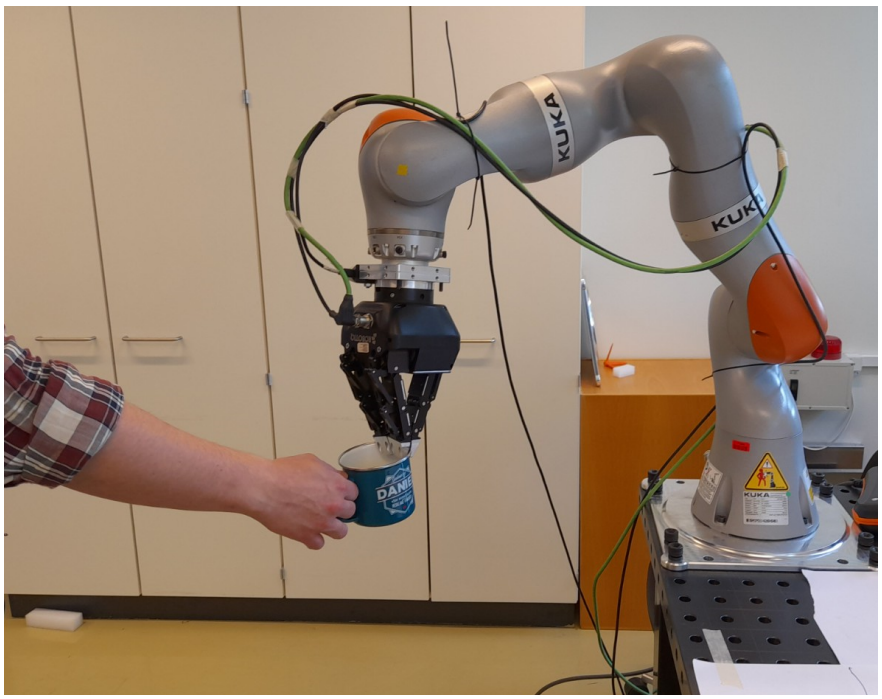
Generating appropriate object orientations for robot-to-human handovers using synthetic object affordances



10th semester project

Group 1064

Aalborg University
Electronics and IT

AALBORG UNIVERSITY
STUDENT REPORT

Robotics
Aalborg University
Department of Electronic Systems
Fredrik Bajers Vej 7B
DK-9220 Aalborg
http://www.robotics.aau.dk

Title:
Generating appropriate object orientations for robot-to-human handovers using synthetic object affordances

Theme:
MSc. thesis

Project Period:
Spring Semester 2022

Project Group:
1064

Participant(s):
Albert D. Christensen
Daniel Lehotský

Supervisor(s):
Dimitris Chrysostomou

Copies: 1

Page Numbers: 90

Date of Completion:
June 1, 2022

Abstract:

This project is an investigation into applying object affordances to robot-to-human handovers. Our research makes two contributions. A state-of-the-art deep neural network for segmentation of object affordances named AffNet-DR, trained solely on synthetic data. Secondly, an object affordance enabled method for orienting objects appropriately for robot-to-human handovers. A user study with 6 participants showed that our method for computing handover orientations outperforms a method that uses random orientations. Finally, a robotic handover system was programmed in ROS Melodic and implemented on a KUKA LBR iiwa 7 R800 with an Intel RealSense D435i RGB-D sensor and a Robotiq 3-finger gripper. The system performs robot-to-human handovers with a success rate of 91.67 %.

Preface

This master thesis was produced in the period from the 1st of February 2020 to the 2nd of June 2022 at Aalborg University as part of the Robotics master program. The thesis was supervised by associate professor Dimitris Chrysostomou from the department of Materials and Production, Aalborg University.

The aim of this thesis is to develop a pipeline that allows for task-oriented robot-to-human handovers using object affordances rather than object classes. Moreover, the pipeline should provide proper orientations at the moment of handover. In order to detect affordances, a synthetic dataset generator was developed, and synthetic data produced by the aforementioned generator were used to a train neural network called AffNet-DR. Part of this work has been submitted to IROS 2022 with the title "Learning to Segment Object Affordances on Synthetic Data for Task-oriented Robotic Handovers", where we present the neural network AffNet-DR and our synthetic dataset generator.

As this master thesis concludes our time as students at Aalborg University, the authors of this master thesis would like to offer their thanks to all the people that helped and supported us throughout our studies. We would like to thank our families, friends, and significant others for emotional support. Next, we would like to thank our supervisors and professors for all the knowledge and experience they shared with us. Furthermore, there is Aalborg University which provided us with all the needed space and equipment to realize our research. Lastly, we also offer our thanks to all the people who participated in our user study and dataset collection.

June 1, 2022, Aalborg University

Albert D. Christensen
<alchri17@student.aau.dk>

Daniel Lehotský
<dlehot17@student.aau.dk>

Contents

1 - Introduction

Robotic handovers are essential to enable robots and humans to perform collaborative tasks together. The handing over of diverse tools and objects could see applications ranging from disassembly applications [1], and assistive surgery [2], to the service industry [3].

A handover is a joint action where an object is transferred from a giver to a receiver for a specific task. To successfully perform such joint actions, the ability to share representations, predict actions, and integrate the predicted effects is needed [4]. This comes naturally to humans, and people, therefore, excel at handovers. However, handovers in the field of robotics remains a field of active research [5, 6, 7].

Whereas humans have an intuitive understanding of how to perform a handover, a robotic system must solve the constituting sub-problems of the handover task. This requires that the robotic system can detect objects in its surroundings, perform precise grasping of said objects, and plan a trajectory that transports the object to the human receiver [8].

Humans are naturally good at anticipating and adjusting their behaviour to that of their handover partner. In order to optimize and achieve a smooth handover, a robot will need to generate motions that allow a human to anticipate and adapt to the actions of the robot [9].

The tasks of the giver and the receiver has shown to influence where and how objects are grasped as well as the object orientation during the handover [8, 10, 11]. A robotic handover system needs to consider the subsequent task of the receiver in order to generate appropriate handovers. Therefore, objects and tools must be grasped and oriented using task-oriented methods.

Grasping an object appropriately for a given task requires that the functional parts of an object are identified [12]. Likewise, proper object orientations during handovers have shown to be dependent on the functionalities inherent to the object [11]. However, current methods for computing object orientations suitable for a handover task are either limited to pre-defined rules [13, 14, 15] or object classes such as a hammer, mallet, saw, knife, etc [16, 10, 11]. Therefore, current object orientation methods either do not consider the functionalities of the object or only consider them implicitly by knowing the object class.

Humans are capable of utilizing their semantic understanding of the functional properties of an object to perform task-oriented grasping [17] and handover object orienta-

tion. Robotic systems lack this semantic understanding. However, affordance theory, as first proposed by Gibson [18], can be used for understanding the semantic and functional properties of objects [19, 20, 21, 22, 23, 7].

Object affordances refer to the functionalities that an object facilitates irrespective of the current state of the object [24]. Objects can offer several different affordances, with each constituting part having a different affordance. This, therefore, allows object affordance detection to be treated as a pixel-wise segmentation problem [19, 25], see Figure 1.1.

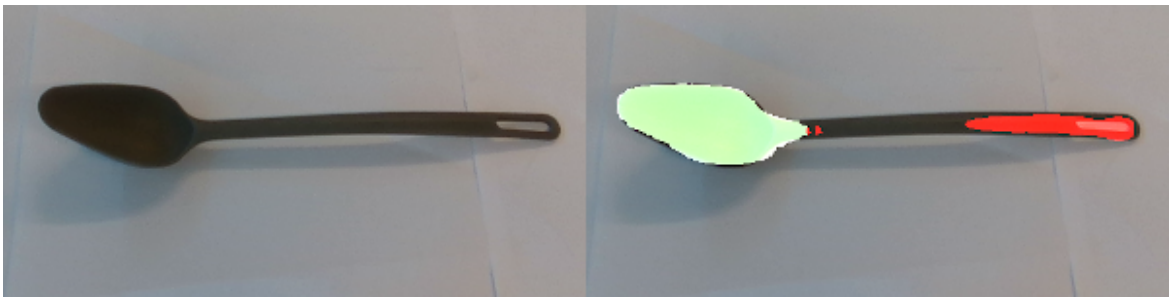


Figure 1.1: Pixel-wise object affordance detection performed on a spoon. Lime green is the scoop affordance. Red indicates the grasp affordance.

Detecting object affordances with neural networks has become the norm since Nguyen et al. [20] outperformed previous methods using a deep neural network. Neural networks require a vast amount of data to learn a given task [26]. Annotating datasets for affordance segmentation requires pixel-wise labelling, which is a resource-expensive task. As such, current object affordance datasets are of a limited size and scope [19, 21]. Synthetic data is considerably cheaper to generate, as the labelling and annotation of the data can be automated [27]. Synthetic data, however, suffers from what is known as the sim2real gap, where the neural network can not generalize to real-world data when trained on synthetic data [28]. Only limited work and investigations has been conducted into using synthetic data for object affordance detection [29, 30].

Domain randomization has been proposed as a method for overcoming the sim2real gap. Domain randomization suggests that varying parameters such as illumination, object poses, textures, environment, and etc. when generating the synthetic data would make the real-world data appear as just another variation to a system trained on synthetic data generated with domain randomization [28].

With this work, the authors propose a novel framework for generating synthetic data for object affordance detection using domain randomization. A Mask-RCNN like neural network named AffNet-DR is then trained on the synthetic data and evaluated on real-world data. Afterwards, the method is validated on a real-world robotic system, where AffNet-DR is implemented into a robot-to-human handover pipeline.

Furthermore, we show that proper handover orientations only need to consider the affordances of an object. Two methods for computing handover orientations based solely on affordances are, therefore, also proposed with this report.

With this work, the following contributions are made to the field of robotic handovers:

- A robot-to-human handover system that can incorporate task-oriented grasping, handover location, and proper handover orientation of objects.
- A state-of-the-art deep neural network named AffNet-DR capable of segmenting object affordances, trained solely on synthetic data.
- A novel synthetic data generator for object affordance detection. Data is generated using domain randomization principles, which we show is capable of overcoming the sim2real gap and enabling real-world robot-to-human handovers.Figure 3.10 on page 41 shows a sample from the dataset. The dataset itself can be found on our GitHub [31].
- A synthetic object affordance dataset, available on the project's GitHub [31].
- A method for learning handover orientations from observations based solely on object affordances.
- A novel rule-based method for computing handover orientations by only considering the object affordances.

This work is a continuation of a 9th semester [32]. The main contributions of the previous work were:

- A pipeline for task-oriented handovers based on object affordances rather than object classes.
- A method for associating task-agnostic tasks with affordances, effectively transforming task-agnostic grasps into task-oriented grasps.

This pipeline was developed on a setup consisting of 6-DOF UR5 robot, a Robotiq 3-finger gripper, an Intel RealSense D435 RGB-D sensor.

The previous work is expanded upon by also considering what is known as the handover location and orientation. The object affordance segmentation is improved upon by utilizing synthetic data. The robot-to-human handover pipeline has been implemented onto a robotic system consisting of a 7-DOF KUKA LBR iiwa 7 R800 robot, a Robotiq 3-finger gripper, an Intel RealSense D435i RGB-D sensor, and a Hokuyo URG-04LX-UG01 laser scanner. UR5 robot was replaced in favor of 7-DOF KUKA iiwa in order to achieve more challenging end-effector poses.

2 - Problem Analysis

The following section analyzes and defines the robot-to-human handovers, and what constitutes an appropriate robot-to-human handover. Later in the section, specific areas related to robot-to-human handovers are investigated more in depth.

2.1 Defining robot-to-human handovers

A handover is defined as a joint action between two agents, a giver and a receiver, where the giver's goal is to present the object as stably as possible to the receiver, and the receiver's goal is to obtain and stabilize the object [33], thereby transferring an object of interest from the giver to the receiver [8]. For humans, handovers are natural and easy. However, in robotics this remains an unsolved problem as robots still struggle to perform handovers. This section examines what makes up a handover and what factors influence a handover between robots and humans.

2.1.1 Phases of the handover

A handover can be split into four phases - initiation, pre-handover, physical handover, and the performance phase [8], see Figure 2.1. Each phase has its own set of intricacies that the system should consider when performing the handover.

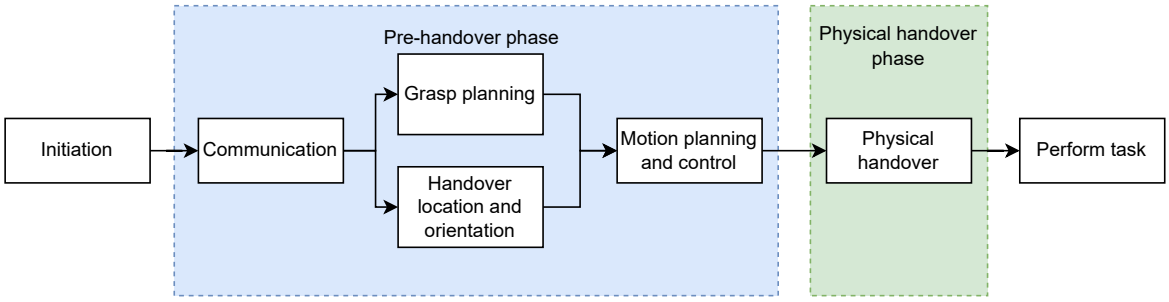


Figure 2.1: Block diagram of a handover process [32].

At first, the handover is initiated. This can be initiation by an object request, which is when the receiver requests an object, or initiation by a task request, which is when the giver requests the receiver to perform a specific task by handing the receiver an object [8]. An example of the initiation by an object request is a person verbally asking for a tool. This report focuses on initiation by object request where the human is always the receiver and initiator and the robotic system is always the giver. This type of handover is known as robot-to-human handover.

