is shown in Figure 41. The method comprises three steps: Firstly, a detector estimates the 2D bounding box of the object from both the current image and reference images. Secondly, a selector identifies the most similar reference image and adopts its pose as the initial estimate. Finally, a refiner utilizes the current image, the estimated pose, and neighboring reference images to calculate the pose residual, thereby further refining the estimated pose.

4.1.2 Real robot experiments

Experimental Setup As our first step, we use simple geometry objects (Figure 40) with minimum occlusion in the experiments. These objects include three different geometric shapes, square, triangle, and semi-circle, with three different colors. To better align the requirements of the DARKO setup, we use a multi-finger soft hand to grasp the object and estimate its 6D pose, as shown in Figure 42.

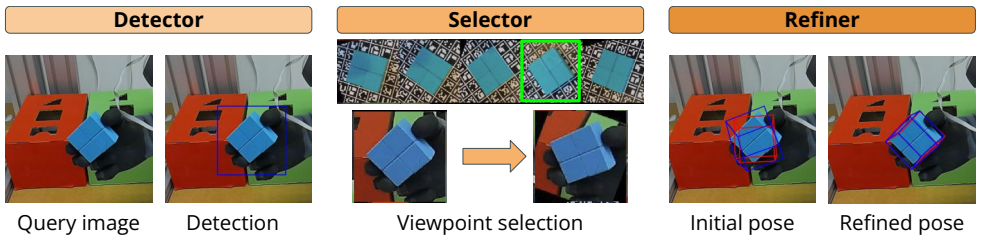

Figure 41: We employ Gen6D [46], a generalizable model-free 6-DoF object pose estimator, to predict the pose of novel objects. This method consists of three steps: (i) detection, where the object is identified in the image; (ii) selection, wherein the most similar view is chosen from reference images, and its pose is used as the initial estimate; and (iii) refinement, where the pose is further refined through regression to obtain an accurate object pose.

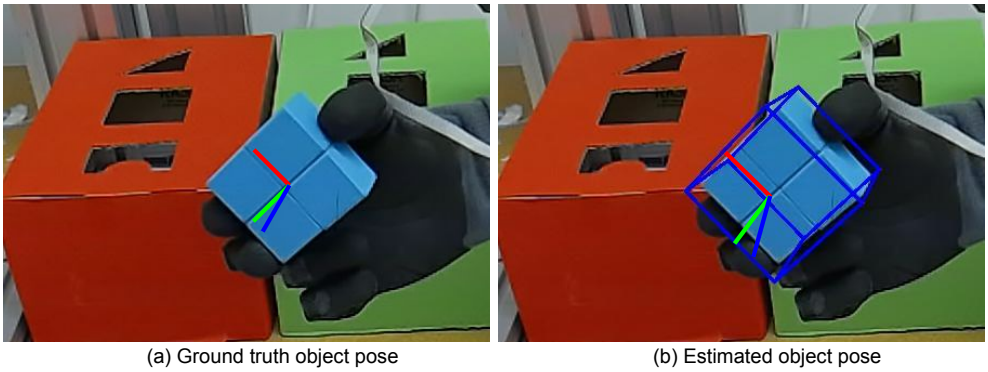

Figure 42: (a) Ground truth object pose. (b) Estimated object pose. The axes R, G, and B correspond to the x-, y-, and z-axes, respectively. The blue 3D bounding box is calculated by the estimated object pose and its model.

Experimental Results To evaluate the 6D object pose estimation method in our setup, we calculate the offset distance between the ground truth object center and the estimated center position which is calculated from the current object pose. We test each five times for each shape and report the average distance. In the square object, the average distance is 1.06 cm. In the triangle object, the average distance is 0.69 cm. In the semi-circle object, the average distance is 3.50 cm.

4.2 Learning to prevent grasp failure with soft hands: from on-line prediction to dual-arm grasp recovery

Soft articulated grippers simplify grasp planning by virtue of their intrinsic capability to adapt the fingers by exploiting the mechanical interaction with the environment. However, their usage poses new challenges in terms of sensing techniques. In fact, while rigid end-effectors' behavior during grasping can be easily represented, deformable grippers bring with them the complexity related to the modeling of hand-object interaction. To overcome these limitations, UNIPI proposes a method to combine distributed tactile sensing and machine learning (through a recurrent neural network, RNN) to detect sliding conditions for a soft robotic hand mounted on a robotic manipulator, targeting the prediction of the grasp failure event and the direction of sliding. The outcomes of these predictions allow for an online triggering of a compensatory action to prevent the failure. As proof of concept, we use a second manipulator to perform the compensatory action to avoid object

slippage. However, without losing generality, this sensing framework can be integrated to perform recovery actions in single-arm configurations. Despite the fact that the network was trained only with spherical and cylindrical objects, we demonstrate high generalization capabilities, achieving a correct prediction of the failure direction in 75% of cases, and a 85% of successful re-grasps, for a selection of twelve objects of common use. We will now describe the proposed approach in more detail.

4.2.1 Methods for soft-hand grasp failure prevention

The goal is to develop a closed-loop framework able to predict online the slippage direction of an object grasped by a soft robotic hand and use this information to trigger a reactive primitive which firmly stabilizes the grasp. To collect data for RNN training we used a 3D-printed object, composed of an interchangeable handle and a support where one or more masses were placed to modify the weight of the object. We considered two different shapes of handles, a sphere and a cylinder, which forced the shape of the hand into two different configurations. These were presented to the robot with two roughness levels, one smooth and one covered with sandpaper (400 Grit). The handle was grasped following two main approaches: top grasp, i.e. with the palm parallel to the horizontal plane, and lateral grasp, i.e. with the palm parallel to a vertical plane. For each of these grasp approaches, we further considered two potential failure types: central slippage, i.e. when the object slips along the long fingers, and lateral slippage, i.e. when failure is caused by a relative motion perpendicular to the long fingers (examples are provided in Figure 43-A/B). To avoid that the direction of gravity could affect the capability of the network to predict slippage direction, we included slippage data where the object was pulled off the SoftHand along a direction perpendicular to the gravity itself, by a second robotic arm-hand system (see e.g. Figure 43-A right; Figure 43-B left; Figure 45-B). The position and orientation of

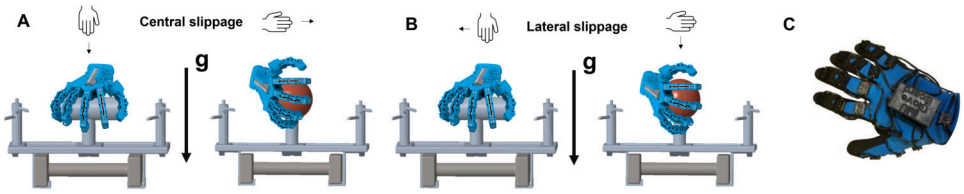

Figure 43: Experimental framework used for data collection. Two different handles (light grey in the figures) were used: a sphere and a cylinder. Handles were used with and without sandpaper coverage to modify the roughness. Panel A shows the two hand configurations that we implemented to replicate central slippage. Panel B shows the two hand configurations for lateral slippage. In both panels, also the gravity vector is reported, to show that not all failures occur along the gravity direction. The object was endowed with two supports for the markers of the 3D motion tracking system (one on the right and one on the left) designed with the shape of a star to always ensure the visibility of at least four markers. The variable weight (in the range of 200-700 grams) was attached to the handle (dark grey in the figures). Panel C reports a picture of the IMU glove we mounted on the SoftHand to continuously collect inertial measurements from each hand phalanx.

the hand and object during experiments were continuously tracked through a 3D motion tracking system (Optitrack Flex 13, NaturalPoint Inc., Corvallis, Oregon (US), refresh rate 120 Hz). The robotic hand that performed the reach-to-grasp task was endowed with a soft glove, on which we mounted 17 Inertial Measurement Units (IMUs), one for each phalanx, fastened on the back of the hand as in [49]. Four IMUs were attached to the thumb, and three to each long finger. One additional sensor was placed on the hand dorsum, close to the wrist, for reference (see Figure 43-C).

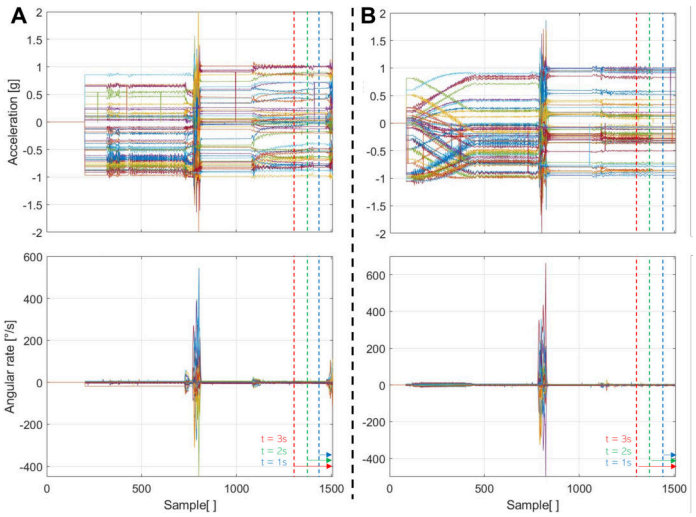

Figure 44: Stream of raw accelerometers (top row) and gyroscopes (bottom row) during a failed grasp (subplot A) and a successful grasp (B). Coloured dashed lines identify the initial frame of the block of signal that is removed from data before training, corresponding to one (blue), two (green) and three (red) seconds. We refer to this quantity as Δ.

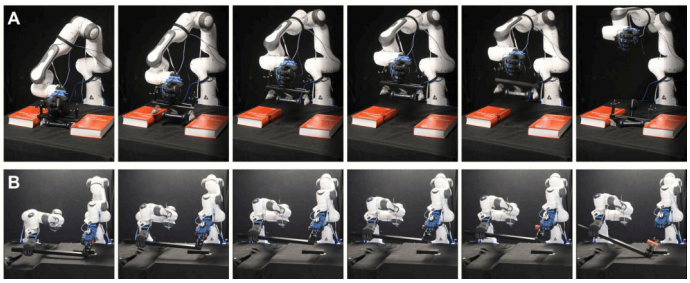

Figure 45: Two examples of failures events executed during data collection. A) Lateral slippage with a spherical smooth handle. B) Lateral slippage with a rough cylindrical handle, caused by the second robot.

Considering all the combinations discussed before, we performed a grand total of 1800 independent acquisitions, of which one-third was composed of successful grasps, one-third of central slippage and one-third of lateral slippage. More details about this procedure can be found in [11].

Once the dataset was built, to teach the network to recognise the event in advance, we removed from the dataset the final block of the signal, corresponding to a time slot immediately before the object drop. This has the twofold purpose of i) removing high peaks in the signal stream caused by the drop of the object and ii) learning to recognise small oscillations that are characteristics of failure events in the first frames of sliding, rather than larger oscillations evident in the final portion of the signal (see Figure 44). We tested three different levels of anticipation, corresponding to one, two and three seconds before the actual drop, reported in Figure 44 with a blue, green and red dashed line respectively. Hereinafter we will refer to the parameter quantifying this anticipation as Δ. Data were then randomized and split into three groups: 20% was devoted to testing and the remaining was further divided into 30% for validation and 70% for training. The neural architecture we selected is based on Gated Recurrent Units (GRU) [50], which are

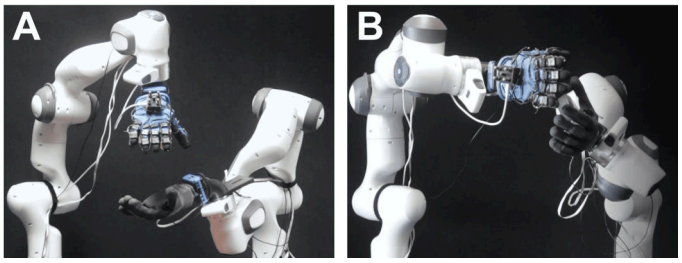

Figure 46: Re-grasp primitives considered for top grasps (A) and lateral grasps (B)

neurons with a feedback channel, which enables to store, and learn from, the history of a time series. The training was performed using an Adam optimizer and cross-entropy as loss function. Early stopping and dropout were also used to prevent overfitting. We tested different combinations of hyperparameters, resulting in three different architectures that demonstrated the highest validation accuracy and the minimum footprint of the network (to minimize inference time), one for each Δ value considered. This is motivated by the fact that the larger is the model the larger is the time to perform inference. Among these, we selected for the implementation the network trained with $\Delta = 2s$, because this provided a time horizon sufficiently large to eventually plan a recovery action while keeping high accuracy values over validation data.

To test the effectiveness of this sensing approach to trigger in real time a set of reactive strategies, we implemented a system using a second manipulator to stabilize the grasp performing different actions to avoid slippage. For further details regarding the reaction part, we refer the interested reader to [11].

We tested our online framework with two additional experiments. First, we replicated the failures with the same setup employed for data collection. We then considered a selection of 12 objects of common use, of which 10 are extracted from the YCB dataset [51] and two are L-shaped objects with smooth and rough surfaces (see Figure 48).

4.2.2 Results for soft-hand grasp failure prevention

As previously mentioned, we tested our framework in two different ways. First, we validated the network by assessing the prediction accuracy over a pool of test data not used during training and validation, consisting of 360 independent samples of three classes: successful grasp, central slippage and lateral slippage. Considering an anticipation time Δ of one, two, and three seconds, we converged to three optimized architectures, all based on GRU neurons. The optimal architecture with $\Delta = 1$ is composed of two layers of 64 neurons and was trained with a dropout of 0.3. This network demonstrated a validation accuracy of 0.93 ± 0.003 over ten different rounds of training (all starting from a random seed). For $\Delta = 2$, the optimal selection converged to a single layer of 128 neurons trained with a dropout of 0.5, achieving a validation accuracy of 0.91 ± 0.01 over ten different rounds of training (all starting from a random seed). Finally, with $\Delta = 3s$, the model consisted of two layers of 64 neurons, trained with a dropout of 0.3, yielding a validation accuracy of 0.87 ± 0.02 over ten different rounds of training (all starting from a random seed). After validation, we quantified the accuracy of prediction also over fresh data, not used during the training phase. This new dataset consisted of 360 samples, 120 for each class. Confusion matrices of the classification for different values of Δ are reported in Figure 47. We obtained an overall test accuracy of 87%, 84% and 76% for $\Delta = 1s$, $\Delta = 2s$ and $\Delta = 3s$ respectively.

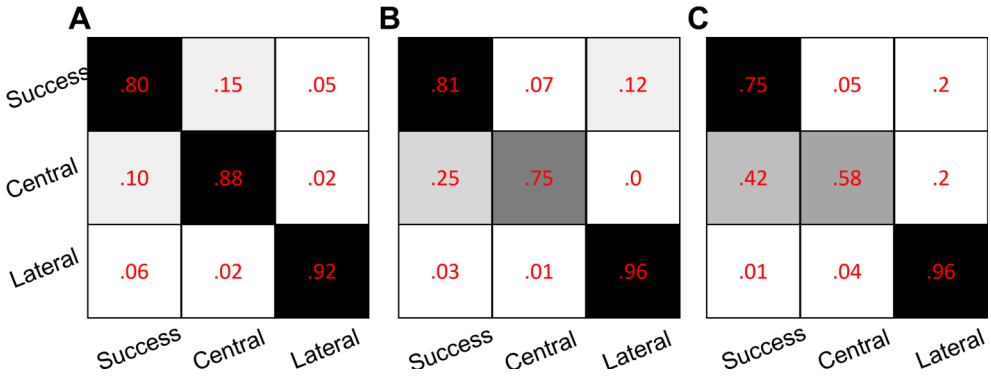

Figure 47: Confusion matrices for grasp failure classification on test data for $\Delta = 1$s (A), $\Delta = 2$s (B) and $\Delta = 3$s (C). Values are in percentage versus the total number of entries of each class. Rows denote the groundtruth class, columns the predicted class. Cells are color-coded: black stands for 100%, white stands for 0%.

In the second part, we implemented our network in an online integrated framework of failure prediction and reactive re-grasp. We decided to consider for the on-line implementation the architecture trained with $\Delta = 2$s, because this represents an appropriate trade-off between satisfactory prediction performances and capabilities of detecting small oscillations that are present in the early stages of sliding. As introduced in the Methods section, we performed two different experiments to test the capabilities of our framework in the online predict-and-regrasp task. In the first case, with a pile that continuously (with a refresh rate of 5 Hz) updates the stream of data given as input to the neural architecture, we obtained a correct classification in $\approx$ 78% of cases, of which $\approx$ 87% resulted in a successful robot regrasp. However, a correct prediction of the failure does not necessarily match with a successful re-grasp, because after triggering the reactive primitive the robot could spend a certain amount of time to plan and execute the trajectory (approximately 1.5 seconds). For this reason, in certain cases, especially with very smooth objects, the total success rate of the re-grasp could be lower, and it is more appropriate to report on the number of successful regrasps over the ones correctly predicted. Indeed, in this first experiment we had that for occurrences of central slippage we were able to successfully prevent the failure in 80% of cases, while the performances increased to 94% for lateral slippage. This is caused by the fact that in the first case, the time of sliding is shorter, on average, than the second one.