The initiation phase is followed by the pre-handover phase. The pre-handover phase encapsulates everything that happens after the initiation and before the physical handover. During this phase, the giver has to plan a trajectory that moves the handover object from its initial position to what is known as the handover location. The trajectory of the giver can be decomposed into segments dealing with grasping of the object, handover location and orientation, as well as trajectory properties, such as velocity and object orientation constraints. All of these aspects are influenced by communication between the giver and the receiver, which can be either verbal or non-verbal, such as eye-gaze and body language [8]. Ortenzi et al. have shown that the perceived quality of the handover increases if the giver also considers the subsequent task of the receiver, in what is known as task-oriented handovers [34]. Task-oriented handovers influence how the giver should grasp the object as well as place and orient it in regards to the receiver amongst other things. For example, if the receiver asks for a hammer, the giver should hand it over with the handle unobstructed and should place and orientate the hammer in such a way that is comfortable for the receiver to reach. The handover phase ends when the receiver makes physical contact with the handover object.

The physical handover phase begins once the receiver makes contact with the handover object [8]. During this phase, the giver continuously decreases the grip force as the receiver simultaneously increases their grip on the handover object. If the giver relaxes the grip force too soon, the object falls to the ground. A smooth transfer of the handover object from the giver to the receiver is, therefore, enabled by correctly timing the grip force modulation between the giver and the receiver. After the physical handover is complete, the receiver will proceed with the task, starting the performance phase of the handover [8].

2.1.2 Planning the pre-handover phase trajectory

As was already mentioned, one of the giver's considerations is the receiver's task. Ideally, the giver hands over the object in a way that is not only comfortable for the receiver but also allows the receiver to continue in the task without any interruptions. This is known as a task-oriented handover, and strongly influences the trajectory that has to be computed in the pre-handover phase, see Figure 2.2. Task-oriented handovers, therefore, influence how to grasp the object, where to place it, and how to orient the object in regards to the receiver, amongst other considerations. This section investigates the trajectory of the pre-handover phase.

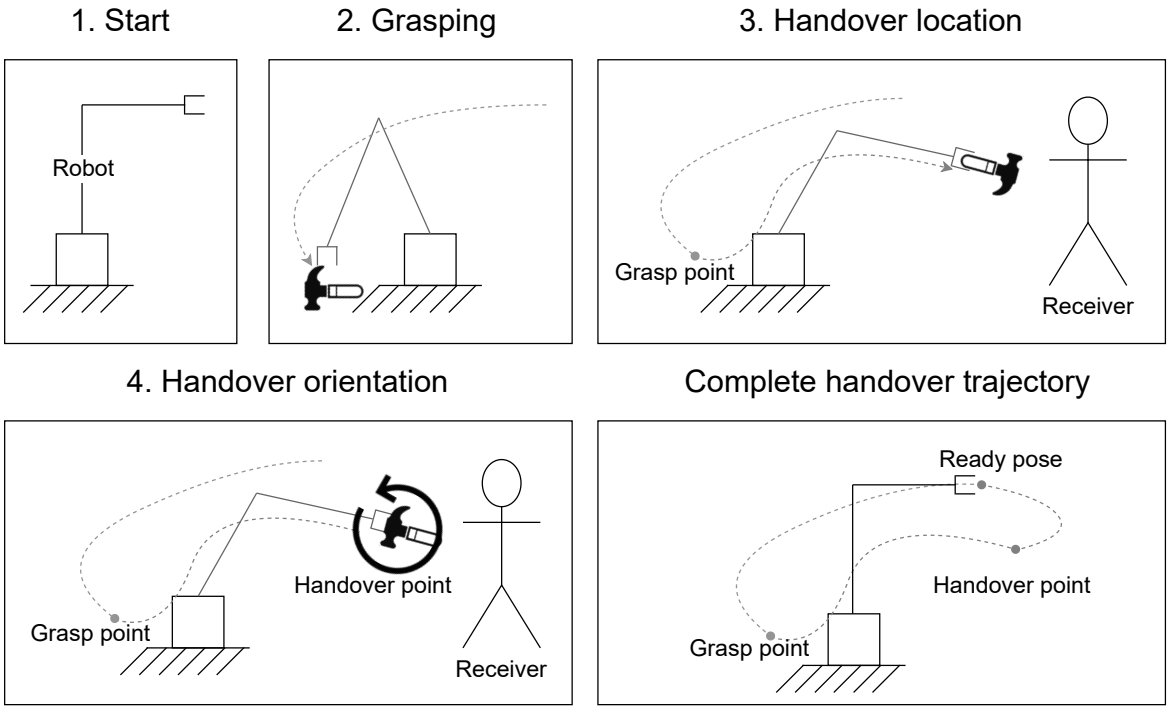


Figure 2.2: A visualization of the trajectory that is planned for in the pre-handover phase.

Grasp point considerations

Task-oriented handovers are tightly coupled with task-oriented grasping [34]. Unlike task-agnostic grasping, which only considers the stability of the grasp, task-oriented grasping is the task of selecting a grasp that is not only stable but also suitable for the receiver's task and is aware of the functionalities that the individual object parts offers. As an example, a human receiver might prefer to grasp the handle of a hammer, and as such, the robotic giver should grasp the object in a way that leaves the handle free and available to the receiver. In terms of the trajectory presented in Figure 2.2, the task is to find the grasp point.

Detry et al. [12] suggest that task-oriented grasping can be solved by solving two sub-problems. At first, the part of an object which is suitable for the specific task must be identified. Afterwards, a mechanically stable grasp, suitable for the specific task, must be generated. It has been shown that humans utilize their semantic understanding of the object's functional properties to grasp items appropriately for a given task [17]. Robots can not do the same as they lack such knowledge. However, affordance theory, which is concerned with understanding the functional properties of objects, has proven a popular approach to the first sub-problem of task-oriented grasping [20, 21, 12, 35, 23, 36, 7]. The concept of affordances is further explained in section 2.2.

Task-oriented grasping was addressed in our previous work [32], where we showed how affordance theory could help solve task-oriented grasping. It was solved by segmenting the affordances of an object in an RGB image using AffordanceNet proposed by [22] and sampling task-agnostic grasps in a point cloud using grasp generator proposed by [37]. Task-agnostic grasps were turned into task-oriented grasps by determining which affordance they would grasp, which was done by examining their direction vectors, see Figure 2.3. The experiments showed that the affordance analyzation method used in our previous work was challenged by varied illumination conditions and object poses. For more details on the implementation of both the grasping and the grasp-association module, see the previous work [32].

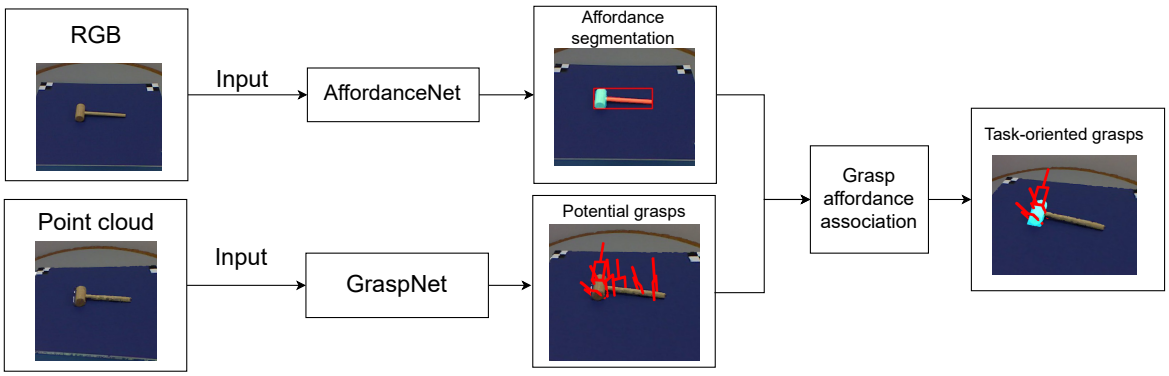


Figure 2.3: Task-oriented grasps are found by combining affordance theory with sampled task-agnostic grasps [32].

Handover location considerations

The handover location is the task of finding the position in the 3D space that is the most suited for the physical handover phase to take place, see step 3 in Figure 2.2.

Hansen et al. [38] show that the handover location in human-to-human handovers happens at the midpoint in the shared workspace and is independent of an object mass. The handover location is chosen dynamically and adapts to when the giver and receiver moves [39].

Choosing the right handover location can help the receiver understand an intent to handover an object [40] as well as increase the perceived safety and comfort of the receiver [9]. Further considerations can be made to the physical strains imposed on the receiver during the handover in order to alleviate said strains [41]. Such considerations can be valuable when interacting with motor-impaired people.

A robotic handover system that can generate suitable handover locations can help a robotic system to perform smooth handovers.

Handover orientation considerations

The orientation of an object at the transition from the pre-handover phase to the physical handover phase is known as the object handover orientation. At this point in the handover, the object is passed with a specific orientation. The position and orientation of an object during a handover has shown to affect the efficiency as well as perceived safety of the handover [9, 42]. The giver can convey an intent to perform a handover by the orientation of the object [42, 14]. Proper handover orientation can, therefore, help achieve a more smooth handover.

Proper handover orientation of various objects depends on the object's geometry, physical properties, functions, as well as object affordances [42, 10, 11]. Human-to-human user studies have shown that people will pass an object with the handle available to the receiver, when handing over objects with a grasp affordance [43, 44].

Computing object handover orientation can, therefore, improve the perceived handover. Additionally, affordance theory might be able to help solve the object handover orientation problem.

Additional considerations

The grasping point, the handover location, and the handover orientation are only a few out of many considerations of the pre-handover trajectory planning. There exist more elements that could be considered, and some of these will be discussed here.

Ortenzi et al. [8] recommend robot trajectories for robot-human handovers to be legible, predictable, safe, robust, reactive, and context-aware. A study by Cakmak et al. [13] showed that human-like configurations of the robot are more readable and, therefore, preferred by the humans. In a study by Huber et al. [45], study participants demonstrated faster reactions and an increased sense of subjective safety when co-working with the robot following a minimum jerk velocity profile of the end-effector. Moreover, using the minimum jerk velocity profile, the receiver's comfort levels remained the same even with the robot moving at a higher speed. The authors of the study suggest that the minimal jerk velocity profile is more similar to the human velocity profile than the commonly used trapezoidal velocity profile in the joint space. Because of the similarity, the study participants felt safer when cooperating with the robot. Therefore, human-like trajectories increase the legibility and predictability of the trajectories along with the user's feeling of safety.

Another constraint of the trajectory planning might be the handover object itself. Individual items might require individual approaches. For example, handing over a cup or a bottle without a cap should be done without spilling it. Handing over a knife should be done without endangering the user in any way. A similar idea applies to dull but still possibly dangerous items, such as hammers or screwdrivers.

As was already mentioned, communication in both verbal and non-verbal forms is crucial during handovers. It can be used for initiation but also to further coordinate the

handover action once it started [46]. For the case of trajectory planning, gaze and body language are important factors. Both Moon et al. [47] and Boucher et al. [48] provided evidence that enhancing the robot system with human-like gaze abilities improves the reaction times of the humans during the handover. In order to improve the naturalness of the handover, Gharbi et al. [49] identified two gaze patterns that achieve similar levels of naturalness, as perceived by the human receiver, regardless if done by the robot or the human giver. With the first pattern, the giver initially looks at the object and then moves its gaze from the object to the receiver once the handover object reached the handover location. With the second pattern, the giver looks at the receiver, then at the handover object, and then back at the receiver once the handover object is at the handover location. The act of looking at the object and then at the receiver once the object reaches the handover location was identified as the most important gaze pattern.

Body language, such as stance, arm pose, and gestures, can also be used to enhance the handover experience. Cakmak et al. [42] identified the failure to convey the intention of handing over and the lack of intuitive signalling to indicate the timing of the handover as the two main reasons for delays in the transfer of the handover object. In order to alleviate these problems, they propose to design handover trajectories using spatial and temporal contrast in the robot's poses. A spatial contrast refers to how recognizable is the pose in which the robot is presenting the item to the receiver compared to all the other poses. A temporal contrast refers to how recognizable is the pose in which the robot is presenting the item to the receiver compared to the robot's previous pose. However, after performing an experiment with study participants, only the temporal contrast is shown to improve the fluency of the handover by removing early attempts at the handover. Nevertheless, the authors of the study argue that the spatial contrast was not properly tested in their experimental settings, thus the effect of the spatial contrast can not be evaluated. Still, the authors noticed that when the robot extended its arm, several human receivers misunderstood this as the invitation to take the object from the robot. This study shows how the body language of the robot induces different behaviour from humans.

The additional considerations covered here show just some of the challenges and intricacies of the pre-handover phase. The pre-handover phase trajectory can be improved upon by allowing it to adapt dynamically to the signals sent by the receiver, or by being more predictable among other considerations. However, in order to perform a basic handover, the grasp point, handover location and orientation must be solved first, thereby, making them the most immediate problems to solve.

2.1.3 Improving upon the pre-handover trajectory

The main focus of this report is the generation of the pre-handover phase trajectory, with special attention given to the planning and computation of the handover orientation. Proper handover orientation and position of various objects depend on the object's geometry, physical properties, functions as well as object affordances [42, 10, 11]. It is, therefore,

worth investigating if object affordance analysis can help solve the issue of object handover orientation.

However, as was shown in [32] and also discussed in section 2.2, current object affordance analysis methods do not generalize very well into the real world as issues in regards to object pose and illumination conditions were observed in our previous work [32]. Therefore, this report also investigates if the current object affordance analysis methods can be made more robust to object pose and illumination condition challenges.