We further validated our framework by performing a second experiment with 12 objects, of which 10 are extracted from the YCB dataset [51] (see Figure 48). Also in this case, we verified the prediction and classification accuracy and then quantified the success rate of the failure prevention over the cases in which we were able to successfully predict the failure. Over a total of 130 experiments, we achieved a classification accuracy of 75% and, for the cases in which the type of failure was predicted correctly, we successfully prevented the failure with our reactive primitive in 85% of cases.

4.3 Simultaneous perception and manipulation

UNIPI has started an investigation of shared control policies to allow soft collaborative end-effectors to autonomously and haptically *explore* the grasped objects, to retrieve important features that allow for an effective autonomous grasping and manipulation – i. e. simultaneous perception and manipulation of unknown objects. The sense of touch is intrinsically an active sense, i. e. motion and perception are two sides of the same coin:

Figure 48: Objects of common use selected for the experiments: a sauce bottle, an apple, a tennis ball, a squeeze tube, a mug, a box, a saucepan, a water bottle, a food box, a shampoo bottle, a rough l-shape, a smooth l-shape. We increased the weight of some of these objects by adding external weight (as done during experiments for dataset collection) to match the range between 200 and 700 grams.

for this reason, it could be useful to deal with robotic perception and manipulation in an integrated fashion. Indeed, as we can borrow from the active sensing literature applied to vehicles locomotion [52, 53], that the motion itself could be valuable source of information on the system itself. In a nutshell, the goal is to identify the optimal movements that a generic object should follow to maximize the accuracy and minimize the uncertainty for the estimation of its unknown mechanical properties, such as its center of mass and inertia tensor. We will now describe the proposed method in more detail.

4.3.1 Method for simultaneous perception and manipulation

The main goal here is to maximize the information on the inertial parameters of a grasped object. To do so, we can model our generic object using the following terms:

- the state π, which needs to be estimated; it contains the mass, the three spatial center of mass terms and the six independent elements of the inertia tensor;
- the vector $q(t) = \left[v, r \right]^T$, which contains the linear and angular velocities v in body frame and spatial orientation r of the object expressed as a quaternion;
- finally, w_{ext}, i.e., the vector of external wrenches acting on the object, expressed in body frame.

Choosing an output equation $y(t) = h(\pi(t), q(t))$, we can obtain the linearized output matrix $C(t) = \frac{\partial h(\pi)}{\partial \pi}$, which depends only on the angular velocities $\omega(t)$ and angular accelerations $\dot{\omega}(t)$, expressed in body frame. So, $C(t) = C(\omega(t), \dot{\omega}(t))$.

To minimize the uncertainty on the unknown state π, we need a metric that encodes the information collected over a time interval $[t_0, t_f]$. For our purposes, the best metric is *constructability*, i.e., the property of determining the final state $q_F = q(t_F)$ from knowledge of present and future system output $z(t)$ and input $u(t)$ over $[t_0, t_f]$. As shown in [53], an effective metric for this problem is the Constructability Gramian (CG) $\mathcal{G}_c(t_0, t_f)$.

To estimate the unknown parameters, an Extended Kalman Filter (EKF) was exploited. In [53] it was demonstrated that the covariance matrix $P(t)$ for the estimation error of the EKF is linked to the CG at time t by the relation $P^{-1}(t) = \mathcal{G}_c(-\infty, t)$. So, maximizing the minimum eigenvalue of the CG corresponds to the minimization of the maximum state uncertainty of the EKF.

Figure 49: Experimental setup for center of mass estimation: the T-shaped object is fixed to the force/torque sensor linked to the robot end-effector.

Hence, the goal is to optimize the input $u(t)$ of the system (in this case, the angular velocities $\omega(t)$ in body frame), such that:

$$u^*(t) = \arg\max_{u(t)} \lambda_{min}(\mathcal{G}_c(-\infty, t_f)) \tag{3}$$

To express the CG in a simple way, the first assumption is that the mass value is known beforehand. Indeed, there are two ways of estimating the mass of a generic load: static or dynamical methods ([54]). The first category takes only gravity into consideration and leads to very good results in terms of estimation error. Moreover, a lot of these methods can also estimate the bias on the force/torque sensors. Instead, dynamical methods consider the inertial behavior of the object. They are not as precise as the static methods (as shown in [55]), so we can assume to perform a static estimation of the mass and estimate the remaining 9 parameters.

Then, it is possible to parametrize each of the 3 components of ω by a family of curves functions of a finite number of parameters. A particular parametrization that satisfies these requirements consists in dividing the total interval $[t_1, t_n]$ into n time segments $[t_j, t_{j+1}]$, each with duration $\Delta_j = t_{j+1} - t_j$. Then, ω_k can be expressed in the $j-th$ interval as a truncated cosinusoidal function $s_j(t)$ with amplitudes A_j and A_{j+1} at times t_j and t_{j+1}, respectively:

$$s_j(t) = \frac{A_j - A_{j+1}}{2} \cos\Big(\frac{\pi t}{\Delta_j}\Big) + \frac{A_j + A_{j+1}}{2} \tag{4}$$

The resulting ω_k will be a continuous function with a continuous derivative $\dot{\omega}_k$ in the entire interval $[t_1, t_n]$, with a null derivative at each interval extremity t_j. Moreover, the function s_j is always limited between the control points A_j and A_{j+1} so that overshoots are totally avoided. In this way, the optimization routine will find the optimal amplitude of the control points and the optimal segment duration for each time segment.

4.3.2 Preliminary results on simultaneous perception and manipulation

Here we report the preliminary results of our method for the estimation of the center of mass in an experimental scenario.

For the experimental setup, we 3D-printed a T-shaped object, which was then filled with materials of known mass and density. The object was then attached to the end-effector in a form closure configuration (see Figure 49). In between, a force/torque sensor, i.e.,

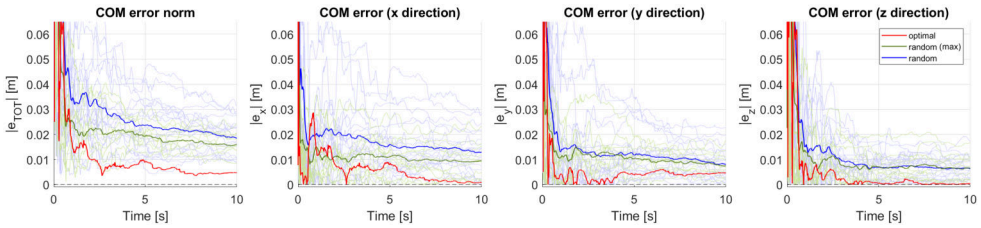

Figure 50: Center of mass estimation error.

a Schunk F/T Sensor Mini 45, was used to collect wrench data. The item was moved through a 7DoF redundant robot manipulator, namely a Franka Emika Panda. For the experiments, we considered 20 random movements, and compared them to an optimal movement computed from an initial guess with null control points. Moreover, we compared the optimal movement with other 10 random movements with similar control energy. Parameters used for the experiments are m = 0.334 kg, c = [-0.004, 0.045, 0.02] m.

In Figure 50, we report the results for the estimation of the center of mass, considering the single errors in the three spatial directions and the whole error norm computed as the distance between the real center of mass and the estimated one. The optimal movement leads to the minimization of the center of mass error norm, which is lower at the final time than the mean of the random movements (in blue) and the random movements with comparable input energy (in green). The error along the three directions considered separately also shows the advantage of the optimal movement, both in terms of error at the final time and of time convergence to a certain error band.

4.4 Remaining challenges and outlook for T2.3: In-hand grasp perception

In task T2.3, we have commenced the evaluation of the employed object pose estimation method, with a primary emphasis on simple geometric objects grasped by a multi-finger hand. Our progress is a foundational step in understanding the complexities introduced by DARKO objects, particularly those with varying degrees of occlusion caused by the hand.

To further advance our methodology, the next phase is going to involve the extension of our current method to address challenges posed by a broader spectrum of DARKO objects, each presenting unique characteristics and levels of occlusion. This expansion necessitates a comprehensive data collection effort specifically tailored to DARKO objects. Concurrently, we aim to refine the estimation performance of our method and bolster its capability to effectively tackle the challenges introduced by occlusion.

5 T2.4: Detecting successful throws

DARKO's work package WP4 (specifically, T4.4) develops methods for throwing objects, rather than placing or dropping them, into target containers. Task T2.4 provides the perception capabilities necessary for assessing the result of such throwing actions. As accurate throwing is a challenging task, it is very important that the throwing system can monitor the success of executed throws: i. e., did the object successfully reach the goal or not. The output of this module can in turn be used to assess and tune the performance of the throwing sub-system of the DARKO robot.

5.1 A trajectory-centric approach to detection of throw success

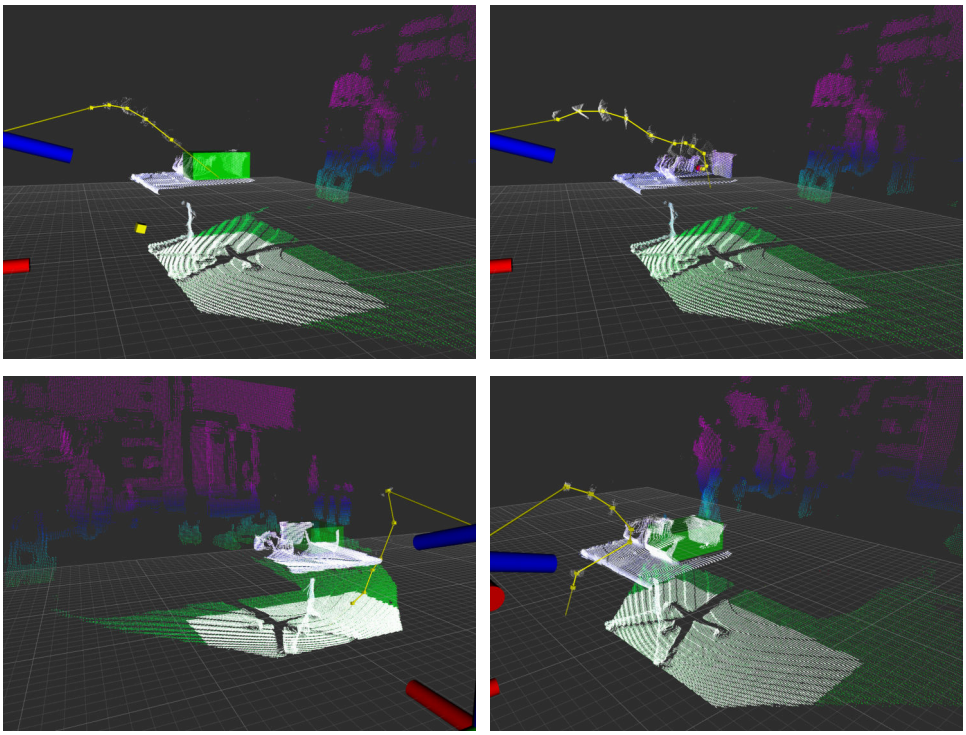

Figure 51: Example visualisations of the throw assessment module in T2.4. The colored point cloud shows the current frame from the top RGB-D camera. The white point cloud is an aggregate of the last 10 seconds of RGB-D frames. The tracked object is shown with a yellow path. The bounding box of the target tray is shown as a transparent green box. The detected landing point (if successful) is shown as a red ball. *Top row:* Successful throws. In the top right example, the object bounces once but stays inside the container.) *Bottom row:* Missed throws. The bottom left example shows a light bag that does not follow a parabolic path, due to air drag; and the bottom right example shows a throw where the object hits the edge of the target tray and bounces out.

As anticipated in the Grant Agreement, we have followed a trajectory-centric approach, estimating the landing location of the thrown object relative to a target bin, as is visualized in Figure 51. In our experiments, the frame rate of the onboard sensors, including the Azure Kinect RGB-D cameras, has been found to be high enough to estimate the trajectory from direct observations in practice. Therefore, we have not pursued the alternative path

(a) (b) (c) (d) (e) (f)

Figure 52: Test objects thrown to verify T2.4.

of incorporating physics models or generative trajectory estimation models. A straightforward segment-and-track procedure gives good results within the throwing distances that are relevant in the DARKO use case, as will be shown later in Table 8.

We assume that the object is thrown such that most of its trajectory is observable within the field of view of one of the Azure Kinect RBG-D cameras. The top camera of the robot platform (see D1.1), mounted on a pan-tilt motor, should be directed to look along the intended throwing direction. Further, we read the 9DOF bounding box of the *target* container from the object-level semantics module (T2.1).

First, the software module listens for activation by a ROS `std_msgs/Bool` message from the overall task manager. After activation, the first point cloud received from the RGB-D camera is stored as background. For subsequent point clouds, the background is subtracted, keeping only points that have no neighbours closer than $\delta = 1$ cm in the background. The remaining foreground points are clustered with Euclidean clustering, using a $t = 2$ cm distance threshold and minimum cluster size $m = 25$ points. Tracking stops once five consecutive frames without foreground points have been observed. The throw-tracking module also subscribes to a list of all objects currently detected from T2.1 and specifically looks for objects with the label “target tray”. The “target tray” object that is closest to the center point of the last observed foreground cluster is selected as the target container for the throw. The final landing point is estimated by ray casting from the last observed point, checking for ray–box intersections with the target tray, to account for cases where the final landing point inside the target tray is not observed. Figure 51 shows example output from Rviz, visualising the process.

The output of the module is the estimated landing position of the object relative to the target’s center point, along with a Boolean flag to signal whether or not it is located inside the target container.

5.2 Experiments and results on detection of throw success

To verify and quantitatively test the module for detecting throw success, we have recorded a data set with a sequence of manual throws at ORU. We have used five types of DARKO objects, shown in Figure 52. The objects were thrown towards a target box in two configurations: either 2 m from the robot base (circa 1.25 m from the reach of the arm) or 4 m away (3.25 m from the reach of the arm). These configurations were selected to be challenging both for throwing and for detecting the throw success.

Table 8 lists the results of these tests. For the *short* configuration, the success rate is 100%. For the *long* configuration, the success rate is $8/12 = 67\%$. However, please note that the *long* configuration is more challenging than what is to be expected in DARKO’s main use case.

Test no.	Distance	Object	Detected result	True result	Success
1	short	Tassimo (52a)	in	in	✓
2	short	Tassimo (52a)	out	out	✓
3	short	Microfilter (52b)	in	in	✓
4	short	Microfilter (52b)	in	in	✓
5	short	Maschinenpfleger (52c)	in	in	✓
6	short	Maschinenpfleger (52c)	in	in	✓
7	short	Reinigungsgel (52d)	in	in	✓
8	short	Reinigungsgel (52d)	in	in	✓
9	short	Pumpensumpf (52e)	in	in	✓
10	short	Pumpensumpf (52e)	in	in	✓
11	short	USB caps (52f)	out	out	✓
12	short	USB caps (52f)	in	in	✓
13	long	Tassimo (52a)	in	in	✓
14	long	Tassimo (52a)	out	in	×
15	long	Microfilter (52b)	out	out	✓
16	long	Microfilter (52b)	out	out	✓
17	long	Maschinenpfleger (52c)	out	out	✓
18	long	Maschinenpfleger (52c)	in	in	✓
19	long	Reinigungsgel (52d)	out	in	×
20	long	Reinigungsgel (52d)	out	in	×
21	long	Pumpensumpf (52e)	out	out	✓
22	long	Pumpensumpf (52e)	out	out	✓
23	long	USB caps (52f)	out	out	✓
24	long	USB caps (52f)	in	out	×

Table 8: Success rate of the T2.4 module for detecting throw success.

5.3 Summary of achievements and outlook for T2.4: Detecting successful throws

The results shown above (section 5.2) indicate that the throw detection module will be useful for training and testing of the throwing modules from WP4. It will be put to more use during the final reporting period of DARKO, in cooperation between ORU and EPFL.

The tests reported here were done with manually annotated bounding boxes for the target container, the activation was triggered manually and the output pose estimate and success flag were not used by the task planning or throwing modules. For upcoming system demonstrations, we will fully integrate it with live object detection from T2.1.

Future iterations of the object detection module in T2.1 are expected to detect target containers also from fish-eye images. We have made initial experiments on data from the fisheye cameras, which provide a much wider field of view than the RGB-D cameras but do not provide any point cloud data. In mostly static scenarios, a frame-by-frame difference filter on the RGB fisheye images can be used to segment the object from background while in-air. In the current implementation, detection of throw success using the fish-eye cameras depends on a manually specified target location and only provides a binary success/failure label, given the lack of depth information.

6 T2.5: Perception of humans and their poses

The fifth and final task of the perception work package WP2 deals with the real-time 3D detection and tracking of humans and their articulated 3D poses from the egocentric perspective of a mobile robot and its onboard sensors, thereby addressing DARKO's objective O2 on efficiency in human-robot co-production and enabling further downstream tasks including building maps of dynamics (T3.3), prediction of human motion and intents (T5.1), communication with human co-workers (T5.2) and reasoning about human-robot spatial interactions (T5.3), which serve as inputs to safe and risk-aware control (T4.2) and local and global motion planning (T6.1, T6.3).

A baseline human perception system, based upon existing components from the ILIAD project, had previously been integrated and deployed on the DARKO robot as part of the preceding deliverable D2.1; in that software deliverable, a Docker-based architecture was introduced to easily deploy the human and object perception system from T2.1 and T2.5 across different robots, thus thereby addressing objective O3 on easy deployment.

In the current deliverable D2.2, the goal has been to advance research and development efforts in perception of humans along the following two directions:

1. *Combining real-time 3D human detection, articulated 3D body pose estimation and tracking on a resource-constrained mobile robot:*

   In previous work, together with RWTH Aachen University, Bosch had developed a monocular, single-frame 3D human pose estimation approach called MeTRAbs, which had been evaluated qualitatively and quantitatively on several academic datasets in an offline fashion. As part of this deliverable, our goal now was to integrate it with the rest of the online perception stack (including the RGB-D YOLO 3D detector and tracking) and deploy the method in real-time on our mobile robot platform, taking resource limitations of embedded compute devices into account and employing model pruning and optimization techniques where necessary. Also, per-frame body pose estimates need to be integrated over time to gracefully handle temporary occlusions by other humans or objects, which are observed frequently in a first-person egocentric perspective.

2. *Maximizing observable field of view with a minimal set of sensors:*

   In practice, downstream applications often require or benefit from a surround-view, 360-degree human perception without large blind spots around the robot. In fact, several DARKO partners in WP3, WP5 and WP6 expressed the need for coverage of a larger field of view than can be obtained with a single Kinect-like RGB-D sensor. Therefore, one focus at the beginning of the project was to examine which sensor-detector combinations work well for human detection in industrial scenarios, taking observable field of view and runtime considerations into account.

   Looking next at the more fine-grained task of *articulated 3D human pose estimation*, it can be noted that existing approaches from the literature are mostly limited to the relatively narrow field of view of a rectilinar pinhole color camera or an RGB-D camera ($\sim 90-120^\circ$ horizontal FOV). One way of extending the field of view could be to combine multiple pinhole cameras in an array, but this would introduce challenging discontinuities at sensor boundaries and lead to higher sensor cost. An alternative method, which we are exploring in this sub-task, is to leverage fewer cameras with fisheye lenses ($\geq 180^\circ$), and/or a high-resolution 3D lidar for articulated 3D body pose estimation with wide field of view, and extend existing 3D human pose estimation approaches in this direction while using transfer learning and augmentation techniques to reduce the required amount of extra training data.

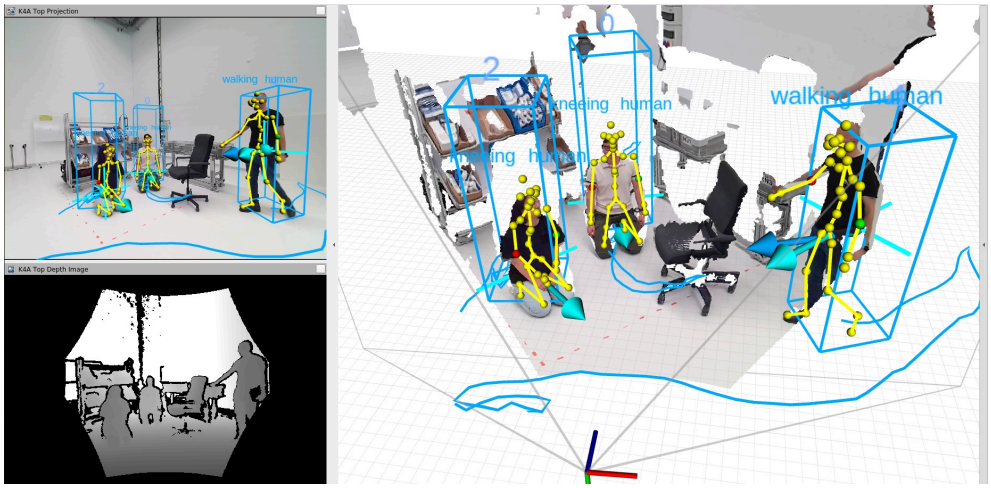

Figure 53: Visualization of the integrated human perception stack that was deployed on the DARKO robot during the MS2 milestone demonstration at ARENA 2036. A single Azure Kinect RGB-D camera was used for human perception during this milestone, its color and depth images are shown on the left. Bounding boxes have a fixed scale and indicate tracked persons, trailing blue lines their past trajectories, blue arrows their velocity vector. Cyan crosses represent oriented 3D centroids from the RGB-D YOLO detector.

Key contributions in T2.5

Our contributions for task T2.5, described in the following subsections, include:

- *A systematic cross-modal comparison of human detection approaches for robotics* conducted by Bosch at the beginning of the project as a continuation of work started in ILIAD. In this case study, which focused on industrial application scenarios and got published at IROS 2021, we examined the pros and cons of different sensor-algorithm combinations (for 2D lidar, 3D lidar, RGB-D cameras), taking also observable field of view and runtime performance into account.
- A *fast ONNX and TensorRT version* of the existing RGB-D YOLO 3D human detector that runs efficiently on an Nvidia Jetson AGX Orin embedded device (with up to +230% speed increase and -70% reduction in memory usage). This major improvement, which has been deployed on the DARKO robot as part of the MS2 milestone, is crucial for parallel deployment along with other DARKO perception modules, e.g. the 3D human pose estimation module, or object-level semantics from T2.1.
- A complex *multi-modal, multi-view recording setup* leading to a *novel dataset for 3D human pose estimation with heterogeneous wide-FOV sensors* (including different fisheye cameras and 3D lidar) to enable further research on this topic; as well as an initial baseline for fisheye-based 3D human pose estimation that extends the top-down MeTRAbs approach with a virtual pinhole reprojection stage.
- An *efficient Transformer-based approach for short-term 3D human body pose prediction* to improve temporal accuracy of skeleton estimates over time especially during temporary occlusions. The extensive results we have obtained are currently being prepared for submission to a conference.
- A prototype of a method for multi-class *human activity classification*, based on input 3D human skeletons, skeletal features, velocities from human tracking and temporal

smoothing. Due to using skeletons as an input representation, the method is very data-efficient and can be trained on new activity classes using just a few short recorded example sequences. It was demonstrated live on the robot during the DARKO MS2 stakeholder meeting.

Figure 53 shows a visualization of the real-time human detection, tracking, pose estimation and classification modules that were deployed on the DARKO robot during the milestone MS2 demonstration and stakeholder meeting.

6.1 A cross-modal comparison of human detection approaches for industrial robotics applications

Advances in 2D and 3D sensing and deep learning-based methods have led to increasingly mature solutions for human detection by robots, particularly in selected use-cases such as pedestrian detection for self-driving cars or close-range person detection in consumer settings. Despite this progress, the simple question *which sensor-algorithm combination is best suited for a person detection task at hand?* remains hard to answer. In our IROS 2021 paper[4] [12], we closely examined this issue by conducting a systematic cross-modal analysis of sensor-algorithm combinations typically used in robotics. We compared the performance of state-of-the-art human detectors for 2D range data, 3D lidar, and RGB-D data as well as selected combinations thereof in a challenging industrial use-case, taking additional aspects such as runtime performance and observable field of view into account.

Experimental Results: As shown in Figure 54, our results indicate a large variance among the different approaches in terms of detection performance, generalization, frame rates and computational requirements: In general, we find that recent state-of-the-art detectors, which have been developed and tested on other domains (e.g. automotive, elderly care facilities), are also the top-performing ones in our industrial target domain. In particular, strong RGB-D methods (such as RGB-D YOLO) work well even when learning 3D localization just from synthetic data, and have no problems with detecting persons e. g. in unusual protective clothing if pre-trained on large-scale, real-world 2D image datasets such as MS COCO. Instead, 3D lidar-based approaches show a larger domain gap that can be mitigated by retraining on data from the target domain: The domain gap that exists *without* any domain adaptation suggests that currently available public datasets for 3D lidar are too small or not diverse enough to make deep learning approaches generalize well to robotics scenarios.

Weakly Supervised Cross-Modal Transfer Learning: Here, we therefore propose a simple, yet effective multi-sensor transfer learning strategy that leverages our strong image-based RGB-D YOLO detector to provide *cross-modal supervision* for 3D lidar detectors in the form of weak 3D bounding box labels, as visualized in Figure 56. To this end, in DARKO, we extend our earlier work from ILIAD [14] with the capability to estimate not only 3D centroids, but also oriented 3D bounding boxes (see extended network architecture in Figure 55). We then show that a 3D lidar detector *weakly* supervised by such 3D bounding boxes from RGB-D YOLO is able to, on real-world test data from the robotics domain, outperform the same method trained in *fully* supervised manner on automotive data. This highlights how important it is to consider the often neglected domain gap between these two application scenarios. The outperformance of our weakly supervised approach is even more astonishing given the fact that the RGB-D YOLO teacher used in our experiments has

[4]This work has been partly conducted within the preceding ILIAD project (around 60%) and was finished in DARKO. Paper writing and cross-modal weak supervision transfer experiments were performed in DARKO.

Modality	Method and training data (if not specified: proprietary training set)	Horiz. FOV °	FPS (Hz)	VRAM (MiB)	AP[0.5m] ↑ in %			Peak-F1 ↑		
					(a)	(b)	(a)+(b)	(a)	(b)	(a)+(b)
2D laser	Arras [22] as in [11] (DROW)	180	25	-	56.0	47.4	50.7	0.58	0.60	0.59
	Leigh [23] as in [11] (DROW)		22	-	67.4	84.6	72.2	0.71	0.87	0.75
	DROW3x, T=1 (DROW) [11] + NMS [24]		49	827	72.7	91.3	78.4	0.72	0.88	0.77
	DR-SPAAM (DROW) [24]		48	829	74.4	91.2	79.8	0.75	0.88	0.79
3D lidar	Object3D SVM Detector [21]	360	8	-	77.1	12.6	52.6	0.83	0.24	0.63
	PartA2-Net (KITTI) [29]		7	5183	87.8	67.0	83.5	0.89	0.64	0.79
	PointPillars (KITTI) [28]		11	1217	77.5	72.8	76.3	0.83	0.75	0.80
	SECOND (KITTI) [26]		9	4999	82.3	87.5	84.1	0.86	0.82	0.85
	SECOND-DV (KITTI) [27]		9	4979	82.9	90.4	86.0	0.88	0.87	0.88
	SECOND-DV (ILIAD), transfer learning		9	5025	94.4	97.9	93.9	0.88	0.94	0.87
RGB-D (Kinect v2)	PCL + HOG-SVM [35]	86	21	-	70.7	67.9	69.6	0.73	0.71	0.72
	Mobility Aids [36]		23	1576	54.4	52.6	54.4	0.71	0.68	0.70
	RGB-D Pose 3D [37]		1	4055	60.3	92.8	71.4	0.75	0.95	0.82
	YOLO v3, naïve depth (COCO) [34]		28	1081	68.7	93.1	74.7	0.80	0.92	0.84
	RGB-D YOLO (COCO + synthetic 3D) [1]		24	3269	91.1	99.0	93.5	0.92	0.99	0.94
2D, 3D, RGB-D	✚ Detection fusion DR-SPAAM + SECOND-DV (ILIAD) + RGB-D YOLO	360	9	9123	-	-	-	0.87	0.92	0.89
3D, RGB-D	✚ Detection fusion SECOND-DV (ILIAD) + RGB-D YOLO	360	9	8294	-	-	-	0.91	0.96	0.93
3D, RGB-D	✚ Detection fusion Same, but fuse 3D lidar only outside RGB-D FOV	360	9	8294	-	-	-	0.92	0.99	0.94

Figure 54: Quantitative results of our cross-modal detection comparison with different sensor-algorithm combinations for human centroid detection. (a) and (b) denote two different evaluation scenarios (a larger open-space scene, and a narrow and cluttered storage room). This work [12], conducted at the beginning of the first period of DARKO, is a conclusion to preceding work from the ILIAD project.

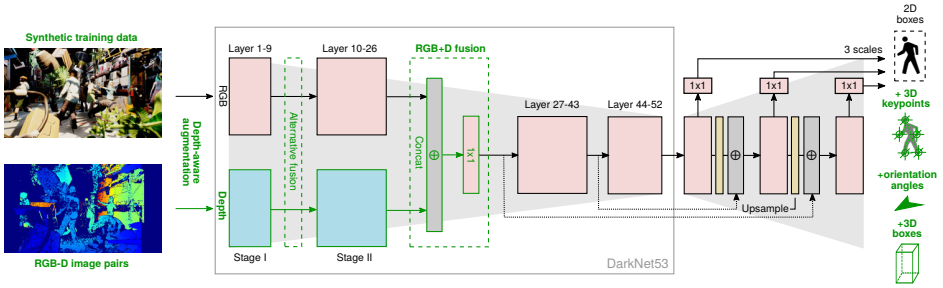

Figure 55: RGB-D YOLO architecture with oriented 3D bounding box regression [12]. This is an extension of our previous work from the ILIAD project [14], which is used for cross-modal supervision transfer as shown in Figure 56.

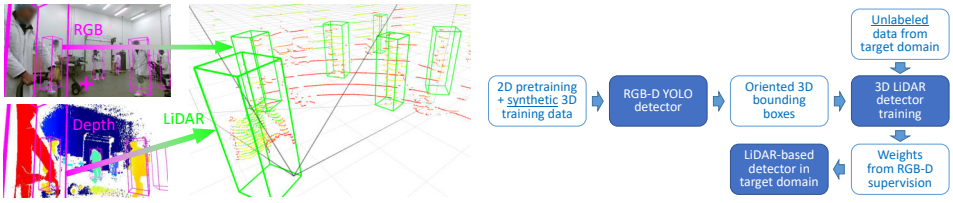

Figure 56: Weak label supervision transfer from the RGB-D YOLO detector, extended to provide oriented 3D bounding boxes, to a SECOND detector operating on unlabeled 3D lidar data from the robotics target domain [12]. This allows us to easily outperform an existing detector trained on automotive data without extensive need for manual 3D labeling in the target domain.

itself learned its human 3D localization capabilities purely from *synthetic* data (generated using Unreal Engine 4), i. e. it never saw any real-world 3D annotations.

More technical details, as well as recommendations to the practitioner for deployment of robots with human detection capabilities, can be found in our IROS 2021 paper [12].

6.2 Efficient deployment of human detection on embedded hardware

One particular challenge to be overcome by the DARKO consortium during the second period, which was already anticipated and highlighted by the project's reviewers during the first review meeting, was the combined real-time deployment of many different computationally demanding algorithms (for perception, prediction and planning) in parallel on the DARKO robot. As a mobile platform, it is limited in its computational resources, therefore computational efficiency of individual components becomes more important once these components are not only tested in isolation anymore. During the second period, we have therefore worked on making the computationally most perception component in T2.5, the human detector, more efficient.

6.2.1 Integration of an Nvidia Jetson AGX Orin embedded compute device on the robot platform

The DARKO robot platform was shipped by the manufacturer with the following integrated compute devices:

- An Intel i7-based platform without GPU (the default "robot PC") used for low-level control, communication with the safety PLCs, navigation, running ROS Melodic;
- An industrial PC (Neousys Nuvo-8108GC) with Nvidia RTX 3060 GPU (12 GB), requested at project start by WP2 for perception tasks, running ROS Noetic.

During the course of the project, it became apparent that more and more components developed by the consortium would require the newer ROS Noetic version. However, upgrading the robot PC from ROS 1 Melodic to Noetic was not a viable option due to the robot manufacturer's lack of support for that version (which means that low-level drivers for motor controllers, LED stripes, safety and so on were not provided for that version).

Subsequently, it was therefore decided by the consortium to deploy any components that would require ROS Noetic on the Nuvo PC. This includes additional components, e. g. from WP4, which were originally not planned to be deployed there, leading to a higher resource utilization than anticipated. Another challenge noted during period 1 was that the combined GPU memory usage of all WP2 deep neural networks alone would exceed the available 12 GB of VRAM available on the Nuvo platform.

Therefore, Bosch took the initiative to install an additional embedded compute device, an Nvidia Jetson AGX Orin (32 GB), onto the robot platform stationed at ARENA 2036. For this, a custom mounting structure had to be designed and 3D printed. For easier accessibility (e.g. to allow for the possibility of directly connecting some sensors), it was decided to install the Jetson on top of the robot's base. An additional 48V-to-12V DC-DC converter had to be added. Mounting structure, power supply and Jetson on top of the robot are shown in Figure 57 (left).