2.2 Affordance analysis and object affordance segmentation

Originally proposed by Gibson [18], affordance analysis is the theory that various objects and places allow for various actions depending on the agent. This report focuses on the zero-order object affordances, which is a set of functionalities that the object provides regardless of the object's current state [24]. Because different parts of an object provide different affordances, object affordance detection can be addressed as a pixel-wise segmentation problem [25], which will be referred to as the affordance segmentation throughout the rest of this report. Therefore, the task is to classify each pixel with its affordance. As such, affordance segmentation is most often solved in the vision domain with data obtained from RGB sensors. An example of the pixel-wise affordance segmentation can be seen in Figure 2.4.

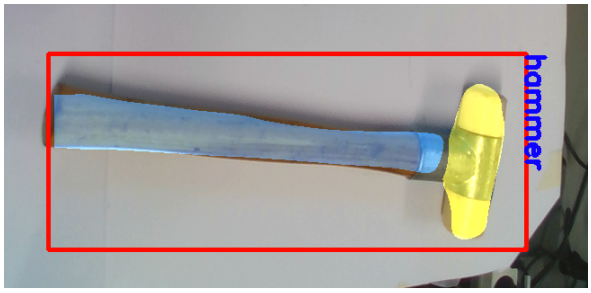


Figure 2.4: An example of pixel-wise segmentation of the hammer. The yellow pixels represent pound affordance. The blue pixels are the grasp affordance.

2.2.1 Evaluation

The weighted F_β^w-measure metric, see equation (2.1), is the most commonly used performance metric when evaluating the performance of affordance segmentation methods. The weighted F_β^w-measure metric, proposed by Margolin et al. [50], expands on F_β-measure by extending the definition of True Positives, True Negatives, False Positives, and False Negatives to non-binary values, which allowed for weighting the error based on its location and neighbourhood.

$$F_{\beta}^{w} = (1+\beta)\frac{Precision^{w}.Recall^{w}}{\beta^{2}.Precision^{w} + Recall^{w}} \tag{2.1}$$

Although multiple affordance segmentation datasets exist, see Table 2.1, the affordance datasets are more limited in size and scope compared to datasets commonly used for object detection, segmentation, and classification. This is due to how resource-expensive it is to annotate images with ground-truth affordances. Out of a few affordance datasets, the UMD [19] and IIT-AFF [21] datasets have become the norm for benchmarking affordance segmentation methods, with the IIT-AFF dataset being the more challenging out of the two.

The UMD dataset [19] contains manually annotated, pixel-wise affordance labels of 105 tools captured in a calibrated environment. The calibrated environment only provides limited variation in the background texture, lighting, and camera viewpoint. Moreover, the dataset provides only a few examples for some of the object categories, further decreasing the variability of the data. A sample from the UMD dataset can be seen in Figure 2.5.

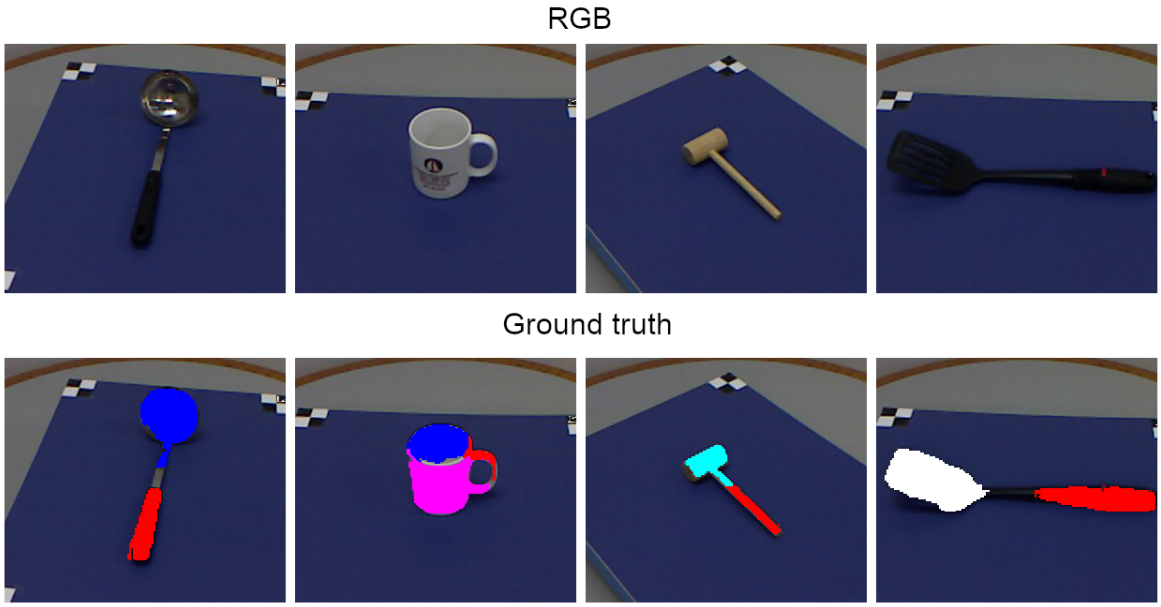


Figure 2.5: A sample of images and their respective ground truths from the UMD dataset.

In order to introduce more diversity and variation, the IIT-AFF dataset [21] also included a subset of images from ImageNet. This made IIT-AFF more challenging by introducing occlusions with a higher variety of background textures and lighting conditions. A sample from the IIT-AFF dataset can be seen in Figure 2.6.

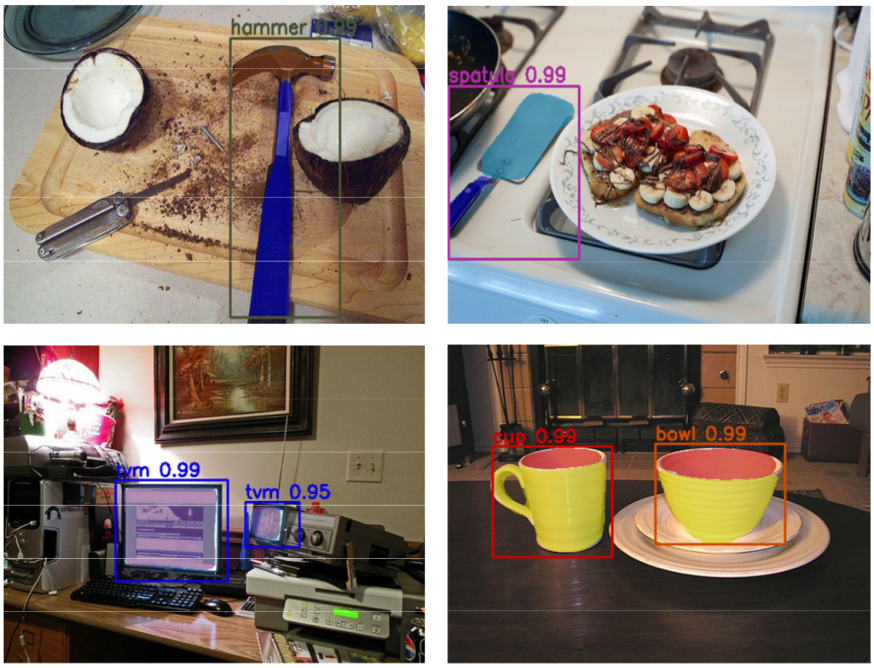


Figure 2.6: A sample from the IIT-AFF dataset. Taken from [22].

Dataset	Year	Size	Modality	Labels
UMD [19]	2015	10,000	RGB-D	*grasp, cut, scoop, contain, pound, support, wrap-grasp*
IIT-AFF [21]	2017	24,677	RGB-D	*contain, cut, display, engine, grasp, hit, pound, support wrap-grasp*
CAD-120-Affordance [51]	2017	9916	RGB	*open, cut, contain, pour, support, hold*
3D-AffectNet [52]	2021	23,000	Point cloud	*grasp, lift, contain, open, lay, sit, support wrap-grasp, pour dislay, push, pull listen, wear, press cut, stab, move*

Table 2.1: A list of affordance segmentation datasets.

2.2.2 Related work concerning improvements to affordance segmentation architectures

This section covers related works that focus on improvements to the architectures of the affordance segmentation methods. The section is split into two parts - performance improvements and the inference time improvements.

Performance improvements

Affordance detection using visual data has been investigated and used in many robotic applications [53, 54, 55]. In 2015, Myers et al. [19] introduced the UMD dataset and were the first researchers to cast the affordance detection as the pixel-wise segmentation problem. Using their own UMD dataset, they trained two solutions - the S-HMP and SRF. These reached F_β^w scores of 0.557 and 0.466 respectively.

Deep learning methods have been shown to exceed the performance of the traditional affordance segmentation methods. Nguyen et al. were approached the affordance segmentation problem with a deep learning approach. Their encoder-decoder solution outperformed previously proposed methods [20], achieving a F_β^w score of 0.770 on the UMD dataset. In 2020, Abdalwhab et al. [56], similarly to Nguyen et al., used a modified SegNet encoder-decoder to perform affordance segmentation. By including skip connections, Abdalwhat et al. managed to achieve a F_β^w score of 0.707 on the IIT-AFF dataset.

Nguyen et al. [21] showed that casting the affordance prediction problem as an instance segmentation problem can achieve higher accuracy. Using a modified Faster-RCNN network in combination with a conditional-random-field as a post-processing step, their method named BB-CNN-CRF achieved a F_β^w score of 0.696 on the IIT-AFF dataset.

In 2018, Do et al. [22] achieved state-of-the-art results on the IIT-AFF and UMD datasets, with F_β^w scores of 0.734 and 0.799 respectively, using their modified Mask-RCNN network called AffordanceNet. This network has since become the baseline that many other methods compare themselves against [57, 58, 59, 60, 61, 62, 63, 64].

In 2020, Minh et al. [58] and Qian et al. [59] experimented with the AffordanceNet architecture by examining the influence of the quality of the features used. This was done by switching the feature extractor from VGG16 to various ResNets. Minh et al. also made further changes to the alignment layer. Both of the approaches saw incremental improvements over AffordanceNet, with Minh et al. achieving a F_β^w score of 0.802 on the IIT-AFF dataset, and Qian et al. achieving a F_β^w score of 0.814 on the UMD dataset.

Attention modules have lately become an area of interest for affordance segmentation networks [57, 60, 61, 64]. Common to these approaches is that they do not perform instance segmentation. Instead, the methods rely on encoder-decoder architectures for segmentating affordances, without performing object detection. In 2019, Zhao et al. [57] proposed a model that implements both attention and relationship-aware modules. Zhao et al. argue that affordances and objects have a symbiotic relationship of which the pro-

posed relationship-aware module can take advantage of. Zhao et al. achieved a F_β^w score of 0.861 on the UMD dataset, and 0.789 on the IIT-AFF dataset.

In 2021, Gu et al. [61] reported state-of-the-art results on the UMD dataset, a F_β^w score of 0.941, by utilizing an encoder-decoder architecture with a DRN network used as the feature extractor. Furthermore, they apply a self-attention module to improve upon the salient details of the affordance map.

In 2022, Yin et al. [64] applied recent advances in image segmentation methods with their network called SEANet, which they modified to include a spatial gradient fusion module and a shared gradient attention module. They achieved a F_β^w score of 0.794 on the IIT-AFF dataset.

Inference time improvements

While some researchers focus on improving the accuracy of the affordance segmentation, others aim to reduce the inference time by finding the balance between the accuracy and the size of the model.

The aforementioned model proposed by Gu et al. [61] not only reportedly achieves state-of-the-art results, but it is also considerably faster than AffordanceNet. Due to the self-attention module, the network proposed by Gu et al. has only 16 million parameters compared to the 41 million parameters of AffordanceNet [62].

Another way to cut the inference time is to replace the encoder in encoder-decoder methods with smaller and more efficient convolution neural networks such as the MobileNet architectures or ESPNetv2. Tsai et al. [62] proposed a one-stage method based on the ESPNetv2 with only 4.6 million parameters, achieving a F_β^w score of 0.610 on the IIT-AFF dataset. Ragusa et al. [63] make use of MobileNetv1 and MobileNetv3, which they combined with the segmentation architectures UNet, SegNet, and LR-ASPP. Ragusa et al. managed to achieve results comparable to AffordanceNet with only 7.6 million parameters, achieving a F_β^w score of 0.750 on the UMD dataset. Moreover, they also proposed a lightweight model with only 0.6 million parameters that can run in real-time on a Jetson Nano. However, this lightweight model managed to achieve a F_β^w score of only 0.586 on the UMD dataset.

An overview of all the discussed methods, both regarding performance and inference time improvements, can be seen in Table 2.2.

Method	Year	Millions of parameters	F_β^w UMD	F_β^w IIT-AFF
Myers et al. S-HMP [19]	2015	N/A	0.557	
Myers et al. SRF [19]	2015	N/A	0.466	
Nguyen et al. [20]	2016	N/A	0.770	
Nguyen et al. BB-CNN-CRR [21]	2017	N/A		0.696
Do et al. AffordanceNet [22]	2018	41	0.799	0.734
Zhao et al. [57]	2019	N/A	0.861	0.789
Minh et al. [58]	2020	N/A		**0.802**
Qian et al. [59]	2020	N/A	0.814	
Abdhalwhab et al. [56]	2020	N/A	0.707	
Xu et al. AffKp [60]	2021	N/A	0.820	
Gu et al. [61]	2021	16	**0.941**	
Tsai et al. [62]	2021	4.6		0.610
Ragusa et al. V1_se [63]	2021	7.6	0.750	
Ragusa et al. V1_U [63]	2021	10.4	0.736	
Ragusa et al. V3_LR [63]	2021	0.6	0.586	
Yin et al. [64]	2022	N/A		0.794

Table 2.2: Summary of the investigated methods. Gu et al. [61] and Minh et al. [58] achieve state-of-the-art results.

2.2.3 Related work concerning the issue of limited data and scalability

The majority of the improvements in the field of affordance segmentation concentrate on improving the architecture. Very few works focus on improving the datasets which are limited in size, scope, and variation compared to the datasets for other computer vision tasks e.g., object detection, segmentation, and classification [65, 66, 67]. As a result, the affordance segmentation methods trained on the UMD or IIT-AFF datasets do not generalize very well. The remainder of this subsection will describe methods that have been proposed to alleviate the issue of the limited data.