6.2.2 ONNX and TensorRT versions of RGB-D YOLO for 3D detection

The best-performing human detector in our IROS 2021 comparison was the RGB-D YOLO method, which we had introduced earlier during the ILIAD project [14] and which had been developed using the Apache MxNet and GluonCV deep learning frameworks. Back then, at inference time we used the C++ bindings of MxNet that need to be compiled from

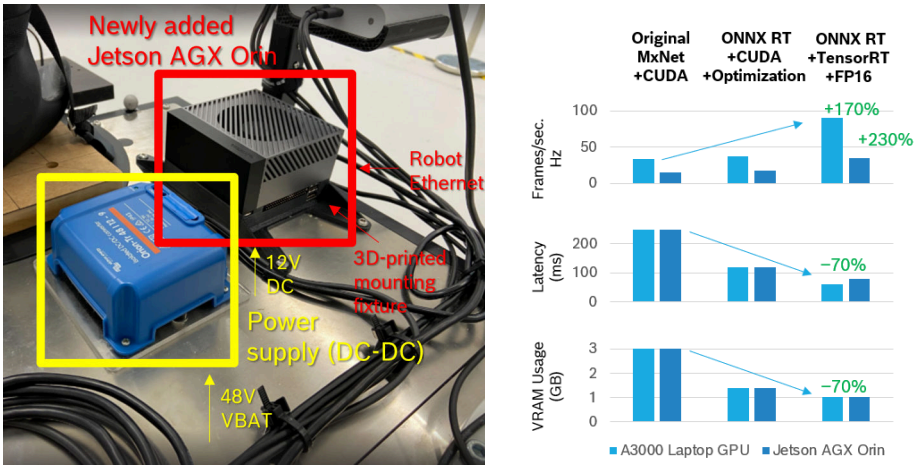

Figure 57: *Left:* Bosch has mounted an additional Nvidia Jetson AGX Orin embedded compute device on top of the robot (grey/black box in the middle rear) using a custom 3D-printed mounting fixture and an isolated 48-to-12V DC-DC converter (blue box) attached directly to the robot's battery. *Right:* Performance improvements obtained by porting the RGB-D YOLO detector from MxNet with CUDA backend to ONNX Runtime (ORT) with CUDA backend, and finally to ORT with TensorRT backend and 16-bit floating point precision. Results for frame rate, latency and memory usage are shown both on a laptop GPU (light blue shade) and the Jetson embedded device (dark blue shade); we obtain a significant reduction in latency and resource consumption, creating more space for the deployment of further perception components.

source on the target machine. The resulting network was thus not easily deployable on an Nvidia Jetson embedded computing device with its ARM64 architecture, where this build process can take up to a day. Furthermore, memory utilization was rather high (4 GB for human RGB-D YOLO, another 4 GB for 9DoF objects RGB-D YOLO from T2.1) and latency sub-optimal for real-time usage ($\approx$ 100 ms). The latter is a problem especially if followed by an additional second-stage top-down 3D human pose estimation module that adds additional latency.

For their desktop and embedded GPUs, Nvidia provides TensorRT as a more efficient, graph-based neural network execution engine which provides numerous optimizations for runtime efficiency during inference. To convert existing networks developed in various deep learning frameworks to TensorRT or other backends, the ONNX (Open Neural Network Exchange) format is an established intermediate format supported by many vendors.

We therefore invested effort in converting the original MxNet model to an ONNX graph, and from there to TensorRT. This involved the following steps:

1. Export of our trained RGB-D YOLO detector from MxNet as a symbolic graph.
2. Export of the MxNet symbolic graph to ONNX using MxNet's `mxnet.onnx` module.
3. Removing and replacing specific parts of the exported ONNX graph that are currently not supported by TensorRT, e. g. the non-maximum suppression and topN nodes, using the open-source `onnx-graphsurgeon` tool.
4. Graph-level sanitization and optimization (with constant folding) of the modified ONNX graph using the `polygraphy` tool.
5. Creating a new version of our C++ ROS node for RGB-D YOLO which uses the Microsoft ONNX Runtime (ORT) instead of MxNet C++ API for inference.

6. Bootstrapping of a CUDA and TensorRT execution provider for ORT in the ROS node.
7. Loading the optimized ONNX model to build an optimized TensorRT engine with FP16 precision at first startup (takes 10-30 minutes, depending on the hardware).
8. Storing the optimized TensorRT engine in a cache on disk for subsequent faster startup (less than 10 seconds).

Implementing these steps led to major performance improvements in both runtime, latency and memory usage, as shown in Figure 57 (right). The resulting TensorRT version of RGB-D YOLO can easily be applied also to the 9DoF RGB-D YOLO for 3D object detection from task T2.1, therefore allowing for parallel deployment of the entire T2.1 and T2.5 software stack on the Jetson Orin at real-time frame rates. This includes the top-down 3D body pose estimation approach based upon MeTRAbs [56], which we so far run in standard CUDA mode and which could be further optimized using ONNX, TensorRT and quantization in the future.

6.2.3 A faster YOLO-based human detector for very low-end embedded hardware

Further research on embedded deployment of human detection have been conducted by partner UoL, targeted at implementing a logistics safety system based on computer vision algorithms. The concrete goal here was to detect people in working environments using a *low-end* embedded device, with an even lower price and power consumption than the Nvidia Jetson embedded GPU currently used on the DARKO robot.

To this end, in a paper [13] accepted at VISAPP 2023, UoL compared multiple implementations of computer vision-based human detection algorithms on two power-efficient and inexpensive devices, namely the Raspberry Pi 3 and 4. First, computationally efficient solutions based on off-the-shelf algorithms were explored, namely the Histogram of Oriented Gradients detector (HOG), Viola's Haar detector, and YOLO v3. These were then compared to a newly-created custom architecture, called *eYOLO*, that tries to address limitations of the existing approaches. While Viola's Haar cascade and the HOG detector proved to be good in terms of runtime performance, they are unsuitable for the actual detection task. A YOLO deep learning architecture, instead, showed to be very accurate in detecting people, but required too much time to process a single frame, making this algorithm not suitable for embedded application on the Rasperry Pi target platform (reaching < 0.1 frames per second). These baseline comparisons inspired the development of a new optimised YOLO-based person detector to reduce the original network's size and achieve satisfactory runtime performance. The adopted strategy to re-design the network is a trade-off between performance and inference time that includes a customised loss function and batch generator for the training data, and the right combination of hyper-parameters and convolutional layers. The resulting method runs approximately at 3 Hz on a Raspberry Pi 4 device. Qualitative results are shown in Figure 58.

6.2.4 Docker stack for easy deployment of broader-level scene understanding modules

To facilitate easy deployment of all broader-level scene understanding components from tasks T2.1 (object-level semantics) and T2.5 (perception of humans) – thereby partly addressing DARKO's objectives O3 on easy deployment, and O5 on demonstration in an integrated system –, Bosch has developed a modular and reusable Docker stack that integrates all requires software components and sensor drivers: Docker images can be built on developer workstations or continuous integration platforms by executing a single build script, and another run script serves as the main entry point that starts up all components

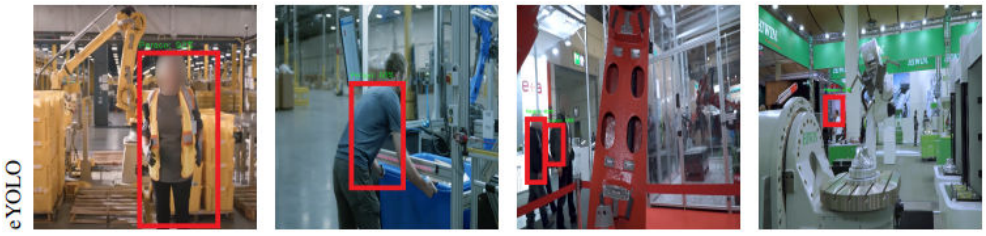

Figure 58: Qualitative results of the proposed eYOLO network for embedded 2D human detection on a low-end Raspberry Pi 3 or 4 device. The method achieves similar frame rates as a HOG-based detector (and is much faster than a standard YOLO v3), but higher accuracy. Some false negatives can be noticed in the background of the rightmost picture.

in different `tmux` panes in a terminal. Details on the technical structure of this software stack can be found in software deliverable D2.1.

The Docker stack has been successfully deployed across the consortium's robot fleet (consisting of three nearly identical platforms), as well as several developer workstations. To do so, it has been ported to both x64 (Intel/AMD) and arm64 (Nvidia Jetson) system architectures, and can be used both in real-time, online mode as well as with pre-recorded sensor bagfiles. Our experiences suggest that a great amount of time can be saved with this Docker-based approach, compared to manually deploying individual components across different computers and robots in a fleet by a software developer expert (as had been done in the previous ILIAD project). For instance, checking out the repository and building it now takes only around 15 minutes of manual effort (compared to days of deployment effort previously required). Another added benefit is that experiments and deployments are more consistent and repeatable using this approach, despite very complex system setups (sensor driver settings, system dependency conflicts, etc.).

6.3 A novel multi-modal multi-view dataset for 3D human pose estimation

The second, major sub-task of task T2.5 deals with the extension of 3D human perception to sensors with a wider field of view, in order to ideally provide full 360-degree surround view coverage around the robot to downstream components. This is a feature that has been widely requested by partners in the DARKO consortium that depend on the perception of humans, and is illustrated in Figure 59.

One possible approach to this could be to use a cylindrical array of classical RGB or RGB-D cameras, like in the recent JRDB-Pose dataset for 2D human pose estimation [57], where they used five stereo camera pairs. This approach can be both costly and computation-intensive, and one needs to deal with numerous discontinuities at sensor boundaries. Two other, promising alternatives to address this task are fisheye cameras with 180+ degrees FOV (e. g. in back-to-back configuration), as well as high-resolution 3D lidar: Fisheye lenses, especially those with S-mount, can be quite affordable (5-10 EUR, or less), and two monocular cameras with such lenses would be sufficient to cover 360 degrees around the robot. Revoluting 3D lidar, on the other hand, offers 360-degree coverage out-of-the-box, and such sensors are often already mounted onboard robots for other purposes such as SLAM, navigation and collision avoidance. Unfortunately, as we will highlight in the following paragraphs, there are practically no suitable existing public datasets that contain either robot-centric fisheye camera or 3D lidar data with accurate 3D body pose groundtruth, while at the same time being usable also for commercial research. Therefore, in this subsection we will subsequently introduce our own, novel dataset that enables further research on 3D human pose estimation using wide-FOV sensors.

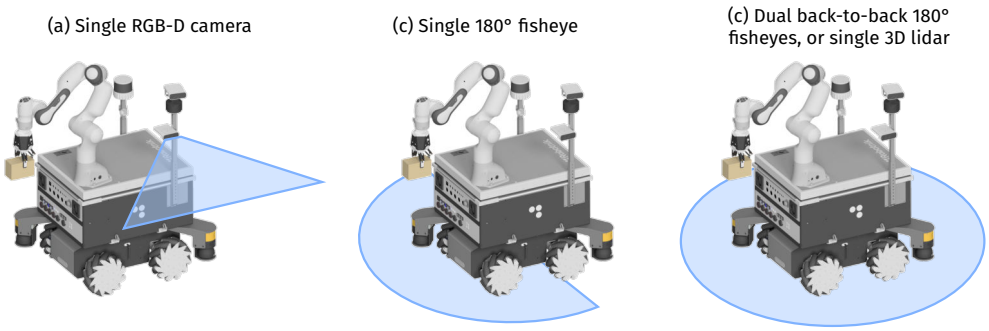

Figure 59: Observable horizontal fields of view obtainable using (a) a single Kinect-like RGB-D camera (around 90° FOV), (b) a single fisheye camera (usually 180-200°), (c) two back-to-back fisheye cameras or a single revoluting 3D lidar, yielding full 360° coverage around the robot. While it would also be possible to combine 4-6 pinhole cameras in a circular array, this would lead to increased sensor cost, computational overheads and more discontinuities.

6.3.1 Related work on 3D human pose estimation using pinhole, fisheye cameras or 3D lidar

Pinhole camera-based 3D human pose estimation: 3D human pose estimation using classical sensing approaches, i. e. monocular rectilinear RGB cameras or RGB-D cameras such as the Kinect sensors, has been widely studied in the past two decades. In their recent work, Sárándi et al. [58] provide an overview of 28 different publicly available datasets recorded with such non-fisheye cameras, that are often well-represented by a standard pinhole camera model with optional radial and tangential distortion (Brown-Conrady distortion model). The authors also show how to learn a mapping between the different, heterogenenous skeleton conventions of these datasets using a geometry-aware autoencoder, and integrate this extension into the MeTRAbs [56] pose estimation approach that is also used in DARKO. One issue of almost all publicly available benchmark datasets for 3D human pose estimation when it comes to use in commercially-oriented research, or potential later productization, are their licenses; out of the dozens of public datasets that exist, only very few (smaller) datasets such as the sports-related ASPSet [58] are released under a permissible, open-source license.

Fisheye-based 3D human pose estimation: Human pose estimation based upon fisheye cameras has, so far, mainly been explored by two different research communities: One major use-case is in video surveillance, where these cameras are usually mounted overhead at the ceiling and downward-facing [59]. This perspective is very different from the one observed by a mobile robot with onboard fisheye cameras. A second research direction are augmented and virtual reality applications. Here, the cameras are usually head-mounted, facing downwards, and focused at the ego-body [60, 61], i. e. the one single person wearing the AR/VR headset, as opposed to other persons in front of the camera. This perspective is therefore also quite distinct from the robot-centric perspective with forward-facing fisheye cameras, and networks trained on such data may not transfer well to our DARKO use-case.

Only few existing papers focus on use-cases that are potentially applicable to our robotics scenario. Kohei et al. (2021) [62] propose a system for multi-person 3D human pose estimation using a single chest-mounted ultra-wide RGB fisheye camera. In their approach, they first reproject input fisheye images to equirectangular (cylindrical) images before running a 2D human bounding box detector and 2D body pose estimation followed

Figure 60: Fisheye images (top row) and rectilinear images from a pinhole camera (bottom row) mounted at a similar position as the respective fisheye. The fisheye images provide significantly more scene context and humans are less frequently truncated at image boundaries. On the other hand, pinhole camera images are much less distorted (since they can easily be rectified) and provide more fine-grained details due to the longer focal length.

by 2D-to-3D uplifting based upon a method by Martinez et al. (2017) that has been extended with an absolute 3D pose recovery stage based on a ground plane assumption. They note that no suitable public training data with first-person fisheye images exists for 3D pose estimation, and therefore render a synthetic dataset that they use for fine-tuning. For testing, they use a real-world dataset for which (weak) groundtruth was acquired using a single Azure Kinect sensor, which can be limited in accuracy according to our own experiences (see Figure 69). Koji et al. (2022) [63] introduce a fisheye-based method for 3D human pose estimation in weightless environments, such as the International Space Station. They argue that the cylindrical projection does not work well in their scenario due to lack of a ground plane in weightlessness, and instead project cropped 2D person bounding boxes to rectilinear images using a virtual pinhole camera with smaller FOV. Similar to the previous method [62], they combine an existing pretrained top-down 2D pose estimator with 2D-to-3D-uplifting. They perform experiments on their own test dataset from a space station environment, for which they acquired groundtruth of only three selected 3D body joints using marker-based motion capture.

In Table 9, we provide an overview of datasets used by these different research communities. Some of these datasets are synthetically generated, while our goal is to obtain a real-world test set (in addition to training data) from which we can draw conclusions that directly relate to our target domain use-case. For those few datasets that have been publicly released, it can be seen that none sufficiently addresses the mobile robotics scenario with its robot-centric, forward-facing perspective that is illustrated in Figure 60.

3D lidar-based 3D human pose estimation: High-resolution 3D lidar sensors with up to 128 vertical scan planes have recently seen more widespread adoption for different tasks in autonomous driving and robotics settings, such as 3D object detection and tracking, SLAM

Name	Cam pos.	Cam dir.	Subjects	Use-Case	Availability
Zhang et al. [59]	Ceiling	Downward	Multiple	Surveillance	N/A
EgoCap [60]	Head	Downward	Ego only	AR/VR	Non-commercial
Mo^2Cap^2 [61]	Head	Downward	Ego only	AR/VR	Non-commercial
UnrealEgo [64]	Head	Downward	Ego only	AR/VR	Non-commercial
Kohei et al. [62]	Chest	Forward	Multiple	AR/VR, MoCap	N/A
ISS [63]	Robot	Forward	Multiple	Robotics (space)	N/A
Ours	Robot	Forward	Multiple	Robotics (mobile)	Commercial use

Table 9: Fisheye datasets with groundtruth for 3D human pose estimation. N/A means that no public source to download the dataset could be found.

or 3D occupancy prediction. Even though the raw data provides abundant depth- and shape-based cues that should facilitate the task as illustrated in Figure 61, so far only very few 3D human pose estimation methods have been presented specifically for 3D lidar data [65, 66, 67]. This is most likely due to the lack of suitable training datasets. As shown in Table 10, the few datasets that have been published so far focus mostly on urban automated driving scenarios, and have either been hand-labeled directly on 3D point clouds [68], use groundtruth from IMU setups prone to temporal drift [65], or been auto-labelled using complex single-view temporal model fitting techniques [69, 70]. Only a single, very recent outdoor dataset [71] that appeared on arXiv at time of submission of this deliverable includes accurate multi-view triangulated 3D human pose groundtruth for 3D lidar. To our best knowledge, there is no similar indoor dataset that provides accurate 3D human pose groundtruth and directly addresses the industrial and intralogistics scenarios along with body poses common in such use-cases relevant to DARKO.

6.3.2 Dataset requirements

A dataset for enabling research on wide-FOV 3D human pose estimation using either fisheye cameras or 3D lidar sensors should fulfill the following requirements:

- Include several time-synchronized, monocular fisheye color cameras.
- Include at least one high-resolution 3D lidar.
- Provide time-synchronized, accurate 3D groundtruth of body joint locations for each subject.
- Use a marker-free approach for groundtruth acquisition, to prevent neural networks from overfitting on such markers during training.
- Contain multiple persons per scene, to learn to handle inter-person occlusions.
- Be diverse in terms of body poses and camera perspectives.
- Be representative of our real-world target use-case, i. e. contain real-world images and not solely consist of synthetic data; and contain foreground and background objects related to the DARKO use-case.
- Be usable in commercial research.

Based upon these requirements, we now describe our complex multi-view recording setup that we developed in DARKO to acquire such a dataset. We include both fisheye cameras and 3D lidar, as a lot of the overall dataset creation effort is shared between these two modalities.

Name	Groundtruth	Subjects	Location	Scenario	Availability
LidarCap [65]	IMUs	Single	Mixed	Urban, no background	Dual licensing
PedX [69]	Single-view	Multiple	Outdoor	Automated driving	Non-commercial
WaymoOpen [68]	Manual labels	Multiple	Outdoor	Automated driving	Non-commercial
SLOPER4D [70]	Single-view + IMU	Single	Mixed	Urban	Dual licensing
Human-M3 [71]	Multi-view	Multiple	Outdoor	Urban	Research only
Ours	Multi-view	Multiple	Indoor	Industrial, logistics	Commercial use

Table 10: 3D lidar datasets with groundtruth for 3D human pose estimation

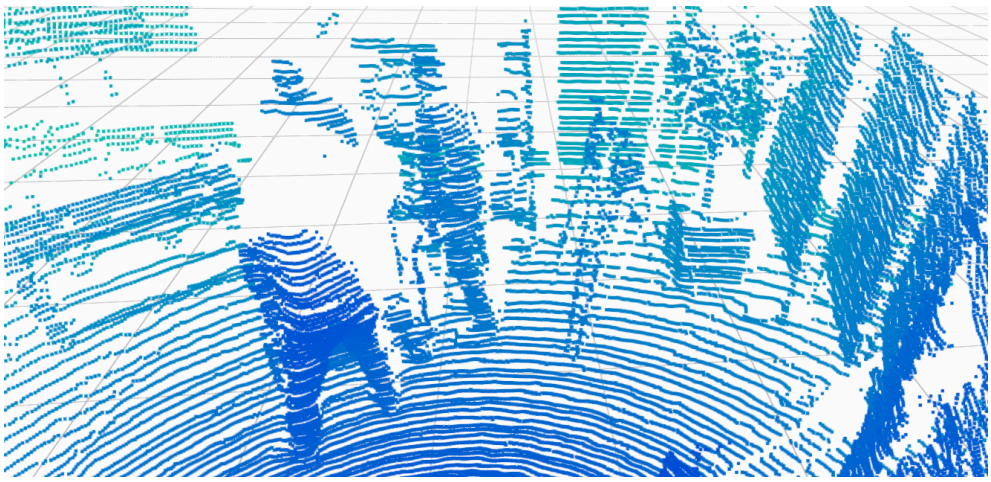

Figure 61: 3D lidar point cloud of a scene with the DARKO conveyor, DARKO shelf, a tripod, a plant and three humans in different body poses distinguishable by the human eye.

6.3.3 Heterogeneous multi-view sensor setup

Our multi-view recording setup is composed of up to ten machine vision color cameras (with fisheye lenses), four Azure Kinect Dev Kit RGB-D cameras, one high-resolution 3D lidar, two PCIe frame grabber cards, and two recording computers with fast storage. The 3D lidar and two of the machine vision cameras are part of a portable sensor box, shown in Figure 3, developed in-house at Bosch in a previous project. The remaining cameras are all placed on tripods, or attached e.g. to desk surfaces using clamps.

The large array of portable fisheye and RGB-D cameras in our setup is easily re-arrangeable, and yields many different views of a scene – which is crucial for efficiently generating large amounts of training data, and for accurate multi-view triangulation of 3D groundtruth poses. At the same time, the Kinect RGB-D cameras provide dense point clouds that can serve as an additional form of reference, and aid with their 3D output in associating person identities across views (if there are multiple persons in a scene).

A summary of all sensors is presented in Table 11. Figure 62 shows a connection diagram of the entire setup. In the following sections, we explain how we solved the challenging problem of time-synchronization across these heterogeneous sensors, and how we calibrated our multi-sensor setup both intrinsically and extrinsically, before we introduce our software solution for multi-view triangulation.

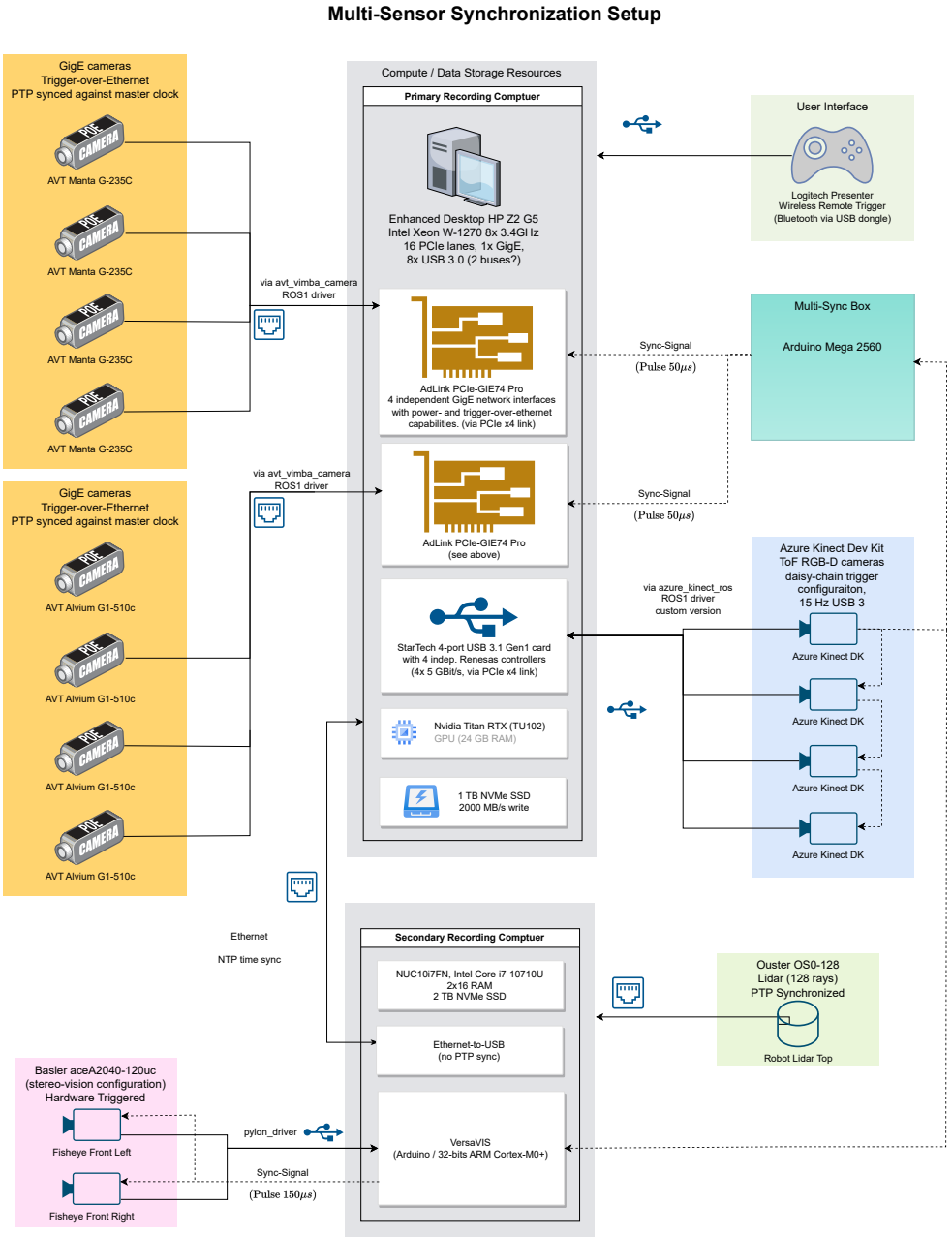

Figure 62: Sensors, recording computers, and their connections. Solid lines represent physical wire connections for data transfer. Dashed lines represent physical wire connections for hardware trigger signals.

Sensor	Modality	#	Interface	Selected Trigger Mode	Timestamping
AVT Manta G-235c	RGB	4	GigE	Trigger over Ethernet	PTP
AVT Alvium G1-510c	RGB	4	GigE	Trigger over Ethernet	PTP
Azure Kinect DK	RGB+D	4	USB 3.0	Fixed Rate on Master Master triggers Subordinates	HW clock + driver offset
Ouster OS0-128	3D Cloud	1	GigE	Fixed Rate (10 or 20Hz)	PTP
Basler acA2040-120uc	RGB	2	USB 3.0	Hardware GPIO	SW-assigned

Table 11: Sensors utilized in the heterogeneous multi-sensor setup for acquisition of 3D human pose estimation groundtruth. Sensors below the dashed line are part of the portable sensor box and attached to a separate computer used for recording.

Multi-sensor synchronization

In a homogeneous multi-sensor setup, one method of synchronization, e. g., hardware triggering, is normally sufficient to ensure that acquired data can be associated across different sensor sources. A heterogeneous multi-sensor setup, however, requires a range of methods to achieve accurate synchronization and data association over time. In our setup, each sensor supports a different subset of triggering modes, therefore not a single way of data capturing, e.g., hardware triggering, can be utilized.

While most data can be associated via their timestamps, sensors might capture data at different, un-synchronized rates, making data association difficult and ambiguous. Complexity is further increased if data is acquired in a distributed fashion, that is, sensors are attached to different computers. Hence, achieving a tight temporal synchronization of recorded data turned out to be a key challenge in developing our heterogeneous multi-view recording setup.

Figure 63: The Arduino-based multi-sensor sync box. Its input is a trigger sync signal generated by, e.g., an Azure Kinect DK camera. It extends the pulse width, and optionally adjusts the trigger rate, for other machine vision cameras in the system.

Synchronization of system clocks: In order to synchronize the clocks of multiple recording computers, two in our case, several options are available. Two predominant and well supported ways are either using NTP or PTP, with the latter being the more precise

option, but only supported on native Ethernet hardware. However, our mobile recording computer to which the LiDAR and the two Basler cameras are attached, offers only one native Ethernet port, which is already used by the LiDAR. Hence, PTP is not available. Instead, we utilize NTP, specifically the *chrony project* implementation thereof, to achieve synchronization between system clocks. Although the secondary computer utilizes an Ethernet-to-USB bridge, precision of around and below 1ms can be achieved, which is sufficiently below the average exposure time and frame rates used.

Camera clock synchronization vs. system clocks: Both types of Allied Vision machine vision cameras, Manta G-253c and Alvium G1-510c, support PTP. Here, the NIC, and its associated *PTP Hardware Clock*, provide the current time to the device. In order to achieve high throughput and low latency, each camera is connected to its own and distinct NIC and thus is assigned a unique and independent hardware clock. We utilize the tool *ptp4l* to configure each NIC and its hardware clock as *PTP master clock*. Additionally, we select one of the hardware clocks to serve as primary master clock and we synchronize all other hardware clocks against it using *phc2sys*. Furthermore, we also synchronize the computer's system clock against this hardware clock. The ROS driver is configured to use the respective camera's hardware PTP time as timestamp for generated image messages.

The Kinect RGB-D cameras do not support PTP. Instead they run an arbitrarily set internal hardware clock. The ROS driver will initialize an offset against the system clock upon arrival of the first camera image. It monitors the current offset between the system clock and the computer's monotonic clock to estimate drift against its internal hardware clock. The offset is updated using a low-pass filter approach. The system clock is, in turn, synchronized against the primary PTP master hardware clock.

Images from the Basler acA2040-120uc USB3 cameras within the portable sensor box (Figure 3) are directly assigned the system clock's time as ROS timestamp by the ROS driver (i.e. in software upon arrival), since the USB3 vision protocol also does not support PTP. The Ouster LiDAR only supports a fixed rate of either 10 or 20 Hz. As it a rotating LiDAR, each point in the pointcloud is acquired at different times and thus a tight clock synchronization is necessary. This is achieved by synchronizing the LiDAR using PTP against the NIC's PTP hardware clock, which in turn is synchronized to the system clock.

Hardware Triggering of Cameras: Table 11 provides an overview which frame triggering mode we use on which sensor of our setup. All machine vision cameras support triggering in hardware (via digital input, or Ethernet). The biggest limitations arise from the Azure Kinect DK cameras, which can only run at fixed rates of 5, 15, and 30Hz. They can, however, be hardware synchronized using their *sync in* and *sync out* ports. Here, one camera is designated as *master* and all other cameras as *subordinate*. We found that externally generated trigger signals caused image artifacts, and only sync signals generated by another Kinect reliably triggered other Kinects. Consequently, we selected one Kinect as generator of the hardware trigger signal for the complete system.

The Allied Vision machine vision cameras are connected to Adlink Tech PCIe-GIE74P frame grabber cards. These provide a dedicated opto-isolated trigger input port, which can be configured to send trigger-over-Ethernet (ToE) Action commands upon detection of a rising or falling edge on the input. The 8-12µs pulse signal generated by the Kinect, however, is too short to be registered reliably by the frame grabber cards. As remedy, we relay the trigger signal via an Arduino microcontroller, which upon a detected rising edge, generates a new, longer trigger signal with a pulse width of 50µs. Figure 63 shows the hardware implementation of our multi-sensor sync box.

Similarly, the Basler cameras inside the portable sensor box (Figure 3) are attached to an Arduino-based Camera+IMU trigger board. We extended its firmware to detect an

attached hardware trigger on its AUX port and trigger the cameras with respect to the external trigger signal, instead of the original, internal timer-based triggering.

This way, overall, we can trigger all cameras in our setup (almost) simultaneously with negligible time differences, which is crucial for the later multi-view triangulation.

Multi-sensor calibration

For diversity of training and testing data, we use different camera-lens combinations in our setup. For example, some of the fisheye lenses are rather cheap and affordable (< 10 EUR) and well-suited for use in low-cost robotics systems, whereas others are more expensive, high-quality, low-distortion C-mount fisheye lenses that can easily cost 800–100 EUR. Therefore, camera intrinsics may vary significantly within our setup. At the same time, an accurate, easy and efficient approach to extrinsic camera calibration is required since our multi-view setup is portable and usually set up and dismounted within a day or two (which makes it possible to set it up e.g. in different environments with different backgrounds).

Intrinsic camera calibration: For intrinsic camera calibration, we placed all cameras including Kinects on tripods, facing into a similar direction, and used an April tag board that we placed on a forklift to keep the board steady in different poses relative to the camera (Figure 64). A commercially available tool (calib.io) was used for intrinsic camera calibration. Kinect camera intrinsics were obtained using the standard OpenCV pinhole model. Fisheye cameras were calibrated using Double Sphere model, extended unified camera model (EUCM), and Mei model (using the Kalibr calibration suite) and initialized using a fisheye-specific initialization method. All fisheye camera-lens combinations used in our setup converged using these three models, while other camera models (such as the OpenCV fisheye model) failed to converge in several instances. Out of these three models, Double Sphere – as the most recently published model [72] – appeared to be the most useful due to its consistently low reprojection errors and availability of an analytical inverse, which is used often during the later multi-view triangulation. It shall be noted that the choice of camera model can also play a role later during absolute pose recovery of fisheye-based 3D human pose estimation.

Extrinsic camera calibration: For extrinsic calibration with the final camera placements, the calibration board was placed on the ground, on a tripod, or hand-held in around 100 different poses relative to the cameras. We use the same commercial tool (from calib.io) for extrinsic calibration. Here, we found sufficient camera resolution to be essential, since even a 1x1 meter April tag board can appear quite small in a fisheye lens at larger distances. For extrinsics to converge, we had to perform calibration in two sequential steps (first obtaining intrinsics using the above procedure and temporary camera setup with all cameras at close-range and facing into a similar direction, then extrinsics of the final camera setup using the previously saved intrinsics). Furthermore, some frames with high reprojection errors had to be excluded from the optimization.