Weakly supervised learning methods

Supervised learning methods require all pixels belonging to a certain affordance to be labelled. However, this is not true for the weakly supervised learning methods, which only require sparse annotations. As sparse annotations are less resource-heavy than pixel-wise annotations, weakly supervised learning methods might be an approach to alleviate the issue of the limited data.

Sawatzky et al. [51] proposed VGG and ResNet based networks that can train on data annotated with only sparse keypoints. They showed it is possible to train on weakly annotated data, but the results are sub-par. In order to improve the performance, they deploy the Grabcut algorithm to refine the train set predictions of the neural network

trained on the keypoint annotations only. Afterwards, the training continues but using the refined annotations, see Figure 2.7.

Sawatzky et al. later expanded upon this method in [68]. In this new work, they removed the Grabcut segmentation step from the pipeline by introducing an approach for adaptive binarization of the prediction during the training process.

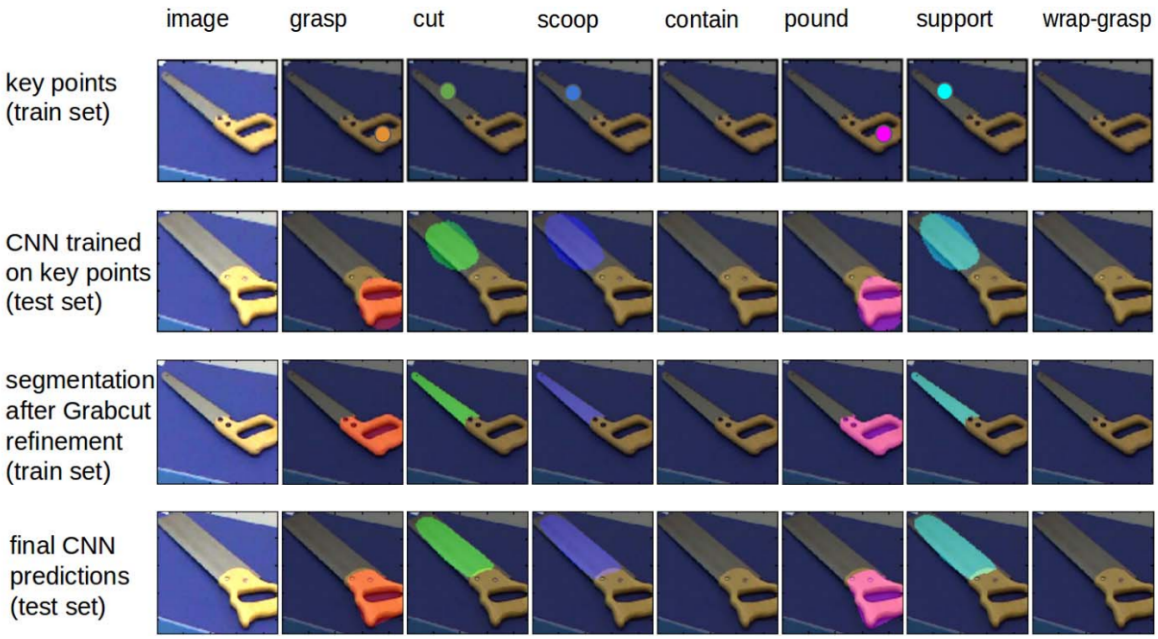


Figure 2.7: Visualisation of the method proposed by [51]. The first row shows the keypoint annotations. The second row shows the predictions of the neural network trained on only the keypoint annotations. The third row shows the refined predictions of the neural network trained on keypoint annotations only. Finally, the fourth row shows the final predictions of the neural network after training on a train set annotated with the refined annotations. Taken from [51].

Gall et al. [69] leveraged already existing large datasets with object annotations, such as object detection datasets, in order to segment affordances. This was achieved by using a semantic alignment network to transfer affordance annotations, in the form of an affordance bounding box, from a small set to a larger set of images with annotations only at the object level i.e., the object class and the bounding box are known but there is no affordance information. After the complete dataset is assembled, consisting of the small manually annotated dataset combined with the large dataset automatically compiled through the semantic alignment network, a VGG-based neural network was trained for pixel-wise affordance segmentation in a weakly supervised manner. See Figure 2.8 for the visualisation of the labels transfer process.

Nevertheless, all three aforementioned weakly supervised methods achieved poor performance compared to the supervised methods.

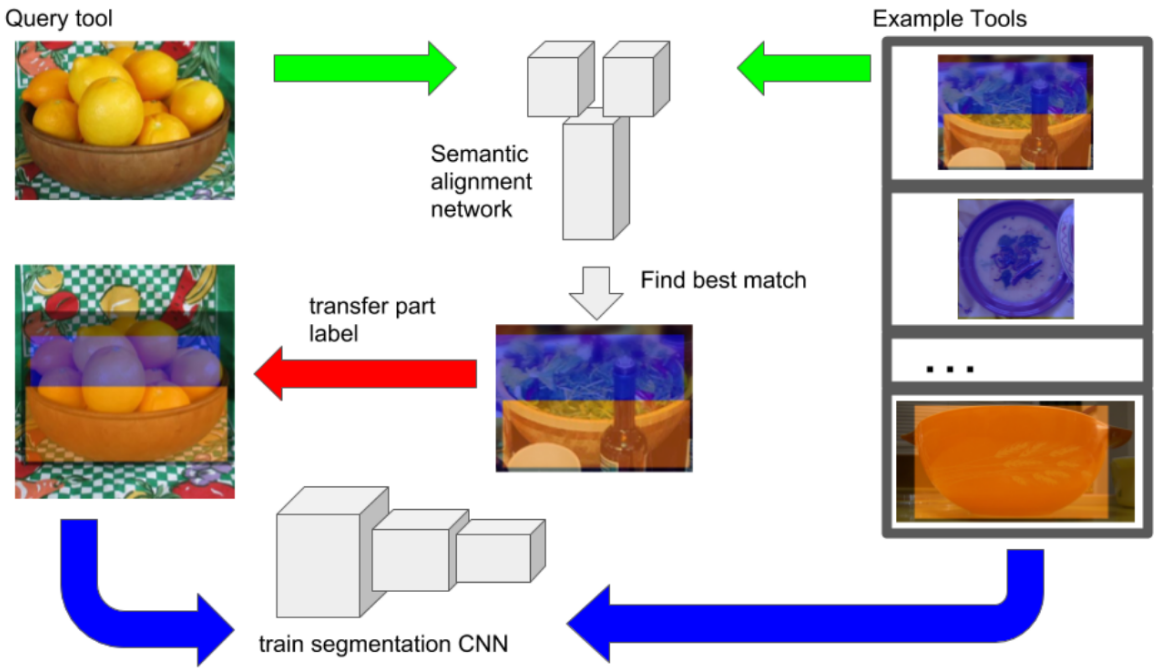


Figure 2.8: The visualisation of the label transfer process proposed by [69]. The training data on the right is a small set of images annotated with bounding boxes for available affordances. The data on the left is collected from an object detection dataset. As such, only the bounding box and the object class are available. In order to transform labels from the training set to new images, the semantic alignment network was used to find the best match. Taken from [69].

Category-agnostic affordance segmentation

Most of the previously described methods are object-dependent, meaning that the methods predict both the object class and the affordance segmentation. For the object-dependent methods, optimizing class detection and affordance prediction simultaneously improves the performance of the models. But this limits the object-dependent based methods to the class labels in the training dataset. In order to improve generalizability, category-agnostic methods have been proposed

Category-agnostic affordance segmentation is done by separating the affordance segmentation from object categories. Apicella et al. [70] and the aforementioned Chu et al. [23], have achieved category-agnostic affordance segmentation by discriminating only between the background and the foreground. Chu et al. [71] later improved their previous solution by introducing the attribute module, which allowed the model to learn the attributes of an object class and the region-based self-attention module. Using this improved model, they achieved state-of-the-art performance on the UMD dataset with a F_β^w of 0.690 on novel object categories. Object-based category-agnostic methods, therefore,

like the weakly-supervised methods, suffer from a performance drop compared to the object-dependent methods.

Finally, it could be argued that the previously mentioned encoder-decoder networks [57, 60, 61, 64] are also category-agnostic as none of them performs object detection. Instead, they learn affordances for all objects present in an image. However, these methods do not perform object detection, which might be detrimental to a robotic handover system.

Affordance segmentation using synthetic data

The limited amount and scope of affordance datasets could be addressed using synthetic data. Annotating real-world data with affordance labels for each pixel is a resource-heavy process that does not scale well. On the other hand, synthetic datasets are easy to generate, provide error-free ground truth labels without the need to annotate manually, and are inexhaustible [26].

The issue with synthetic data is that methods trained on such data might perform poorly in real-world scenarios [27]. This is commonly referred to as the sim2real gap, which refers to a challenge of transferring the capabilities of the learned model from the simulated environment to the real-world. The sim2real gap exists because models of the simulated environment are not sufficiently accurate to represent real-world models [72].

Nevertheless, the sim2real gap can be bridged, or at least shortened, using either domain adaptation, domain randomization or both methods [28]. Domain adaptation is used for fitting models trained in a source domain to a previously unseen target domain. Domain randomization speculates that with enough variety in the synthetic data, the real-world data could appear as another variation to the model. This can be achieved by randomizing parameters of the synthetic data, such as scene lighting, lighting colour, object poses, or background textures.

The remainder of this subsection will investigate the methods proposed for affordance segmentation using synthetic datasets.

In 2017, Detry et al. [12] trained a segmentation network called MultiNet, which used the VGG16 as the backbone and the FCN encoder-decoder architecture. They synthesized a small dataset of manually annotated 3D meshes, which were later used to generate synthetic scenes. The dataset contained ten meshes, five from the YCB dataset, where each mesh was binary annotated for tasks transport, handover, pour, and open, see Figure 2.9. Nevertheless, this method was not evaluated on either the UMD or IIT-AFF dataset.

(a) (b)

Figure 2.9: (a) Annotated meshes with binary labels for each task. Green represents an area suitable for a task, red represents unsuitable for a task. For example, the item on the left shows the handover task constraints for a brush. The robot can grasp the item anywhere in the green area, leaving the handle, red area, unobstructed for the user. Similar constraints were defined for all combinations of the four tasks and the dataset objects. (b) An example of the generated synthetic scene. From the left - RGB, depth, and annotations for the transport task [12].

In 2019, Hämäläinen et al. [30] made use of the synthetic data for training manipulation policies based on a latent representation of affordances. Affordance information was extracted using the custom encoder-decoder structure named VAED. The work proposed three neural networks, all trained on synthetic data, including the VAED network. The object of interest for this work was a mug with the affordances contain and wrap-grasp.

In order to train the three neural networks, Hämäläinen et al. synthesized a dataset with the following domain randomization techniques:

- randomized textures,
- randomized mug geometry,
- randomized distractor objects,
- randomized camera viewpoints.

In total, 1 million pairs of RGB images and affordance masks with the affordances contain and wrap-grasp were generated, see Figure 2.10. However, Hämäläinen et al. do not report any evaluation metric on either the IIT-AFF or UMD dataset after training on synthetic data. Therefore, it is impossible to measure quantifiably measure how well the authors managed to overcome the sim2real gap.

The most comprehensive work done on affordance segmentation using synthetic data was done by Chu et al. [29]. They collected 93 object meshes from 3DWarehouse [1] covering 17 object categories and seven affordance categories present in the UMD dataset, which were used to create a dataset of 37k images, see Figure 2.11, compared to 28k images in the UMD dataset. A network similar to AffordanceNet was then trained on this synthetic data in a supervised manner. In order to tune the weights from the synthetic domain to the real world domain, Chu et al. implemented three domain adaptation modules into

[1] https://3dwarehouse.sketchup.com/?hl=en

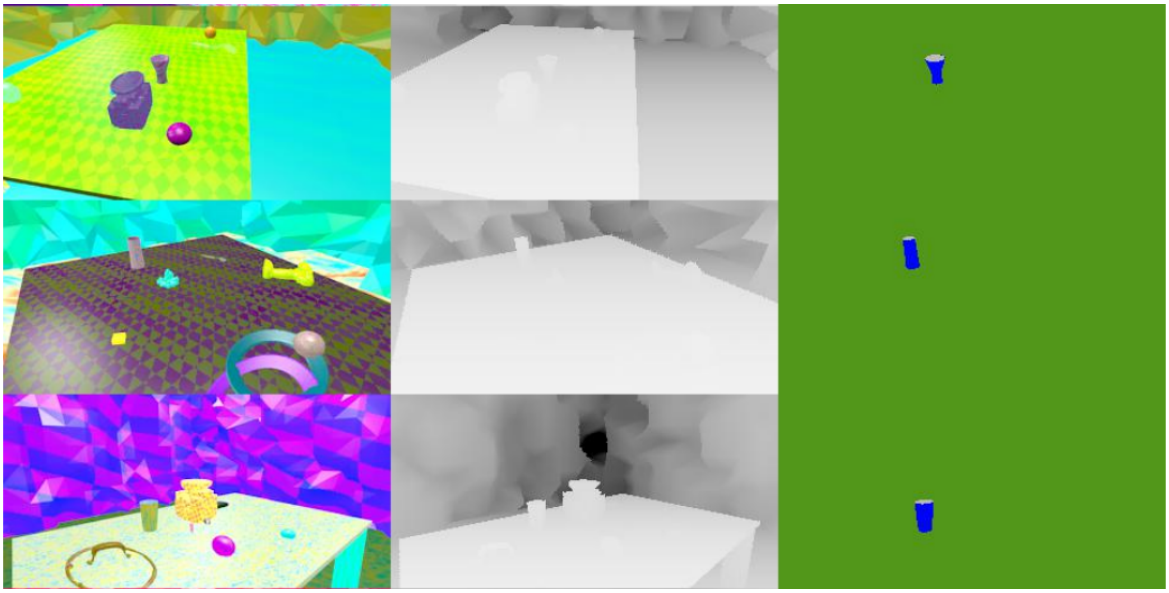


Figure 2.10: Sample from the synthetic dataset generated that the VAED network was trained on. From the left - RGB, depth, and ground truth affordance mask [30].

their network. One after the feature extractor, and the other two in the two task branches. The domain adaptation modules were trained in an unsupervised manner.

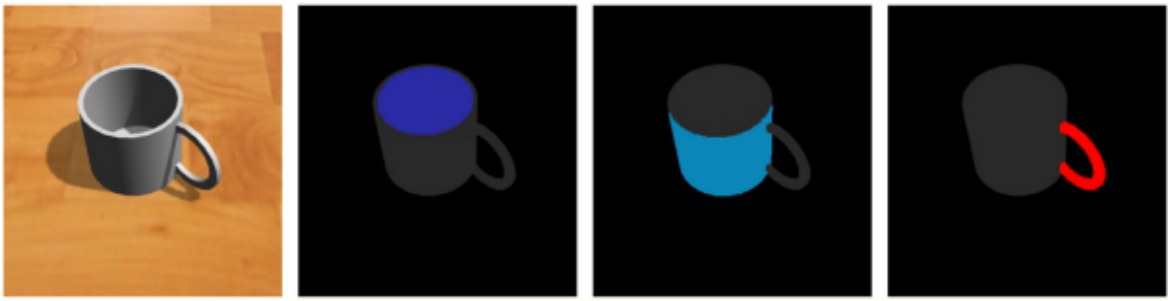


Figure 2.11: Sample from the synthetic UMD dataset with the ground truth affordances - contain (blue), wrap-grasp (light blue), and grasp (red) [29].

The method by Chu et al. trained on their synthetic UMD data achieved a F_β^w score of 0.546 on the real UMD dataset compared to a F_β^w score of 0.799 achieved when they trained their network on the real UMD dataset with no synthetic data. In order to achieve a better performance, Chu et al. experimented with fine-tuning of the network trained on the synthetic data on a subset of the real UMD dataset after training on the synthetic data. By sampling 25 % of the real UMD data, they attained a F_β^w score of 0.714. With 50 % real data, they get managed to achieve a F_β^w score of 0.730.

2.2.4 Identified areas for improvement

In conclusion, all three of the aforementioned approaches i.e., weakly supervised methods, category-agnostic affordance segmentation, and synthetic data, aim to address the issue of limited data and scalability in one way or another. On the one hand, weakly supervised methods do not require large datasets with pixel-wise annotations, but they do not perform very well compared to the supervised methods. On the other hand, category-agnostic methods achieve high performance, but they are still limited by the datasets they are trained on.

Affordance segmentation using synthetic data appears to be a promising way to address the data scalability limitations. As was mentioned before, synthetic datasets are easy to generate, have error-free annotation, and are inexhaustible. Yet, affordance segmentation from synthetic data has only been briefly explored so far. Although Chu et al. [29] achieved a comparable F_β^w score to that of the AffordanceNet, this was only achieved after applying the domain adaption technique with 50 % samples from the real UMD dataset. Without it, a 30 % performance drop was recorded. Hämäläinen et al. [30] also trained their network using synthetic data. In order to overcome the sim2real gap, they used the domain randomization technique instead. However, because the produced synthetic dataset only contained a single class, the capabilities of their proposed VAED network were also quite limited.

2.3 The problem of object orientation at handovers

As mentioned in section 2, proper orientation of an object at the handover time has shown to improve the perceived quality of the handover. This section, therefore, investigates previous methods for computing object handover orientations as well as how to evaluate said methods.

2.3.1 Related work in object handover orientation

This section covers related work that focuses on computing a suitable or proper object handover orientation. Early work focused simply on performing handovers with orientations specific to an object. The authors of these methods used predefined handover orientations for each object. As such, these methods do not not scale well [13, 14, 15].

Cakmak et al. [13] proposed two methods for object handover orientation. The first method computes the handover location such that the human receiver does not have to move. The position and orientation of the object at the handover was based on a kinematic model of a 162 cm tall human. The object orientation is computed by maximizing the possible number of grasps available on the object presented to the user, using a kinematic model of a human hand. The second method learns handover orientations from human guidance. A user study was conducted where each participant had to select four good and

four bad handover orientations in a simulation for five different objects, examples can be seen in Fig. 2.12. They found that people prefer the learned method.

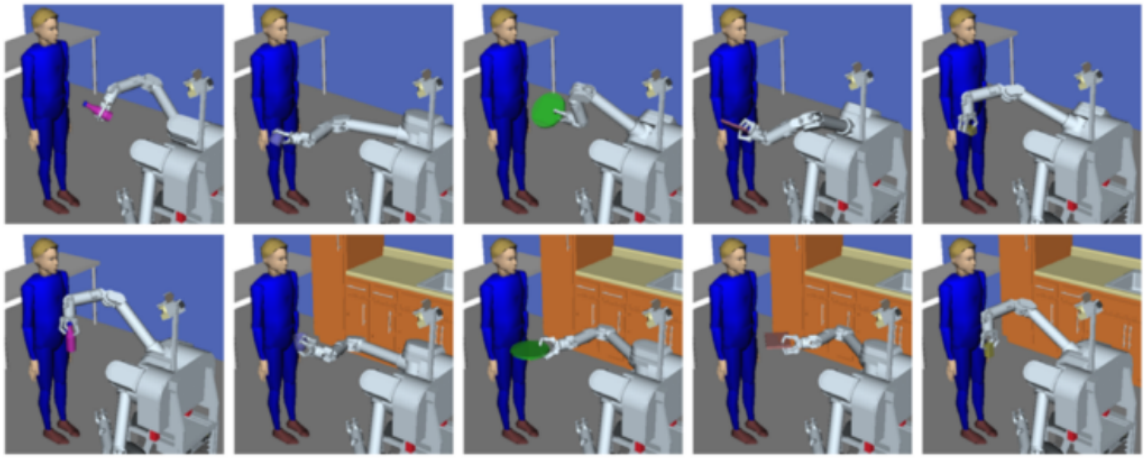


Figure 2.12: Examples generated with the methods from [13]. Top is the first method based on a kinematic model, bottom row is generated using their learned method.

The initial work by Aleotti et al. [14], later slightly extended in [15], presents a system that can handover objects with a six DOF robot combined with a Kinect depth sensor. An object, known in advance, is scanned and segmented into its part using the REEB graph method, see Figure 2.13. The position of the receiver is detected using the Kinect depth sensor, and it is used to compute the handover location and orientation. One of the segmented parts, chosen in advance for each object, is oriented towards the receiver.

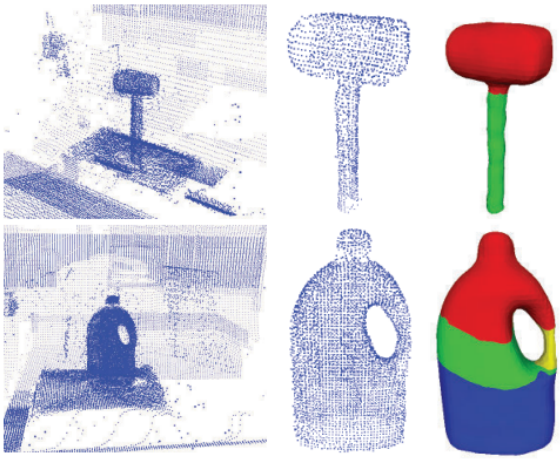


Figure 2.13: The segmented parts of an object using the method presented in [14]. One of the segmented parts is chosen in advanced to be oriented towards the user.

The methods presented so far all rely on knowing the object in advance and assigning a handover orientation manually on a per object basis. Orientations computed with these methods are often also not appropriate or natural because they do not consider the object's affordances. Cakmak et al. further found that people prefer handover orientations learned from human examples rather than a loss function proposed by the authors [13].

Chan et al. [16] addressed this issue when they proposed to learn handover orientations from natural handovers as performed by humans on 10 different objects. A knowledge base is created by observing objects being used. From this, they can extract manually defined features describing the object affordances. Features regarding the affordances *cut*, *screw*, *translate* and *slide* are extracted. At the same time, a database of proper object orientations with respect to the receiver is also created. Objects along with their associated handover orientations are then clustered based on the extracted features. Object orientations can then be computed by classifying a new unseen objects using k-nearest neighbour algorithm using the extracted features. In their follow up work [10], a different method for computation of object handover orientation was developed which does not rely on features nor a knowledge database. Instead, handover orientations are learned directly from observing human handovers on a per object basis. For each object, they compute what they name the 'affordance axis' at the moment of handover. They argue the affordance axis orientation depends on the affordances of the object. Their method can also differentiate between a good and a bad handover orientation, which is something they slightly expanded upon in their follow up work [11]. Finally, Razalli and Demiris [73] applied deep learning to the problem and developed a multitask variational autoencoder that could predict human and object pose from the dataset created in [10].

As an addition, it is also worth mentioning that some works address the handover orientation problem implicitly. Ardon et al. [7] developed a method for choosing the proper handover location as well as configuration of the robot for people with limited arm mobility. The method considered *appropriateness*, *safety*, and *reachability*. Task-oriented grasps are sampled based on affordances, and the sampled grasp location that best leaves the *grasp* affordance free for the receiver is chosen. In order to maximize safety, they maximize the distance between the robot and the human, thereby, unintentionally addressing some of the handover orientation problem. Such methods do not generalize to all types of objects.

Improving upon current methods

Research into object handover orientation remains rather limited, with most of the related work using orientations tied to specific object categories, such as a hammer, a spoon, or a teapot. These methods do not generalize to novel object categories. Learning object handover orientation from human observations in combination with affordances in such a way that the object categories can be completely disregarded remains an open area of

research. By separating learned handover orientations from their object categories, a more generalizable system might be achieved.

The majority of the related work has focused on learning handover orientations from human handover examples. However, obtaining datasets of human handovers can be expensive and time consuming. Works such as Ray et al. [43] and Cini et al .[44], show that people orient objects with the grasp affordance i.e., a handle, towards the receiver. A set of pre-defined rules might, therefore, be sufficient for generating object handover orientations, as long as the object affordances are considered.

2.3.2 Methods to evaluate the object handover orientation

Evaluation methods and metrics are used to quantitatively express the performance of a method and allows for comparison to other methods. The methods covered in this section does not have a unified framework for evaluation, but the evaluation methods can be split into two broad categories - quantitative evaluation on a dataset [16, 10, 11, 73] and human-robot-interaction (HRI) experiments [13, 14, 15].

Evaluating with human-robot-interaction experiments

Methods that evaluate using HRI experiments often conduct small scale user studies of 10 to 16 participants. The experiments are set up to compare two methods against each other, and therefore, only report relative performance of the two methods.

Cakmak et al. [13] test their two proposed methods against each other by performing HRI handover experiments with 10 participants. The methods are compared on the metrics *liking*, *naturalness*, *practicality* and *appropriateness*, where the participants are asked to consider the position and orientation of the object during the handover. These criteria do not capture handover orientation directly, but relates to the performance of the whole robotic system.

Aleotti et al. [14, 15] conducted a user study with 16 participants that had to perform handovers, see Fig. 2.14. Their proposed method was compared against a perturbed version of their own method. In the perturbed version, the object handover position and orientation were randomly pertubed. The methods were compared on the criteria of *comfort*. Their proposed method achieved higher levels of *comfort* compared to the perturbed version.

Quantitative evaluation on a dataset

Where qualitative evaluation, such as the aforementioned HRI experiments, suffers from reproducibility issues, which complicates the comparison of methods, quantitatively evaluating methods on a dataset do not. Chan et al. [16] produced a training set of 10 different objects with 20 handover orientations and an evaluation dataset containing three novel

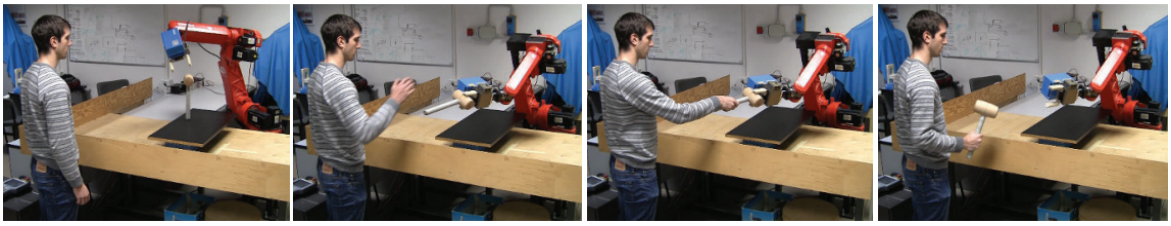


Figure 2.14: An example of the handover experiment carried out by Aleotti et al. [14] while evaluating the proposed system.

objects not present in the training dataset. They treated the evaluation as a classification problem.

In another study, Chan et al. [10] captured a dataset of handovers demonstrations with 20 objects and 20 participants. The participants were asked to handover the objects under three different conditions - A, B, and C. Condition A was natural handover with no instructions. Under the condition B, the giver had to prioritise their own comfort. In condition C, the giver had to prioritise the receiver's comfort. Based on this data, they showed that their method could differentiate between condition B and condition C.

Razali and Demiris [73] is the only study that reported orientation errors quantitatively on a continuous scale. This was done on the dataset described above produced by Chan et al. [10]. Errors were reported as the L1 error in radians around each axis of a 3D coordinate system, see Table 2.3.

X-axis	Y-Axis	Z-axis
0.961	0.452	1.749

Table 2.3: Handover object orientation error reported in radians by [73].