Camera-to-lidar calibration: For camera-to-lidar calibration on the SLAM-Box, we used a recently proposed approach for direct visual-lidar calibration by Koide et al. (2023) [73]. Using the resulting transformation, and the previously obtained extrinsic camera calibrations, we can finally transform all cameras into lidar frame, and vice versa.

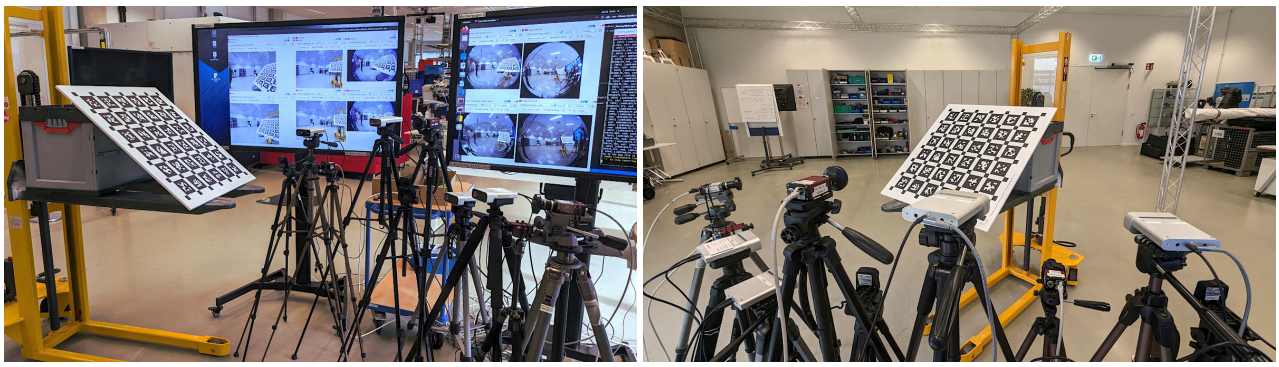

Figure 64: Intrinsic calibration of the fisheye cameras and Kinects used in the multi-view data recording setup.

6.3.4 Multi-view recording sessions

Using the previously described multi-view recording and triangulation setup, Bosch has started to incrementally create a diverse training and test set for wide-FOV 3D human pose estimation. The dataset consists of multiple recording sessions. While initial recording sessions were recorded in stop-motion fashion, i. e. participants had to 'freeze' while frame capture was manually triggered, the latest session that is currently being recorded also features continuous motion sequences, which can be useful for developing approaches that use temporal features for tracking, filtering, prediction and motion classification.

First recording session: In a first recording session in March 2023, multimodal data (RGB and depth images, 3D lidar point clouds) and 3D human pose groundtruth of up to 4 persons per scene was recorded in an industrial scenario replicating the DARKO use-case with DARKO shelf, conveyor, and robot in the background. 10 individual human subjects, entirely comprised of researchers who volunteered to participate in the recording and have signed consent forms, were part of this recording in a controlled lab environment. Scripted activities include walking, standing, sitting, throwing, doing push-ups, handing over objects, unloading a cart, re-stocking a shelf, pointing at objects, making a phone call, or drinking from a bottle. Four AVT Manta G-235C cameras (2x Fujinon CS-Mount FE185C086HA-1 fisheye lens, 2x Evetar M118B029528W M12 fisheye lens via CS-mount adapter), two low-cost USB IMX323 fisheye cameras with stock lens, one Ouster OS0-128 3D LiDAR on the DARKO robot with 128 rays vertical resolution, and four Azure Kinect DK RGB-D cameras were part of this setup. All cameras were mounted on tripods in a circular arrangement of approx. 4 meters diameter. Frame acquisition was manually triggered using a wireless bluetooth trigger. Cameras on the robot were not used due to time synchronization issues. See Figure 65 for a picture of the scene and sensor placement.

Second recording session: The second recording session in July 2023 comprises only a single human subject performing various actions, including sitting on the ground, carrying around boxes, standing on a step-stool. The same set of cameras was used as in the first session, but no 3D LiDAR as the DARKO robot was already at ARENA 2036 for the MS2 demonstration. In contrast to the first session, fisheye cameras were placed more diversely in the scene, including low to the ground and facing upwards (at 45° or 90° pitch angle), or higher up and facing downwards at around 45°, to represent different possible robot setups. This session was again recorded in stop-motion fashion using a wireless trigger. See Figure 66 for an example picture and the camera placement.

Figure 65: First multi-view data recording session setup for 3D human pose estimation, with the DARKO shelf, conveyor and other industrial objects as background. The right picture shows the placement of the 4 machine vision cameras with fisheye lenses, 2 low-cost USB fisheye cameras, 4 Azure Kinect RGB-D cameras and an Ouster OS0-128 3D lidar (onboard the DARKO robot).

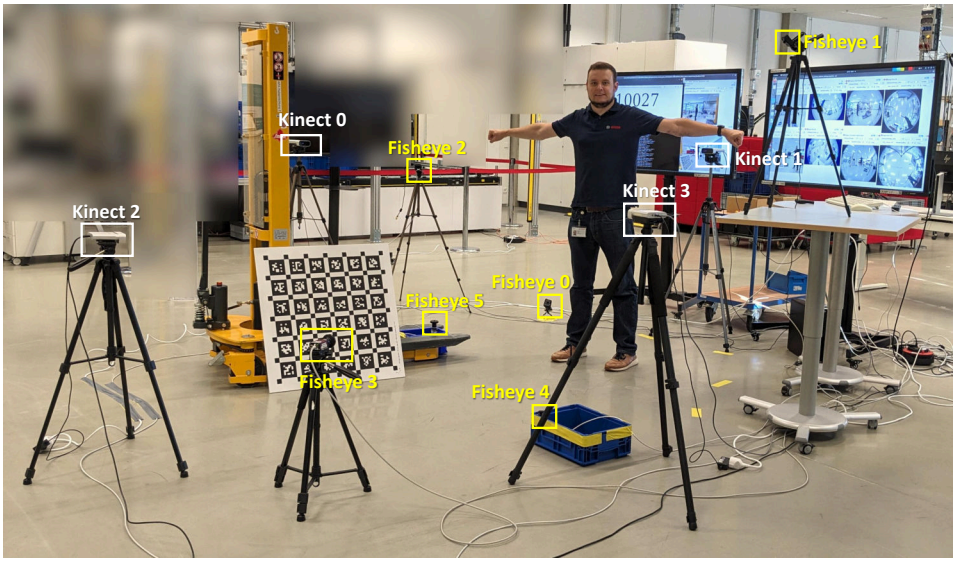

Figure 66: Second multi-view data recording session, with a different more close-up camera setup and two fisheye cameras in upright orientation.

Third recording session: A third recording session has been started in December 2023 and is planned to be finished in February 2024. In contrast to the previous two recording sessions, we now use hardware-based time synchronization for acquisition of continuous motion sequences and integrated Bosch's mobile sensor carrier with its Ouster OS0-128 3D lidar to replace the DARKO robot that is now permanently stationed at ARENA 2036, as described in Section 6.3.3. We have further extended the set of cameras to eight GigE machine vision cameras (4 AVT Manta + 4 AVT Alvium) in total, omitting the low-cost USB fisheye cameras which cannot be easily time-synchronized. The final dataset is planned to include diverse scenes with various backgrounds (industrial / lab environment, office scenario, household scenario) to improve cross-domain generalization of the methods trained on it. We will clarify in the final project period if it is possible to make this dataset or parts of it publicly available for research purposes.

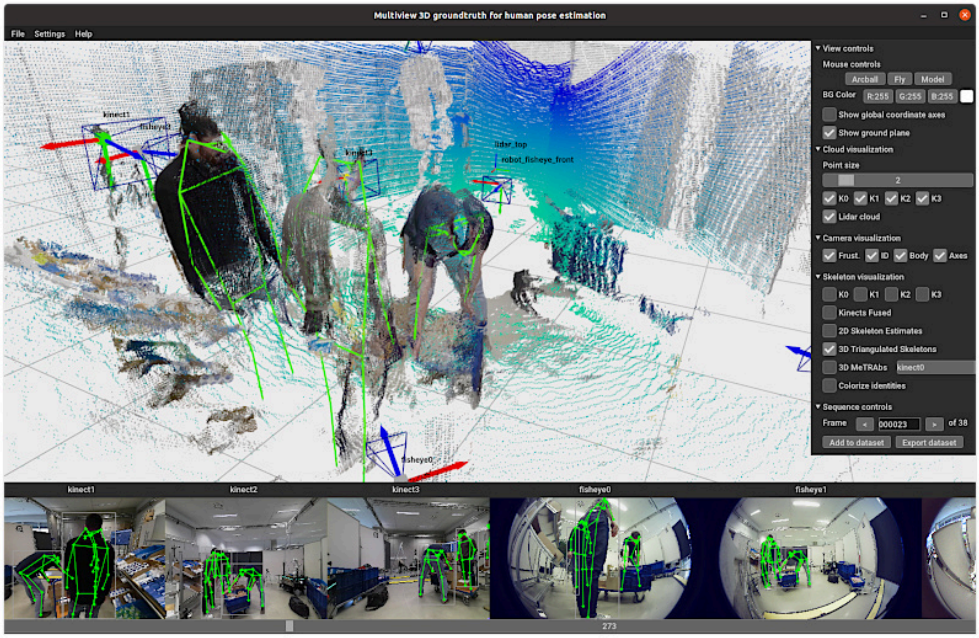

Figure 67: An interactive 3D visualizer developed using Open3D for asserting correct sensor registration and evaluating groundtruth and predicted 3D poses from our multi-view sensor setup. Shown in green are the triangulated 3D groundtruth body poses based upon the MS COCO skeleton, bottom row shows projections of 3D poses into the different 2D camera views.

6.3.5 Multi-view triangulation of groundtruth 3D body poses

Interactive 3D visualization GUI: To ensure accurate, high-quality 3D groundtruth poses, proper multi-sensor calibration and registration, visualize alignment with RGB-D and LiDAR 3D point clouds and be able to iteratively develop and debug the multi-view triangulation approach, we developed an interactive, Python-based 3D visualizer for our multi-view dataset using the Open3D open-source toolkit. Our interactive visualizer, shown in Figure 67, loads post-processed and time-synchronized sequences of our dataset and displays them frame by frame. Different skeletons, for instance 3D single-view skeletons from the Azure Kinect sensors, fused 3D Kinect skeletons, 2D skeletons estimated by an off-the-shelf 2D human pose estimator, as well as resulting triangulated multi-view 3D skeletons can easily be shown or hidden in both the main 3D view, as well as projected into the different 2D camera views (at the bottom of the window). The interactive visualizer also provides functionality for exporting selected frames, or the entire dataset, into different target formats that are used for training 3D human pose estimation methods. Via command-line, it can also be operated in a batch fashion to process an entire dataset, or selected frames, at once without need for manual intervention.

Estimation of groundtruth 3D body poses: The interactive visualizer internally invokes routines for multi-view triangulation in order to derive the groundtruth 3D body poses. In the following, we briefly describe how this has been implemented.

2D body pose estimation: Multi-view triangulation tries to approximate the global 3D position of every body joint of a given human from corresponding multi-view 2D joint positions. The first step is therefore to estimate 2D human body poses across all camera

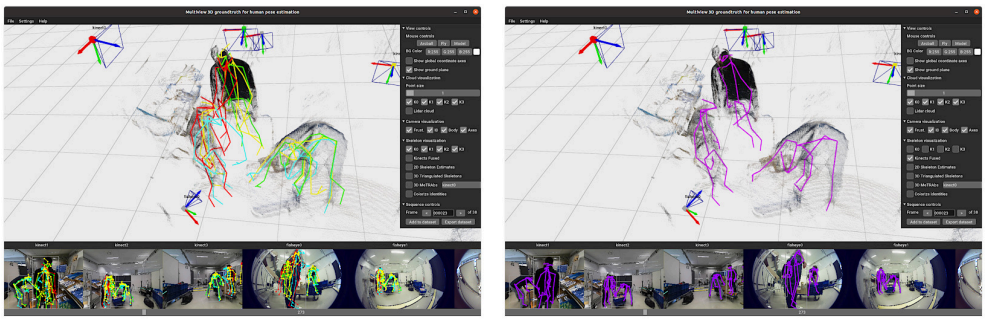

Figure 68: *Left:* Rough and noisy initial single-view 3D body pose groundtruth estimates from individual Azure Kinect views (distinguished by colors) obtained using the official Body Tracking SDK, with smoothing disabled. *Right:* Fused multi-view Kinect skeletons. There are still several inaccuracies, see for example the arms of the person in the top middle.

views. We experimented with recent 2D body pose estimation methods from OpenMMLab's `mmpose` framework, and qualitatively found that state-of-the-art e. g. Transformer-based methods perform well in image space even on strongly distorted fisheye images. We therefore use such an off-the-shelf top-down pose estimator (ViTPose-huge) [74] for this purpose, which is combined with a state-of-the-art 2D object detector such as CO-DETR (we use CO-DINO with Swin-L backbone trained on Objects365 followed by MS COCO) [75] from `mmdetection` for detection of human 2D bounding boxes as an initial step.

Naive Kinect 3D skeleton fusion: Using the Azure Kinect RGB-D cameras in our multi-view setup, we can obtain an initial set of rough 3D body skeleton estimates using the official SDK provided by Microsoft. These single-view 3D RGB-D skeleton estimates are often noisy and not always correct (Figure 68, left). We can improve the quality of these initial 3D skeleton estimates by fusing the skeletons from all four Kinect sensors. To achieve this, we first associate skeletons from the four Kinect cameras in 3D space via hierarchical clustering. We then fuse the skeletons of each cluster (containing the 1..N views in which the particular human was detected) by geometric averaging with statistical outlier removal. The resulting fused 3D Kinect skeletons (Figure 68, right) are more accurate, but sometimes still incorrect, for example persons sitting on the ground are systematically predicted as standing with the legs protruding into the ground. Since this happens in every view, the outlier removal cannot resolve these artifacts. In the following, we describe how we subsequently obtain more accurate 3D groundtruth via 2D-to-3D triangulation.

2D to 3D identity association: Several scenes in our dataset include multiple persons. Before we can triangulate each person's pose one after another, we need to associate all those 2D body poses from all camera views that belong to the same person with each other. In 2D image space, this is more challenging than in 3D because each camera uses its own local coordinate space. Existing multi-view 3D human pose estimation frameworks such as OpenXR's XRMoCap [76] therefore use complex temporal tracking and multi-way matching approaches with appearance-based affinity estimation for this purpose.

However, in our case, we can use a more lightweight approach that leverages the fused multi-view Kinect 3D skeletons from the previous step to bridge between the 2D and 3D coordinate systems, by projecting them into every single 2D camera view using the known camera intrinsics and extrinsics. Next, we perform 2D skeleton association by solving a linear assignment problem using 2D pixel-wise distances of selected key joints (e. g. pelvis,

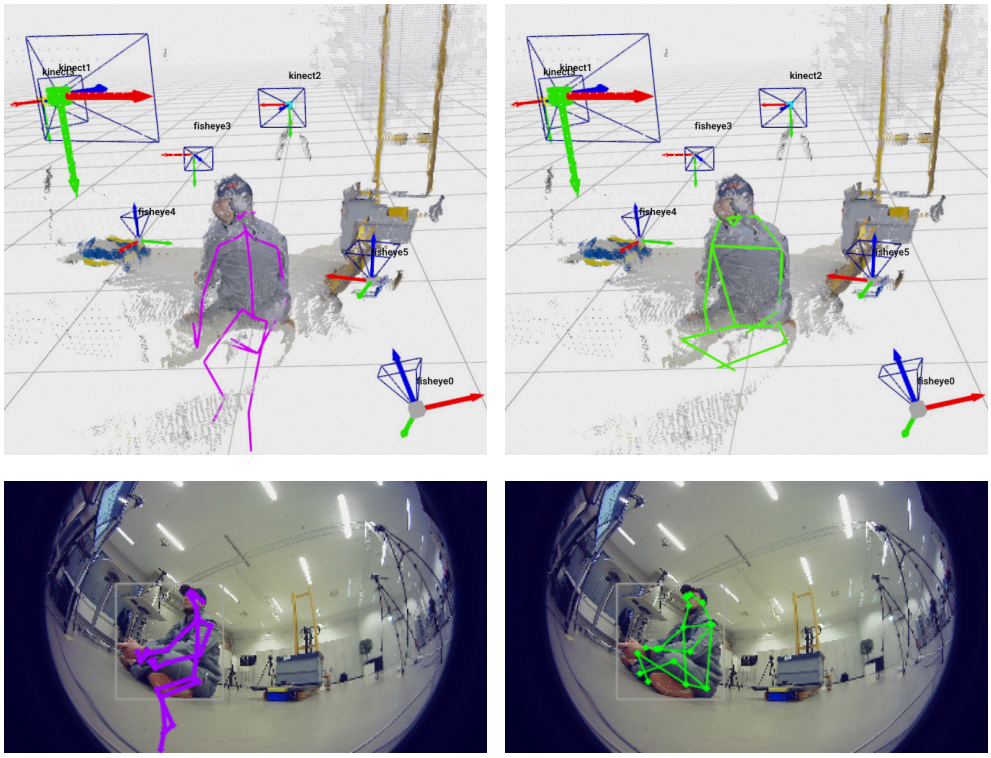

Figure 69: *Left:* Rough initial 3D body pose groundtruth obtained by fusing up to 4 Kinect skeletons. In this example, all single-view Kinect skeletons suffer from the same issue (standing instead of sitting pose estimated, leading to groundplane penetration), which cannot be mitigated by outlier rejection. *Right:* Final triangulated 3D body pose groundtruth, based upon multi-view skeleton estimates from an off-the-shelf 2D pose estimator run on 10 camera views. The bottom row shows the projection into an exemplary fisheye view for illustration purposes. Note that Kinect and triangulation use different skeleton conventions.

shoulders, ankles) as an objective function to minimize. Finally, all 2D skeletons associated with a given fused 3D Kinect skeleton in 2D image space are assigned the same identity.