Quantitatively measuring the orientation errors, as done by Razali and Demiris [73], suffers from the issue that the orientation of some objects are equivariant or multipolar [11]. This means that the orientation around one or several axis is irrelevant or equally good. As an example, the rotation around a bottle's vertical axis can vary drastically in a handover as said rotation does not affect the perceived quality of a handover.

Evaluation by user studies, therefore, seems as the most promising method for evaluating the performance of object handover orientation methods.

2.4 Problem formulation

Before the final problem formulation is proposed, a summary of the problem analysis and its findings is provided.

Ortenzi et al. [8] argues that the handover action is usually initiated to allow the receiver to carry out a task with the transferred object. Thus, one of the giver's key con-

siderations during the handover should be the subsequent task of the receiver. As such, task-oriented handovers are preferred over their task-agnostic counterparts.

In order to perform a handover, a trajectory must be planned. The process of trajectory planning involves, among other considerations, grasp synthesis and computation of a handover location and object orientation. A study by Ortenzi et al. [34] showed that what is considered a good grasp depends on the task that is to be performed with the grasped item. Given the focus of this report is robot-to-human handovers, a suitable grasp is a grasp that allows for a proper handover. However, an appropriate handover grasp depends not only on the mechanical stability of the grasp but also on the subsequent task of the user, as shown in [8]. Thus, task-oriented handovers are tightly coupled with task-oriented grasping, which was addressed in our previous work [32].

Our solution to task-oriented grasping is built upon the concept of affordances. As was shown in section 2.2, currently available affordance segmentation datasets are limited in size, scope, and variation, but creating a new dataset is a resource-heavy process. This in turn affects the affordance segmentation methods trained on said datasets, as smaller datasets lack variety. This was observed in our previous work [32], in which change in illumination and object pose severely affected the performance of the affordance segmentation method.

In order to overcome the issue of the limited data, three strategies have been explored in past works, namely weakly supervised methods, category-agnostic methods, and training on synthetic datasets. The weakly supervised methods perform poorly, while the category-agnostic methods perform better than weakly supervised methods, but still suffer from a performance drop compared to the object-dependent methods. Synthetic data could be used to alleviate the issue of limited data. However, as was shown in section 2.2, the use of synthetic data has been investigated only briefly, and with nearly no research into a synthetic dataset using the domain randomization method. Therefore, this report implements affordance segmentation trained on a synthetic dataset generated using the domain randomization method which, to the best of the authors' knowledge, has not been investigated yet.

As argued in section 2, the object's orientation during the handover affects both the efficiency and the perceived safety of the handover [9, 42]. Therefore, proper object handover orientation can improve collaboration during the handover.

As presented in section 2.3, related work in the field of the object handover orientation is rather limited and consists mostly of object-based methods. Object-based methods do not generalize to novel object categories. Only considering the object affordances and disregarding the object categories could alleviate the issue of generalizability.

A rule-based method using affordances was proposed by Ardon et al. [7]. Nevertheless, orientations calculated via the object-based or rule-based methods might not be as suitable as orientations learned from human-human handover observations [13].

However, as Cakmak et al. [13] argue, the reason why object handover orientations should be learned from human-to-human observations is that such handover orientations

would implicitly encode the affordances of the objects. Therefore, two methods will be implemented - an observation-based method, where the handover orientations are learned from human-to-human observations, and an affordance-aware rule-based method. Later, the observation-based and rule-based method will be compared against each other in order to see if it is indeed necessary to learn from human-to-human handover observations or if the affordances-aware rule-based method can perform similarly well.

Reflecting on the aforementioned findings, the problem formulation for this report is as follows:

How can robot-to-human handovers be performed using object affordance segmentation in a collaborative robot setup?

To consider affordance segmentation and handover orientation, the problem formulation stated above is expanded with the following sub-problem formulations:

- *How can affordance segmentation methods trained on synthetic data enhance the performance of robot-to-human handovers?*
- *How can object affordance theory improve upon handover orientation in robot-to-human handovers?*

2.4.1 Custom object set

The object set used in this report consists of items that can be found in the UMD dataset plus a bottle from the ITT-AFF. Therefore, the object set consists of a bottle, a bowl, a cup, a hammer, a knife, a ladle, a mallet, a mug, scissors, three scoops, a spatula, and two spoons. The object set can be seen in Figure 2.15.This object set is used as the object set for a human-to-human handover dataset and real-world robotic manipulation.

Figure 2.15: The object set used in this project.

2.5 Requirements

In order to evaluate the quality of the proposed solution, requirements to pass must be met. The selection of these requirements is discussed in this section.

2.5.1 Task-oriented handover

While the main focus of this report is the computation of the handover object orientation and affordance segmentation using synthetic data, both of these are investigated in an effort to improve the handover experience. Therefore, the success rate for the task of the task-oriented handover must be set. In our previous work [32], the success rate of the task-oriented handover was set to be 65 %. During the testing, we managed to achieve a success rate of 72.22 %. As this report aims to advance our previous work, the desired success rate of task-oriented handover should be higher, thus, we set the requirement to 75 %.

Requirement

- Success rate of handover of 75 %

Finally, because this report aims to provide a pipeline for task-oriented handover with proper object handover orientations, it would be natural to set up a success rate requirement of how often the desired object orientations are reached. However, such a requirement would be arbitrary as, to the best of the authors' knowledge, no other researchers have provided a number to compare against. Nevertheless, the success rate of how often the desired object handover orientation is reached will be provided along with the full system test results.

2.5.2 Affordance segmentation

As mentioned in section 2.2, Hämäläinen et al. [30] synthesized a dataset using the domain randomization method, and as such, at first glance it appears as the most similar work to the work of this report. However, Hämäläinen et al. did not report any evaluation metric on the IIT-AFF or UMD dataset. Furthermore, it was limited to a single object category.

Nevertheless, Chu et al. [29] also synthesized a dataset, but applied the domain adaptation method instead. While domain adaptation differs from domain randomization, both methods are used to overcome the sim2real gap. Furthermore, Chu et al. also reported a F_β^w score of 0.546 their method achieved on the real world UMD dataset after exclusively training on the synthetic data. Therefore, the aforementioned F_β^w score reported by Chu et al. [29] is set as the benchmark for our affordance segmentation method.

Requirement

- $F_\beta^w > 0.546$

2.5.3 Handover orientation

Unlike affordance segmentation, no widely agreed-upon evaluation method, qualitative or quantitative, exist for handover orientations. Furthermore, quantitative methods suffer from the problem of orientation equivariance and multi-polarity. Due to these problems, our handover orientation solution will be evaluated using a human-robot user study.

The purpose of this study will be to identify a method of handover orientation that produces object handover orientations that the users find the most comfortable, legible, and appropriate. As such, these three metrics will be the focus of the user study. Three investigated methods will be:

- random-based method, when the handover orientation is a pre-defined pose with randomly sampled orientation,
- rule-based method, when the handover orientation is selected based on the pre-defined rules,
- observation-based method, when the handover orientation is selected based on the human-to-human handover observations.

Hypotheses of the user study

- We expect the user to perceive:
  - Higher comfort, legibility, and appropriateness of the object handover orientations for the rule-based and observation-based methods compared to the random-based method.
  - Comparable comfort, legibility, and appropriateness of the object handover orientations between the rule-based and observation-based methods.

3 - Implementation

A system capable of performing a robot-to-human handover, that at the same time aims to fulfill the requirements from Section 2.5, was designed and implemented. To improve upon the system designed previously [32], the system must have the following capabilities.

- To enable the task-oriented handover, a method for affordance segmentation must be implemented. The affordance segmentation method can utilize synthetic data to alleviate the data limitations and scalability issues of the past affordance datasets.
- The system must be capable of task-oriented grasping and designing a trajectory that considers the handover orientation of different objects. Since object affordances has shown to affect the object handover orientation [11], the method for computing handover object orientations should compute orientations based on object affordances.
- The handover object orientation is defined in relation to the receiver, therefore, the system must also consider the handover location as well as the location of the receiver.

This section covers the design and implementation of the task-oriented handover system proposed in this report.

3.1 System overview

The hardware setup used for this project is depicted in Figure 3.1. The setup consists of a KUKA LBR iiwa 7 R800 collaborative robot with 7-DOF, an Intel RealSense D435i RGB-D sensor, a Hokuyo URG-04LX-UG01 2D laser scanner, and a Robotiq 3-finger gripper. The system is mounted on a custom modular table provided by Technicon. Interfacing with KUKA was achieved using the *iiwa_stack* ROS package [74]

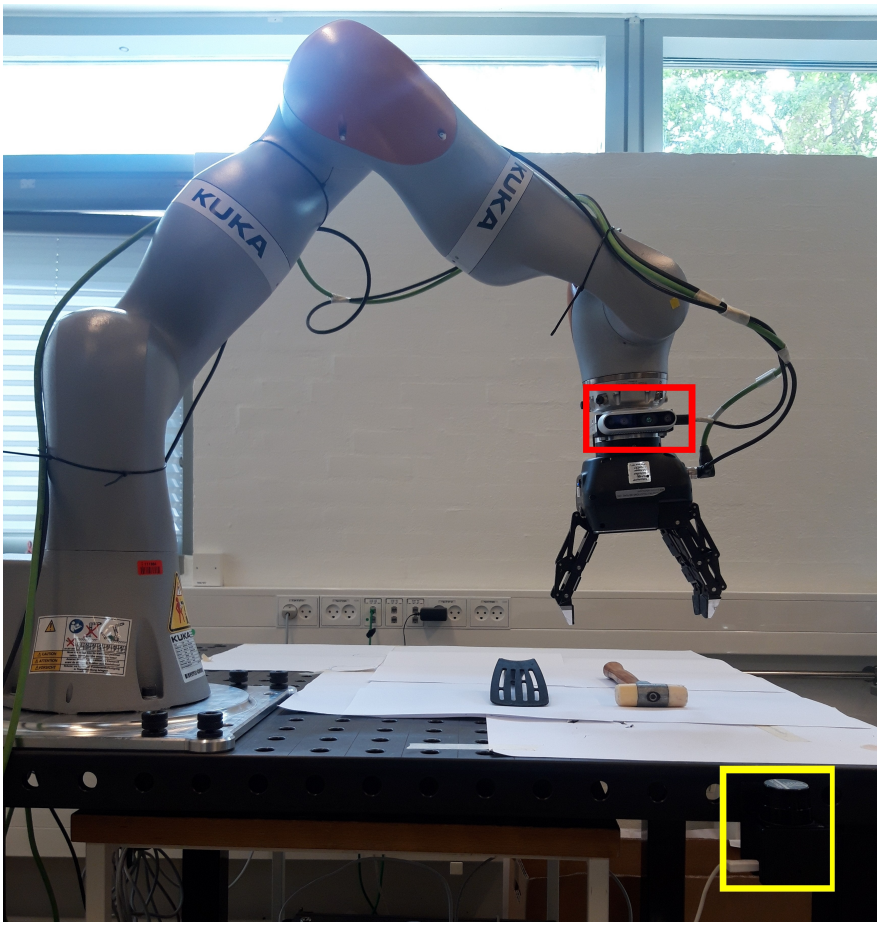


Figure 3.1: The hardware setup used in this project. An Intel RealSense camera is highlighted by the red box. A Hokuyo laser scanner is highlighted by the yellow box.

A system was designed to perform task-oriented handovers. The system should be capable of performing task-oriented grasping, where the object is grasped in a way that leaves the handle free for the receiver to grasp. A handover location must be chosen such that the handover happens at the midpoint between the robotic giver and the human receiver. Finally, the handover object must be oriented in such a manner that it considers the comfort of the receiver as well as the object affordances. Such a system is visualized in Figure 3.2

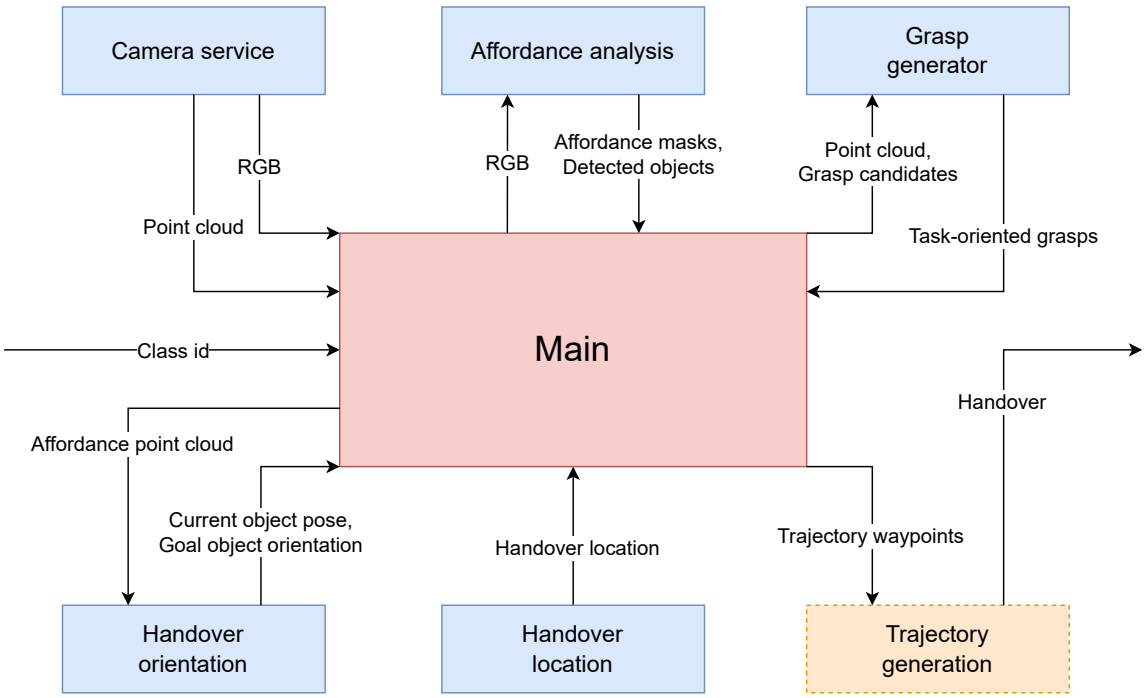


Figure 3.2: A block diagram of the implemented system. Blue boxes indicates that the module has been implemented as a ROS service node. Orange modules are tasks carried out by the KUKA sunrise controller. Best viewed in colors.

The system was implemented in the ROS Melodic framework, which facilitates standardized interprocess communication as well as providing standardized methods for communicating with sensors. The system was designed with an object-oriented design philosophy in mind. Each module solves a single task related to the task-oriented handover or solves auxiliary tasks such as generating trajectories or providing sensory information. The *main* module collects and combines all of the output of the sub-modules in order to compute a task-oriented handover that also considers handover position and orientation. The only module not made by the authors is the *trajectory generation* module. The trajectory generation is done via KUKA's proprietary point-to-point trajectory algorithm.

The section starts by covering the *main* module in order to describe the overall system. Afterwards, the various submodules are described. The *camera service*, *handover position*, and *grasp generator* modules are covered briefly, while the affordance analysis and *handover orientation* modules are explained more in depth, as they are the main focus of this project.

3.1.1 The main module

Given an object to handover, the *main* module, depicted in Figure 3.3, computes a trajectory that enables a task-oriented handover. To do this, it utilizes the outputs of all the other modules. The *main* module acts as the binding link between the other modules, and it is

in charge of receiving and sending information between the modules. Using the sensory information received from the *camera service* module and the Hokuyo URG-04LX-UG01 2D laser scanner, a set of waypoints consisting of the pre-grasp pose, the grasp-pose and the end-effector handover pose. The waypoints are computed using information from the *grasp generator*, *handover orientation* and *handover position* modules. The grasp-pose itself is computed by the *grasp generator* module. The set of trajectory waypoints computed by the *main* module consists of the pre-grasp pose and the end-effector handover pose. The robot avoids unintended collisions with the objects, by moving to the pre-grasp before performing the grasp. The pre-grasp pose shares the orientation of the grasp, but it is placed -10 cm along the z-axis of the grasp. The end-effector handover pose describes an end-effector's pose that positions and orients the handover object as desired.

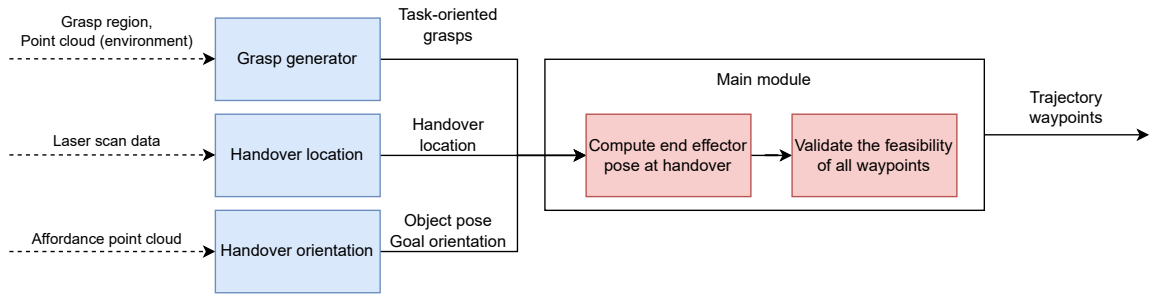


Figure 3.3: An overview view of the *main* module. The blue boxes represents external modules, while red boxes are computations done within the *main* module.

In order to compute the end-effector handover pose, the *main* module requires the task-oriented grasp, the current pose of an object, and the desired handover location and orientations as the inputs. To describe the computation of the end-effector handover pose mathematically, let us first declare the four transformations of interest:

- transformation from the world frame to the end-effector frame ${}^{World}_{Ee}\mathbf{T}$,
  - This transformation can at any point be requested from the ROS *tf* tree.
- transformation from the world frame to the current pose of the object ${}^{World}_{O}\mathbf{T}$,
  - This transformation describes the estimated pose of the object as computed by the *handover orientation* module.
- transformation from the world frame to the object's desired pose ${}^{World}_{Goal}\mathbf{T}$,
  - This transformation describes the goal handover location, as computed by the *handover location* module, and the goal handover orientation, as computed by the *handover orientation* module.
- transformation from the world frame to the end-effector frame in which the grasped object achieves its desired object pose ${}^{World}_{Ee\_goal}\mathbf{T}$.

– This transformation is the one that needs to be computed. This transformation is relevant because the robot's planning and trajectory execution is defined as the robot's end-effector pose in relation to the world frame.

In order to visualize what these transformations represent and where they are placed on the robotic system, the aforementioned transformations are visualised in Figure 3.4.

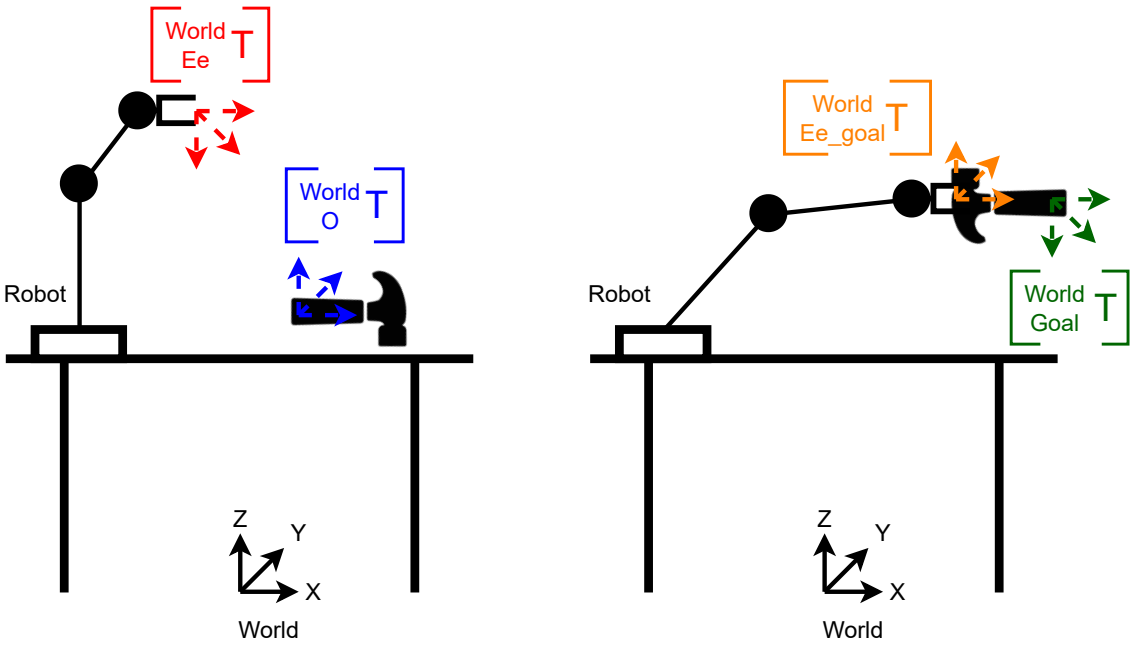


Figure 3.4: Visualisation of the transformations from which it is possible to calculate a goal end-effector pose that achieves the goal object handover location and orientation.

In order to compute ${}^{World}_{Ee\_goal}\mathbf{T}$, following equation must be solved:

$$ {}^{World}_{Ee\_goal}\mathbf{T} = {}^{World}_{Goal}\mathbf{T}\ {}^{O}_{Ee}\mathbf{T} \tag{3.1} $$

where ${}^{O}_{Ee}\mathbf{T}$ is the transformation from the object's pose to the end-effector. After the object is grasped, this transformation remains constant as the object no longer moves in relation to the end-effector.

Transformation ${}^{O}_{Ee}\mathbf{T}$ can be computed using the grasp pose found by the *grasp generator* module, denoted as transformation ${}^{World}_{G}\mathbf{T}$, and transformation ${}^{World}_{O}\mathbf{T}$. Moreover, at the moment of grasp, ${}^{World}_{G}\mathbf{T}$ equals ${}^{World}_{Ee}\mathbf{T}$. Therefore, ${}^{O}_{Ee}\mathbf{T}$ can be computed as follows:

$$ \begin{gathered} {}^{Ee}_{O}\mathbf{T} = {}^{World}_{G}\mathbf{T}^{-1}\ {}^{World}_{O}\mathbf{T} = {}^{World}_{Ee}\mathbf{T}^{-1}\ {}^{World}_{O}\mathbf{T} \\ {}^{O}_{Ee}\mathbf{T} = {}^{Ee}_{O}\mathbf{T}^{-1} \end{gathered} \tag{3.2} $$

To reiterate, after grasping the object, transformation ${}^{O}_{Ee}\mathbf{T}$ remains constant, which means it is now possible to compute equation 3.1.

All of the waypoints describing a handover trajectory are at this stage computed. The waypoints, grasp pose, and handover pose describe the pre-handover trajectory as was visualized in Figure 2.2 on page 7.

3.1.2 The camera service module

The functionality of *camera service* module is identical to its counterpart in the previous project [32], but it is explained here for the benefit of the reader. The *camera service* captures information about the workspace from the Intel RealSense D435i RGB-D sensor. It captures and provides RGB images as well as colored point clouds. The RGB images are captured at a resolution of 1280 by 720 pixels. The point clouds are post-processed with a hole filter and decimation filter using the librealsense2 library. The point clouds are thresholded such that all points closer than 0.1 meter and further away than 1.0 meter are discarded. The module was implemented in C++ for this project as opposed to Python 3 in the previous project.

3.1.3 Grasp generator module

Computing the pose of the end-effector, that results in the system grasping an object, is the task of the grasp generator. A method for computing task-oriented grasps was proposed in our previous project [32]. However, the hardware setup has since changed dramatically. The previously used grasp sampler [37] produces grasps that are often not reachable for the KUKA robot. It was, therefore, chosen to replace the task-oriented grasp generator method with a method where the individual parameters can be tuned according to the needs of this project.

The grasp generator takes as input a point cloud of the environment and a region where a grasp is desired, see the pipeline in Figure 3.5.

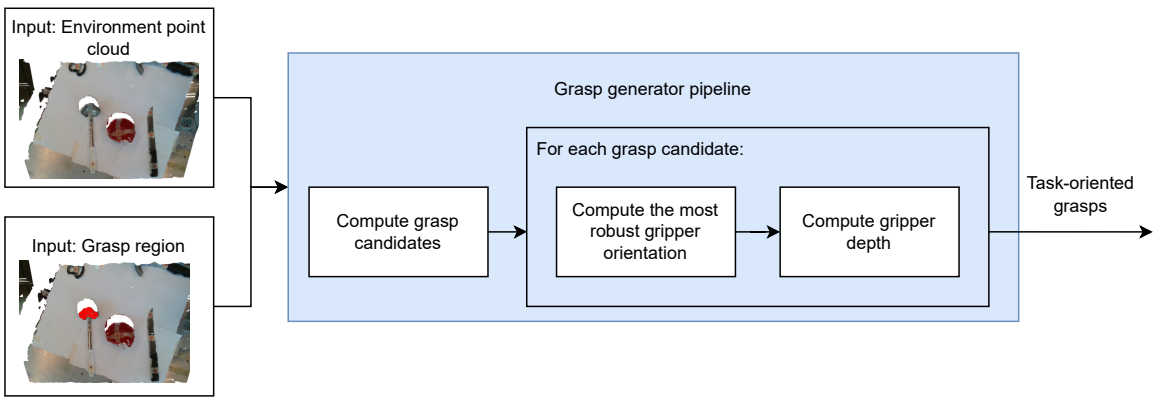
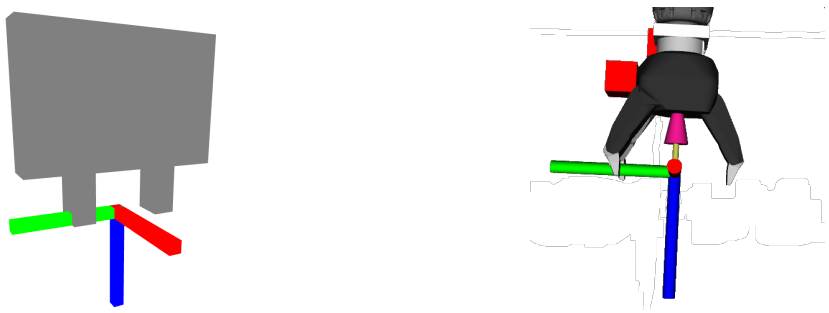


Figure 3.5: The grasp generator pipeline. Note that the grasp region is marked in red, while the rest of the point cloud is the environment point cloud.

When performing handovers, the giver should grasp the object in such a way that leaves the handle free for the receiver. Therefore, the grasp generator always generates grasps for the functional affordance e.g., pound, cut, support, etc., making it suitable for the task of handover. The grasp region is, therefore, found as the point cloud of the functional affordance of an object. It is found by performing affordance segmentation in an RGB image, which is then projected into a point cloud using the calibrated RGB and depth sensors of the Intel RealSense D435i sensor. The grasp region is, therefore, represented as a point cloud.

Grasp candidates are positions in 3D space which the system should investigate. Grasp candidates are found by downsampling the grasp region using a voxelization strategy with a distance between each voxel of 2 cm. Each point in the downsampled point cloud acts as a grasp candidate, where it is the task of the grasp generator to evaluate the feasibility of grasping of each grasp candidate.

Using the point cloud of the environment, each grasp candidate is investigated by checking if the gripper would be in collision with the environment at various rotations. A simplistic geometric model of the Robotiq gripper was modeled as three boxes, one box for the chassis and two boxes for the gripper's fingers, see Figure 3.6.



(a) (b)

Figure 3.6: (a) The simplistic geometric model of the gripper used for collision checking. (b) A visualization of the gripper in RVIZ, note how the coordinate frames are aligned alike.