Multi-view triangulation (2D to 3D): Now we know which 2D skeleton estimates from the off-the-shelf 2D human pose estimator belong to the same person, such that we can feed them into a triangulation pipeline. For triangulation, we use an existing code base that iteratively optimizes the weighted 2D reprojection error of all 3D body joints using a Pytorch optimizer. To obtain initial 3D estimates, we use the Mid-point method[5] (with 2 camera views) or an iterative pairwise version of it inspired by Roy et al. [77] (for N camera views). Our implementation includes routines to unproject from image to camera space and vice versa for three different camera models (pinhole, Mei omnidirectional and Double Sphere). Exemplary triangulated 3D body poses, which we use as the final 3D groundtruth in our dataset, are visualized in Figure 67 and Figure 69, where we also show a comparison to the fused rough initial 3D skeleton estimates from the Azure Kinect SDK and can see that our triangulated body pose is much more accurate.

[5] https://en.wikipedia.org/wiki/Triangulation_(computer_vision)#Mid-point_method

6.4 Methods for wide-FOV 3D human pose estimation

Fisheye-based methods: As previously noted, body joint localization in 2D image space works relatively well on fisheye images with existing 2D pose estimation approaches. The more challenging issue is to correctly recover absolute 3D poses from fisheye imagery; the existing MeTRAbs approach [56] cannot be used as-is on fisheye images since the absolute 3D pose recovery stage is based upon differentiable geometry and assumes a pinhole camera model. As a first working baseline on fisheye images, we have therefore implemented a virtual camera that reprojects fisheye-based human bounding box crops to a virtual pinhole view. The advantage of this approach is that the existing 3D pose estimator can be used as-is without retraining. A limitation of this approach is that pinhole reprojection leads to heavy distortions when the field of view is larger than 120–140 degrees, which occurs when a person is very close to the sensor and its bounding box thus occupies the majority of the fisheye image. However, for persons at medium to large distances, the required field of view is usually smaller and thus virtual pinhole reprojection, similar to what is done in [63], can work relatively well. As part of an ongoing academic collaboration outside of DARKO, a more principled approach to this problem will be developed that will yield accurate 3D poses also at close-range.

3D LiDAR-based methods: As motivated in Section 6.3.1, 3D lidar-based human pose estimation is a promising and growing research field relevant to different robotics use-cases. Given the availability of the previously introduced novel multi-modal dataset, Bosch was planning to conduct research on this topic with a master thesis student during the second period. However, due to recruiting delays, this research had to be postponed and will now commence at the beginning of the third period. The goal of this thesis is to evaluate, compare and extend existing approaches that operate on depth images or 3D point clouds for the 3D human pose estimation task on lidar data. If research progresses well, first results will be demonstrated during the MS3 demonstration in May 2024.

6.5 Temporal filtering of articulated 3D human poses

Strong or partial occlusions of persons are frequently observed from a mobile robot's egocentric perspective. Based upon practical observations during DARKO's MS2 demonstration, single-frame 3D human pose estimation approaches may fail or produce noisy estimates during such occlusion events. While such noise on the 3D joint locations can be reduced via fixed-lag smoothing, this introduces additional lag which is undesired in many e.g. safety-relevant use-cases. Therefore, being able to properly predict a 3D articulated human motion into the future for a limited time horizon can be an important ingredient to a robust perception of 3D human poses.

6.5.1 A novel approach for temporal full-body 3D human pose prediction

Many earlier kinematics prediction approaches use techniques based upon classical Bayesian filtering. However, modern learning-based approaches nowadays appear more promising since they can easily learn and incorporate anatomic body constraints and further geometric and social contextual cues into the prediction.

As a joint work between T2.5 and T5.1 (prediction of human motions and intents) and as part of a jointly supervised master thesis at Bosch, an efficient Transformer-based approach for joint short-term 3D full-body pose prediction and global trajectory prediction has been developed. The novel approach, which is currently in preparation for submission to IROS 2024 or RA-L, enables real-time full-body 3D spatial dynamics prediction for robotic applications demanding immediate responses. As intellectual property is currently

Figure 70: Various navigational behaviors exhibited by different subjects in DARKO data. For each subject, we predict poses for 2 seconds. Green is groundtruth, red the prediction. Sampled poses are shown from left to right, ending with a bird's-eye view of their overall path.

still being secured, we will omit technical details of the proposed novel approach in this deliverable and focus on first experimental results. A full description of the Transformer-based architecture and how it is trained will be presented in deliverable D5.1.

6.5.2 Results for temporal full-body 3D human pose prediction

To assess our novel method's performance, we conducted extensive experiments on a real-world dataset collected with the DARKO human perception stack of the robot, as well as on two standard public datasets of full-body human motion. Quantitatively, our

solution excels in predicting both pose dynamics and trajectories across diverse walking styles, including running, acceleration, deceleration and turning. These are the types of motion expected in an industrial environment which are important for safe and socially-compliant robot navigation and operation (e.g. throwing). Additionally, the method provides noteworthy qualitative results, particularly in prediction of plausible skeletal structures. Finally, the proposed approach offers considerable computational advantages over the prior art (e.g. 46% fewer parameters). Figure 70 shows qualitative 3D body pose and trajectory prediction results for different human walking sequences. In such scenarios, poses and trajectories are accurately predicted for the next 2 seconds, using past 3D body pose estimates from MeTRAbs as an input. In the next project period, we aim to integrate this approach with the real-time 3D skeleton tracking on the DARKO robot.

Human actions where the body is fully articulated – for example handing over an item to a robot, throwing or catching an object, or working with tools at a workbench – are much more challenging to predict. Here, incorporation of further scene context (e. g. pose of other objects and humans in the scene, the current room type, past human actions) can likely lead to significant improvements, which Bosch intends to further examine as part of T5.1 with a follow-up student.

6.6 Human activity recognition

Being able to recognize and distinguish different human activities can be a key skill which can inform e. g. task planning and semantic-aware navigation modules on a mobile robot. Bosch therefore investigated how to leverage the previously described articulated 3D skeletons from a per-frame human pose estimator to classify body poses into human activities in a data-efficient way, without requiring additional large-scale training datasets.

6.6.1 A data-efficient skeleton-based approach for human activity recognition

Our method exploits articulated 3D skeletons as a low-dimensional abstraction from human visual appearance and camera pose. Therefore, it requires significantly less training data than competing action recognition methods trained directly on image or video data. In our case, 2-3 short video sequences per activity from a single perspective are sufficient, making it a remarkably data-efficient approach. Figure 71 illustrates some of the activity classes that our proposed method is currently trained on; not shown here are the *doing push-ups* activity and various pointing gestures and human-object interaction.

The workflow at inference time is as follows:

1. Transformation of all per-frame skeletons from sensor frame into an upright coordinate frame where the XY axes define a ground plane.
2. Generation of appearance-, scale- and perspective-agnostic skeletal features (per-frame), by estimating the shortest angle between pairs of all relevant bones of the skeleton, as well as additional virtual bones (chest normal, head normal, ground plane normal), for each human in the scene.
3. Multi-class classification via a previously trained support vector machine ensemble with radial basis function kernels, i. e. $n \cdot (n-1)/2$ RBF-SVMs using the one-versus-one approach for classification of n different activities.
4. Temporal identity association of per-frame skeletons with tracked person identities via an API provided by the tracking framework.

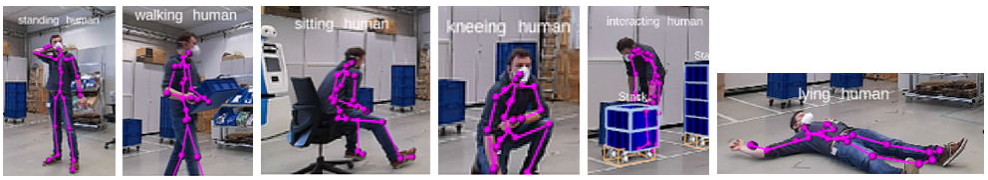

Figure 71: Different classes of human activities recognized by our data-efficient classifier.

5. Temporal aggregation of predicted class labels for each tracked person over a fixed time window, e. g. two seconds, for fixed-lag smoothing.
6. Selection of the class label that occurred the most often in the given time window.
7. Rule-based adjustment of predicted class labels by incorporating velocity information from the tracker, e. g. from *standing* to *walking* if average speed exceeds a certain threshold, as well as further contextual cues like close-by objects detected by T2.1.

The virtual bones used in feature generation (step 2) are computed by estimating a normal for the plane spanned by three key body joints associated with the respective virtual bone (e.g. left shoulder, right shoulder, spine for the chest normal). Since only angles between bones, and no absolute joint positions or bone lengths are used, the trained classifier is agnostic to the height and appearance of the person and thus requires only little data for training; it efficiently leverages the 3D human pose estimator as a bridge from the high-dimensional image space to a low-dimensional skeletal feature space.

6.6.2 Results for human activity classification

Our per-frame classifier, once trained on recorded skeleton sequences of a single human subject performing all scripted activities, achieves an accuracy of 93.2% (weighted F1-score: 94.6%) on a held-out validation set composed of the same activities performed by two different volunteers. As can be seen in Figure 72 (even without any temporal integration!), the only confusion observed on our validation set is between the *'sitting'* and *'kneeing'* classes, possibly indicating an unclear separation of the demonstrated activities in the training or validation sets. Overall, these results highlight the data efficiency of our approach, thanks to the abstraction provided by the 3D human pose estimator.

Finally, Figure 73 shows the activity classifier fully integrated into the real-time human perception stack running online on the DARKO robot during the milestone MS2 demonstration. In future work, more complex temporally varying actions and activities (e. g. dancing) could be incorporated by performing feature computation and classification on a temporal window, if skeleton-to-track association were performed prior to feature generation.

6.7 Summary of achievements and outlook for T2.5: Perception of humans

In conclusion, we have successfully achieved several sub-goals of task T2.5. Integrating the different human perception components into the DARKO system and increasing their runtime efficiency for deployment on embedded compute device have been important steps toward a fully functional DARKO demonstrator, on which many components need to run in parallel on the final system and computational resources are limited.

In order to significantly extend the field of view of existing 3D human pose estimation approaches by using fisheye cameras or 3D lidar, we introduced a novel, multi-modal, multi-view dataset with focus on industrial scenarios comprising accurate marker-less

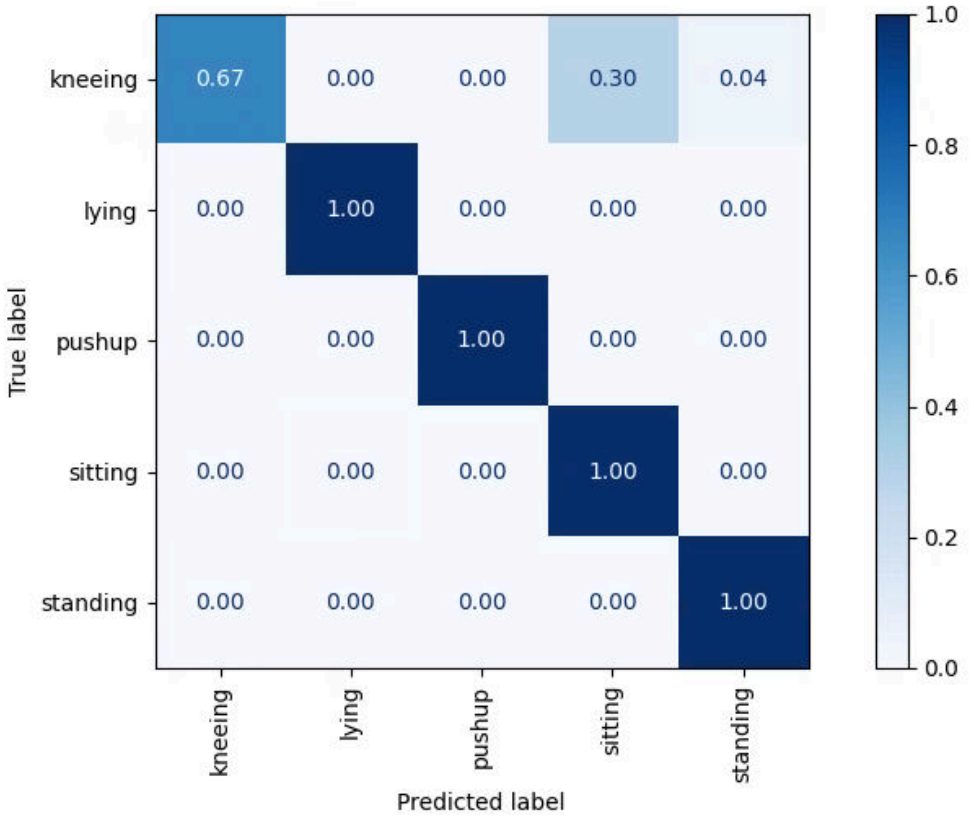

Figure 72: Confusion matrix for our per-frame human activity classifier on a separate validation set. It can be seen that the classifier overall achieves very high accuracy thanks to the skeletal abstraction provided by the 3D human body pose estimator, and the only significant confusion is observed between the *'sitting'* and *'kneeing'* classes.

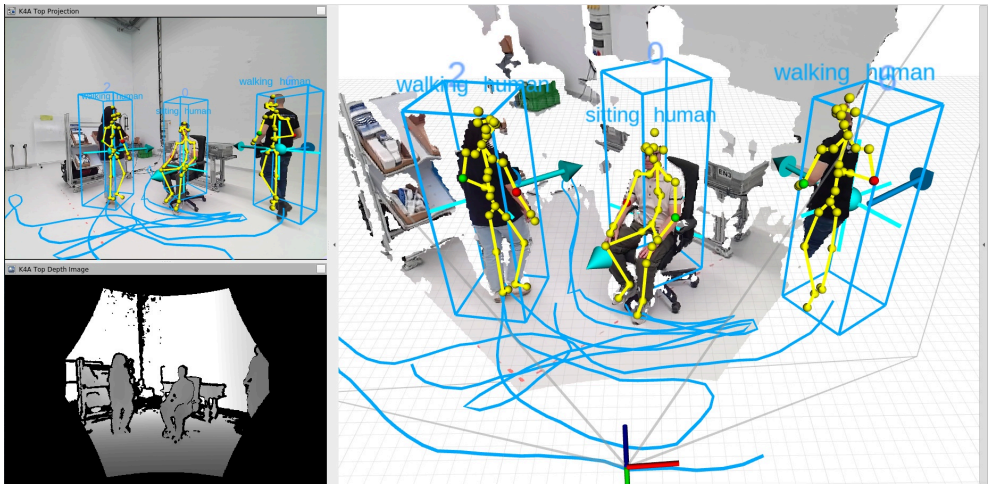

Figure 73: The per-frame activity classifier integrated into our real-time perception system and enhanced with temporal filtering and additional contextual cues is able to classify body poses into activities such as *standing, walking, sitting, kneeing, lying on the ground, doing push-ups*.

3D body pose groundtruth obtained through multi-view 3D triangulation. This dataset opens up many future interesting research directions. Based on this novel dataset, which will also facilitate later commercial exploitation of our research results, a first prototype method for fisheye-based 3D human pose estimation using virtual pinhole reprojection has been developed that we plan to deploy on the robot during the upcoming MS3 milestone.

To improve upon temporal robustness of estimated 3D body poses, we have researched a novel learning-based approach for short-term prediction of articulated 3D human body poses that can make 3D body pose tracking more accurate during temporary occlusions.

Finally, we introduced a data-efficient approach for skeleton-based human action and activity recognition that abstracts away human appearance and differences in anatomy, and can distinguish different body poses while achieving remarkable accuracy.

While task T2.5 officially ends at the time of submission of this deliverable, we will keep exploring novel methods for fisheye- and lidar-based 3D pose estimation based on the newly created dataset as part of ongoing thesis projects.

7 Conclusion

In this deliverable, the DARKO consortium has introduced novel methods, datasets and experimental results for broad-level scene understanding (tasks T2.1 and T2.5) and perception for manipulation and throwing (T2.2, T2.3, T2.4). These results represent major steps forward towards the research goals of DARKO and the final, fully integrated robot demonstration at the end of the project. All major goals of the respective tasks have been met, or even exceeded, during the first two periods of the project.

Intensive focus has been put by the consortium on experiments with actual data or objects from the industrial target domain of DARKO's lead use-case (as opposed to standard datasets from e. g. household scenarios which are more commonly publicly available). A further focus was on the efficient real-time deployment of an integrated system on a resource-constrained mobile robot, which is technically very challenging, but at the same time brings the results closer to practical application in future robotic products.

Several novel methods described in this deliverable have already successfully been deployed and demonstrated during the MS2 demonstration milestone and the associated stakeholder meeting at ARENA 2036, Stuttgart in June 2023. More details on this can be found in the related deliverable D8.3. Further practical demonstrations occurred at the Automatica fair in Munich in June 2023, and a project status day event at ARENA 2036 in September 2023, and have raised great interest among the audiences.

Those methods that were not yet ready to be demonstrated, will be considered in the upcoming planned MS3 intermediate demonstration as part of deliverable D8.4, or the final MS4 demonstration in D8.5. At the end of the third period, the subsequent and final deliverable D2.3 of this work package will report on the final DARKO perception system, and discuss the latest advances of the further ongoing tasks T2.1, T2.2 and T2.3.

8 References

WP2 publications

[1] Rishabh Jain, Narunas Vaskevicius, and Thomas Brox. “Towards Self-Supervised Pre-Training of 3DETR for Label-Efficient 3D Object Detection”. In: *Workshop on Transformers for Vision at IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022.

[2] Sebastian Koch, Pedro Hermosilla, Narunas Vaskevicius, Mirco Colosi, and Timo Ropinski. “Auto3DSG: Autoencoding for 3D Scene Graph Learning via Object-Level Scene Reconstructiion”. In: *ICCV 2023 Workshop on Scene Graphs and Graph Representation Learning (SG2RL)*. 2023.

[3] Sebastian Koch, Pedro Hermosilla, Narunas Vaskevicius, Mirco Colosi, and Timo Ropinski. “SGRec3D: Self-Supervised 3D Scene Graph Learning via Object-Level Scene Reconstruction”. In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2024.

[4] Sebastian Koch, Pedro Hermosilla, Narunas Vaskevicius, Mirco Colosi, and Timo Ropinski. “Lang3DSG: Language-based contrastive pre-training for 3D Scene Graph prediction”. In: *International Conference on 3D Vision (3DV)*. 2024.

[5] Yash Goel, Narunas Vaskevicius, Luigi Palmieri, Nived Chebrolu, Kai Oliver Arras, and Cyrill Stachniss. “Semantically Informed MPC for Context-Aware Robot Exploration”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2023.

[6] Yash Goel, Narunas Vaskevicius, Luigi Palmieri, Nived Chebrolu, and Cyrill Stachniss. “Predicting Dense and Context-aware Cost Maps for Semantic Robot Navigation”. In: *IROS 2022 Workshop on Perception and Navigation for Autonomous Robotics in Unstructured and Dynamic Environments (PNARUDE)*. 2022.

[7] Dinh-Cuong Hoang, Johannes A Stork, and Todor Stoyanov. “Voting and attention-based pose relation learning for object pose estimation from 3d point clouds”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 8980–8987.

[8] Dinh-Cuong Hoang, Johannes A Stork, and Todor Stoyanov. “Context-aware grasp generation in cluttered scenes”. In: *IEEE Int. Conf. on Rob. and Autom. (ICRA)*. IEEE. 2022, pp. 1492–1498.

[9] Shih-Min Yang, Martin Magnusson, Johannes Andreas Stork, and Todor Stoyanov. “Data-driven Grasping and Pre-grasp Manipulation Using Hierarchical Reinforcement Learning with Parameterized Action Primitives”. In: *IROS 2023 Workshop on Leveraging Models for Contact-Rich Manipulation*. 2023.

[10] Shih-Min Yang, Martin Magnusson, Johannes A Stork, and Todor Stoyano. “Learning Extrinsic Dexterity with Parameterized Manipulation Primitives”. In: *arXiv preprint arXiv:2310.17785* (2023).

[11] Giuseppe Averta, Federica Barontini, Irene Valdambrini, Paolo Cheli, Davide Bacciu, and Matteo Bianchi. “Learning to Prevent Grasp Failure with Soft Hands: From Online Prediction to Dual-Arm Grasp Recovery”. In: *Advanced Intelligent Systems* 4.3 (2022).

[12] Timm Linder, Narunas Vaskevicius, Robert Schirmer, and Kai Oliver Arras. “Cross-Modal Analysis of Human Detection for Robotics: An Industrial Case Study”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[13] Francesco Pasti and Nicola Bellotto. “Evaluation of Computer Vision-Based Person Detection on Low-Cost Embedded Systems”. In: *Proceedings of the 18th International Conference on Computer Vision Theory and Applications (VISAPP)*. 2023.

Other references

[14] Timm Linder, Kilian Y. Pfeiffer, Narunas Vaskevicius, Robert Schirmer, and Kai O. Arras. “Accurate Detection and 3D Localization of Humans Using a Novel YOLO-Based RGB-D Fusion Approach and Synthetic Training Data”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020.

[15] Jason Ku, Ali Harakeh, and Steven L Waslander. “In Defense of Classical Image Processing: Fast Depth Completion on the CPU”. In: *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE. 2018, pp. 16–22.

[16] K. S. Arun, T. S. Huang, and S. D. Blostein. “Least-Squares Fitting of Two 3-D Point Sets”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9.5 (1987), pp. 698–700.

[17] Garrick Brazil, Abhinav Kumar, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. “Omni3D: A Large Benchmark and Model for 3D Object Detection in the Wild”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 13154–13164.

[18] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. “Deep Hough Voting for 3D Object Detection in Point Clouds”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019.

[19] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. “Pointnet++: Deep hierarchical feature learning on point sets in a metric space”. In: *Advances in neural information processing systems* 30 (2017).

[20] Charles R. Qi, Xinlei Chen, Or Litany, and Leonidas J. Guibas. “ImVoteNet: Boosting 3D Object Detection in Point Clouds With Image Votes”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[21] Hao Yang, Chen Shi, Yihong Chen, and Liwei Wang. “Boosting 3D Object Detection via Object-Focused Image Fusion”. In: *arXiv preprint arXiv:2207.10589* (2022).

[22] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. “Deep Layer Aggregation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[23] Yunzhi Lin, Jonathan Tremblay, Stephen Tyree, Patricio A. Vela, and Stan Birchfield. “Single-Stage Keypoint- Based Category-Level Object Pose Estimation from an RGB Image”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 1547–1553.

[24] Dennis Nienhüser. *annotate*. https://github.com/earthwings/annotate. 2021.

[25] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. “Open3D: A Modern Library for 3D Data Processing”. In: *arXiv:1801.09847* (2018).

[26] Ishan Misra, Rohit Girdhar, and Armand Joulin. “An End-to-End Transformer Model for 3D Object Detection”. In: *ICCV*. 2021.

[27] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J Kusner. “Unsupervised point cloud pre-training via occlusion completion”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9782–9792.

[28] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin Zhu. “Spatio-temporal self-supervised representation learning for 3d point clouds”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 6535–6545.

[29] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. “Barlow twins: Self-supervised learning via redundancy reduction”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 12310–12320.

[30] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. “SUN RGB-D: A rgb-d scene understanding benchmark suite”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 567–576.

[31] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone. “Kimera: from SLAM to Spatial Perception with 3D Dynamic Scene Graphs”. In: *Intl. J. of Robotics Research* 40.12–14 (2021). arXiv preprint: 2101.06894, pp. 1510–1546.

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. “Learning Transferable Visual Models From Natural Language Supervision”. In: *CoRR* abs/2103.00020 (2021). arXiv: 2103.00020.

[33] Bosch Corporate Research Semantic SLAM Team. “Panoptic hierarchical Semantic SLAM”. In: *CVPR Embodied AI Workshop*. 2022.

[34] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. “Segment Anything”. In: *arXiv:2304.02643* (2023).

[35] Grounded-SAM Contributors. *Grounded-Segment-Anything*. Apr. 2023.

[36] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. “OpenMask3D: Open-Vocabulary 3D Instance Segmentation”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023.

[37] Kilian Kleeberger, Christian Landgraf, and Marco F Huber. “Large-scale 6d object pose estimation dataset for industrial bin-picking”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 2573–2578.

[38] Romain Brégier, Frédéric Devernay, Laetitia Leyrit, and James L Crowley. “Symmetry aware evaluation of 3d object detection and pose estimation in scenes of many parts in bulk”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 2209–2218.

[39] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. “Dual attention network for scene segmentation”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 3146–3154.

[40] Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu. “Attentional shapecontextnet for point cloud recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4606–4615.

[41] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. “Graspnet-1billion: A large-scale benchmark for general object grasping”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11444–11453.

[42] Wenxuan Zhou and David Held. “Learning to Grasp the Ungraspable with Emergent Extrinsic Dexterity”. In: *Conference on Robot Learning (CoRL)*. 2022.

[43] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. “Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation”. In: *Advances in neural information processing systems* 29 (2016).

[44] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor”. In: *International conference on machine learning*. PMLR. 2018, pp. 1861–1870.

[45] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. “Rainbow: Combining improvements in deep reinforcement learning”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[46] Yuan Liu, Yilin Wen, Sida Peng, Cheng Lin, Xiaoxiao Long, Taku Komura, and Wenping Wang. “Gen6D: Generalizable model-free 6-DoF object pose estimation from RGB images”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 298–315.

[47] Johannes Lutz Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[48] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *European Conference on Computer Vision (ECCV)*. 2016.

[49] Visar Arapi, Yujie Zhang, Giuseppe Averta, Manuel G Catalano, Daniela Rus, Cosimo Della Santina, and Matteo Bianchi. “To grasp or not to grasp: an end-to-end deep-learning approach for predicting grasping failures in soft hands”. In: *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE. 2020, pp. 653–660.

[50] Rahul Dey and Fathi M Salem. “Gate-variants of gated recurrent unit (GRU) neural networks”. In: *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. IEEE. 2017, pp. 1597–1600.

[51] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. “Yale-CMU-Berkeley dataset for robotic manipulation research”. In: *The International Journal of Robotics Research* 36.3 (2017), pp. 261–268.

[52] Brian T Hinson, Michael K Binder, and Kristi A Morgansen. “Path planning to optimize observability in a planar uniform flow field”. In: *2013 American Control Conference*. IEEE. 2013, pp. 1392–1399.

[53] Paolo Salaris, Marco Cognetti, Riccardo Spica, and Paolo Robuffo Giordano. “Online optimal perception-aware trajectory generation”. In: *IEEE Transactions on Robotics* 35.6 (2019), pp. 1307–1322.

[54] Nikos Mavrakis and Rustam Stolkin. “Estimation and exploitation of objects’ inertial parameters in robotic grasping and manipulation: A survey”. In: *Robotics and Autonomous Systems* 124 (2020), p. 103374.

[55] Wouter M Bergmann Tiest and Astrid ML Kappers. “Haptic perception of gravitational and inertial mass”. In: *Attention, perception, & psychophysics* 72.4 (2010), pp. 1144–1154.

[56] István Sárándi, Timm Linder, Kai O. Arras, and Bastian Leibe. “MeTRAbs: Metric-Scale Truncation-Robust Heatmaps for Absolute 3D Human Pose Estimation”. In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 3.1 (2021), pp. 16–30.

[57] Edward Vendrow, Duy Tho Le, Jianfei Cai, and Hamid Rezatofighi. “JRDB-Pose: A Large-scale Dataset for Multi-Person Pose Estimation and Tracking”. In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2023.

[58] István Sárándi, Alexander Hermans, and Bastian Leibe. “Learning 3D Human Pose Estimation from Dozens of Datasets using a Geometry-Aware Autoencoder to Bridge Between Skeleton Formats”. In: *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2023.

[59] Yahui Zhang, Shaodi You, Sezer Karaoglu, and Theo Gevers. “Multi-person 3D pose estimation from a single image captured by a fisheye camera”. In: *Computer Vision and Image Understanding* 222 (2022), p. 103505.

[60] Helge Rhodin, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt. “EgoCap: egocentric marker-less motion capture with two fisheye cameras”. In: *ACM Trans. Graph.* 35.6 (2016), 162:1–162:11.

[61] Weipeng Xu, Avishek Chatterjee, Michael Zollhoefer, Helge Rhodin, Pascal Fua, Hans-Peter Seidel, and Christian Theobalt. “Mo^2Cap^2 : Real-time Mobile 3D Motion Capture with a Cap-mounted Fisheye Camera”. In: *IEEE Transactions on Visualization and Computer Graphics* (2019).

[62] Kohei Aso, Dong-Hyun Hwang, and Hideki Koike. “Portable 3D Human Pose Estimation for Human-Human Interaction Using a Chest-Mounted Fisheye Camera”. In: *Proceedings of the ACM Augmented Humans International Conference 2021*. 2021, pp. 116–120.

[63] Koji Minoda and Takehisa Yairi. “3D Human Pose Estimation in Weightless Environments Using a Fisheye Camera”. In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. 2022.

[64] Hiroyasu Akada, Jian Wang, Soshi Shimada, Masaki Takahashi, Christian Theobalt, and Vladislav Golyanik. “UnrealEgo: A New Dataset for Robust Egocentric 3D Human Motion Capture”. In: *European Conference on Computer Vision (ECCV)*. 2022.

[65] Jialian Li, Jingyi Zhang, Zhiyong Wang, Siqi Shen, Chenglu Wen, Yuexin Ma, Lan Xu, Jingyi Yu, and Cheng Wang. “LiDARCap: Long-Range Marker-Less 3D Human Motion Capture With LiDAR Point Clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 20502–20512.

[66] Zhenzhen Weng, Alexander S. Gorban, Jingwei Ji, Mahyar Najibi, Yin Zhou, and Dragomir Anguelov. “3D Human Keypoints Estimation From Point Clouds in the Wild Without Human Labels”. In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 1158–1167.

[67] Dongqiangzi Ye, Yufei Xie, Weijia Chen, Zixiang Zhou, and Hassan Foroosh. *LPFormer: LiDAR Pose Estimation Transformer with Multi-Task Network*. 2023. arXiv: 2306.12525 [cs.CV].

[68] Pei Sun et al. “Scalability in Perception for Autonomous Driving: Waymo Open Dataset”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[69] Wonhui Kim, Manikandasriram Srinivasan Ramanagopal, Charles Barto, Ming-Yuan Yu, Karl Rosaen, Nick Goumas, Ram Vasudevan, and Matthew Johnson-Roberson. “PedX: Benchmark Dataset for Metric 3D Pose Estimation of Pedestrians in Complex Urban Intersections”. In: *IEEE Robotics and Automation Letters (RA-L)* 4.2 (Apr. 2019), pp. 1940–1947.

[70] Yudi Dai, Yitai Lin, Xiping Lin, Chenglu Wen, Lan Xu, Hongwei Yi, Siqi Shen, Yuexin Ma, and Cheng Wang. “SLOPER4D: A Scene-Aware Dataset for Global 4D Human Pose Estimation in Urban Environments”. In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 682–692.

[71] Bohao Fan, Siqi Wang, Wenxuan Guo, Wenzhao Zheng, Jianjiang Feng, and Jie Zhou. *Human-M3: A Multi-view Multi-modal Dataset for 3D Human Pose Estimation in Outdoor Scenes*. 2023. arXiv: 2308.00628 [cs.CV].

[72] Vladyslav Usenko, Nikolaus Demmel, and Daniel Cremers. “The Double Sphere Camera Model”. In: *International Conference on 3D Vision (3DV)*. 2018. eprint: http://arxiv.org/abs/1807.08957.

[73] Kenji Koide, Shuji Oishi, Masashi Yokozuka, and Atsuhiko Banno. “General, Single-shot, Target-less, and Automatic LiDAR-Camera Extrinsic Calibration Toolbox”. In: *IEEE Int. Conf. Robot. & Autom. (ICRA)*. IEEE. 2023.

[74] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. “ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation”. In: *Advances in Neural Information Processing Systems*. 2022.

[75] Zhuofan Zong, Guanglu Song, and Yu Liu. “DETRs with Collaborative Hybrid Assignments Training”. In: *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*. 2023.

[76] XRMoCap Contributors. *OpenXRLab Multi-view Motion Capture Toolbox and Benchmark*. https://github.com/openxrlab/xrmocap. 2022.

[77] Soumava Kumar Roy, Leonardo Citraro, Sina Honari, and Pascal Fua. “On Triangulation as a Form of Self-Supervision for 3D Human Pose Estimation”. In: *2022 International Conference on 3D Vision (3DV)*. 2022.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017274