For each grasp candidate, collision checking is performed with a set of discrete rotations for the gripper. The gripper is rotated around its z-axis such that the gripper always approaches the object from above. As noted by Fang et al. [37], humans tend to choose the grasp pose that allows for the biggest error. The grasp generator of this report, therefore, identifies the rotation around the z-axis which maximizes the distance of the gripper's fingers to the grasp region. This ensures that the grasp is perpendicular to the surface it will attempt to grasp.

The depth, at which the gripper should grasp the grasp candidate, is found once the orientation of the gripper has been determined. The depth is computed by evaluating the grasp candidate along with its orientation at various depths until a collision is found. The depth value is taken as the median of the depth values for which there is no collision. The

output of the system is a selection of grasp poses capable of grasping the desired part of an object.

3.1.4 Handover location module

The system must be capable of locating the receiver and computing a handover location, and this is the task of the *handover location* module. Hansen et al. [38] demonstrated that the handover location happens at the mid-point between the giver and the receiver. The system should, therefore, locate the receiver and compute the handover location as the mid-point between the giver and receiver. The *handover location* module also computes what is known as the giver's frame as proposed by Chan et al. [10]. In the giver's frame, the x-axis always points from the giver to the receiver, and the z-axis is the same as the world z-axis, which is orthogonal to the ground plane. The y-axis is the axis that completes the right-handed coordinate system. The giver's frame serves as the frame for which the orientation is ultimately defined, see Section 3.3.1. The overall pipeline can be seen in Figure 3.7.

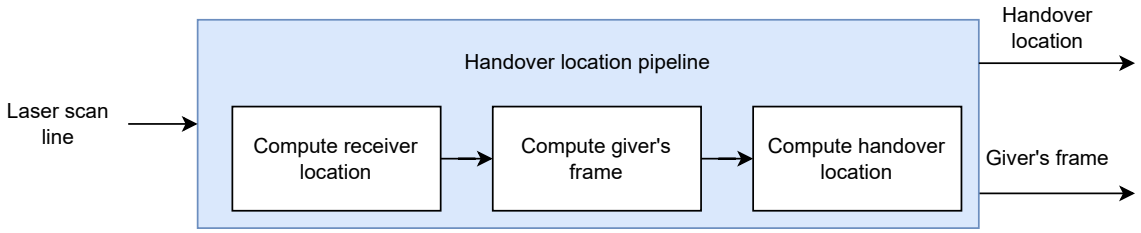


Figure 3.7: The pipeline for the computation of the handover orientation and the giver's frame.

The *handover location* module makes use of sensor readings from a Hokuyo URG-04LX-UG01 2D laser scanner. The laser scanner provides a line scan of its surroundings in the format of orientation θ and distance to detections, referred to as range. The line scan is filtered using a simple range threshold filter, that only allows points that are more than 0.5 m and less than 1.5 m away from the laser scanner. Afterwards, a median value of the remaining points is calculated. The median value represents the position of the receiver in 2D polar coordinates. The polar coordinates are turned into 3D coordinates using the following equations:

$$\begin{aligned} x &= range_{median} * cos(\theta_{median}) \\ y &= range_{median} * sin(\theta_{median}) \\ z &= 0.4 \end{aligned} \tag{3.3}$$

where the value of a z coordinate was manually chosen. As the last step, the receiver pose is transformed from the laser scanner frame into the world frame.

Using the position of the receiver, expressed in the world coordinate frame, a yaw angle by which the giver's frame should be rotated to point towards the receiver can be com-

puted using the Pythagorean theorem and trigonometric ratios of right-angled triangle, see Figure 3.8. The equation is described in (3.4).

$$yaw = sin^{-1}(\frac{y}{\sqrt{x^2 + y^2}}) \tag{3.4}$$

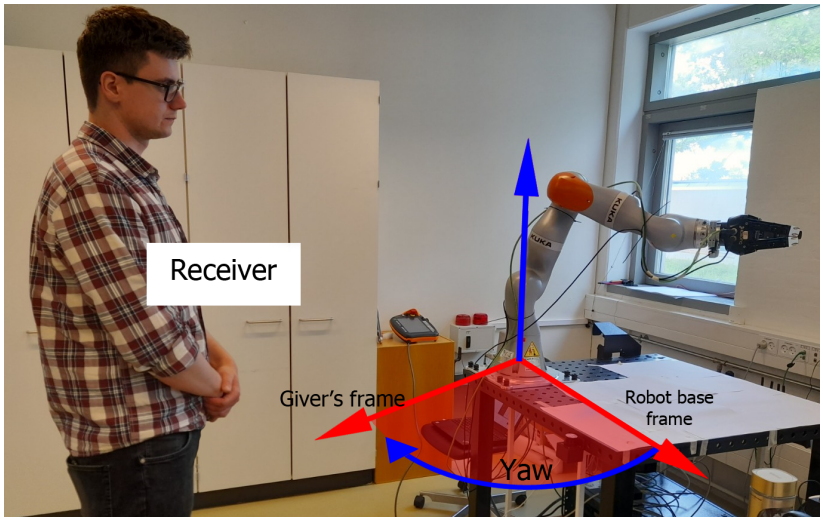


Figure 3.8: The world frame is rotated around the z-axis with yaw value, thereby producing the giver's frame.

The handover location can now be found in the giver's frame. As Hansen et al. [38] argues, the ideal handover location is at the midpoint between the giver and the receiver. The handover location is expressed in the giver's frame and relies upon the location of the receiver, which is also expressed in the giver's frame, where the x-axis always points from the giver to the receiver. As such, the handover location can be computed as follows:

$$\begin{aligned} x &= x_{receiver}/2 \\ y &= 0 \\ z &= 1.2 \end{aligned} \tag{3.5}$$

3.2 Affordance analysis with synthetic data

While extensive research has gone into improving the performance of deep learning models on existing affordance datasets, only limited work has been done on solving the limitations of the current affordance datasets. This section describes and explains how we propose to overcome the scalability issues of current affordance datasets, with synthetic data. In order to overcome the sim2real gap, we make use of synthetic data generated using domain randomization principles.

3.2.1 Creating the synthetic dataset

In order to detect affordances, a modified Mask-RCNN network was trained. However, to train the network, a dataset with ground truth pixel-wise segmentation masks, bounding boxes, and object class labels was required. For this purpose, a synthetic data generator was developed using the Unity game engine as the framework.

In order to generate data, the generator requires a set of 3D models to render. For this reason, a synthetic variant of the UMD [19] and IIT-AFF dataset [20] was assembled, as the UMD and IIT-AFF datasets are amongst, if not the most, widely used object affordance datasets. The dataset consists of 84 annotated objects, covering 19 object classes and 8 affordance classes present in both the UMD and IIT-AFF datasets. The affordance classes present are: Grasp, wide-grasp, cut, scoop, contain, pound, support, and engine.

The objects were collected as 3D meshes sourced from the YCB dataset [75], the website Sketchfab [1], and some were modelled by the authors. The objects were imported into the Unity engine as a mesh and a texture. Each object has two types of textures - a photo-realistic texture and an annotation texture. The photo-realistic texture is an unmodified model texture. This texture imitates the real-life texture of a given object. The annotation texture encodes the affordances of an object with an RGB value. The RGB values are constant for the same affordance e.g., any occurrence of the grasp affordance, whether it is a handle of a hammer or a knife, is encoded with the RGB value (0,0,255) which is the color blue. An example of an object with a photo-realistic and an annotation texture can be seen in Figure 3.9.

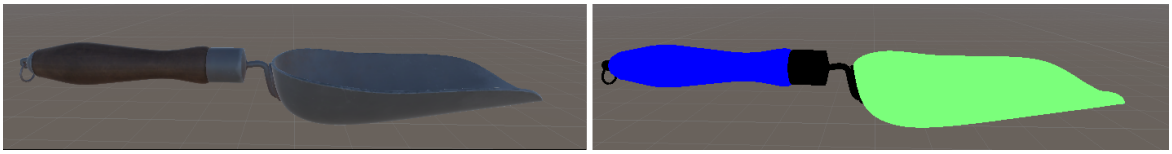


Figure 3.9: An object with a photo-realistic texture on the left, and the same object with an annotated texture on the right. Blue: Grasp affordance. Lime green: Scoop affordance. Black: No affordance.

In order to bridge the sim2real gap, the domain randomization technique was applied. The assumption of domain randomization is that a network trained on a dataset with a great variance will be able to also work on real-world data as the real-world data could be perceived by the network as yet another variation of the simulation. Inspired by work in [28] and [76], we vary the parameters listed in Table 3.1. Each parameter was sampled using a uniform distribution.

Geometric shapes of random dimensions and textures were rendered along with the 3D models. These geometric shapes were used as distractors. The role of distractors in a scene is to train a network for situations when not all foreground objects are classifiable [28]. In Figure 3.10, distractors can be seen in the bottom left figure. As can be seen from the corresponding ground truth on the right, distractors are labelled as background.

[1] https://sketchfab.com/feed

Randomized paremeter	**Values**
No. of objects in a scene	3 to 25
No. of distractors in a scene	5 to 10
Pose of objects and distractors	Random position and random orientation within camera view
Scale of objects and distractors	1 to 5 times
Object distortion	Each axis of an object is scaled independently of the other axes from 0.75 to 1.25 times
Textures	Sinusoid noise, checkerboard pattern, perlin noise, and photo-realistic textures
Light sources	1 to 3 with random poses
Light color	0 to 255 in all RGB channels
Light intensity	1 to 4 Unity game engine units
Screen res. width	400 to 600 pixels
Screen res. height	400 to 600 pixels

Table 3.1: Randomized parameters and their range.

Lastly, our generator only produces images where all objects are fully within the camera's field of view and are not occluded. Objects are always spawned within the camera's field of view, but after randomizing their shapes, some of the objects may exceed the boundaries of the camera's field of view. Such objects are removed from a scene. A similar approach is taken towards occlusions. If a previously spawned object is occluded by either a newly spawned object or a distractor, the previously spawned object remains and the occlusion-causing object is removed. Two examples of images produced by our generator can be seen in Figure 3.10.

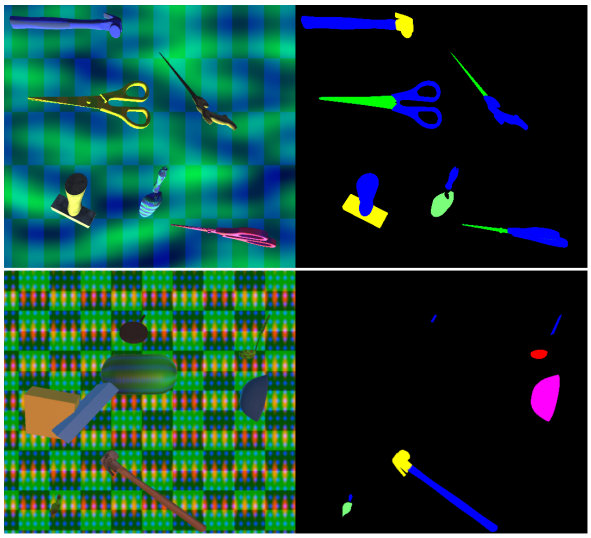


Figure 3.10: An example of images produced by our generator with their corresponding ground truth affordance masks. A bottom row shows a scene with distractors. As can be seen from the corresponding ground truth on the right, distractors are assigned a background label.

3.2.2 AffNet-DR

Once a synthetic dataset has been generated as described in Section 3.2.1, it can be used to train a deep learning method for affordance segmentation. It was decided to train a variant of AffordanceNet [22] since AffordanceNet often acts as the benchmark against which other affordance segmentation methods are compared. It also has the network architecture that most closely resembles the work of Chu et al. [29], who applied domain adaptation when training their affordance segmentation method named AffNet-DA on synthetic data. Furthermore, AffordanceNet performs both affordance segmentation and object detection, which allows a handover system to handover specific objects.

Overview of the AffNet-DR architecture

The proposed method named AffNet-DR (AffordanceNet - Domain Randomization) performs object detection and affordance segmentation with RGB images as input. Where Chu et al. [29] made use of images with a red channel, a green channel, and a depth channel (RG-D images), AffNet-DR does not. Simulated depth images often vary substantially from real-world depth images as real-world depth images contain significantly more noise [77, 78]. AffNet-DR, therefore, only uses RGB images as an input to the network, as the noise of current RGB sensors is negligible when compared to depth sensors.

The architecture of AffNet-DR is based on AffordanceNet [22] which in turn was based on Mask-RCNN [79]. The network, therefore, has a CNN backbone combined with a feature pyramid network that acts as the feature extractor for the whole network. Region proposals are generated and then fed into three task branches. The regression branch predicts a bounding box for each object in the input image, and the classification branch predicts the class of the object. Finally, the affordance mask branch predicts the affordance class of each pixel in the bounding box predicted by the regression branch.

As mentioned, AffNet-DR is a slightly modified version of AffordanceNet. These alterations are described here and can also be reviewed in Table 3.2. The VGG16 backbone present in AffordanceNet has been replaced with a ResNet-50 backbone, since ResNet based backbones has shown to result in higher quality extracted features which in turn results in better affordance segmentation [58, 59]. The mask branch has been reduced compared to AffordanceNet due to limited computational resources. The upsampling layers in AffordanceNet has 512 channels, this has been reduced in AffNet-DR to 128 channels. The kernel size of the first upsampling layers has also been reduced from 8 to 4 and the stride from 4 to 2. The network was implemented by the authors in the Pytorch framework and is available on the report's GitHub repository [31]. For a visualization of the network architecture, see Figure 3.11.

	AffordanceNet [22]	AffNet-DR
Backbone	VGG16	ResNet-50
No. of channels in the upsampling layers	512	128
Kernel size of the first upsampling layer	8	4
Stride of the first upsampling layer	4	2

Table 3.2: Differences between the proposed AffNet-DR and AffordanceNet [22]

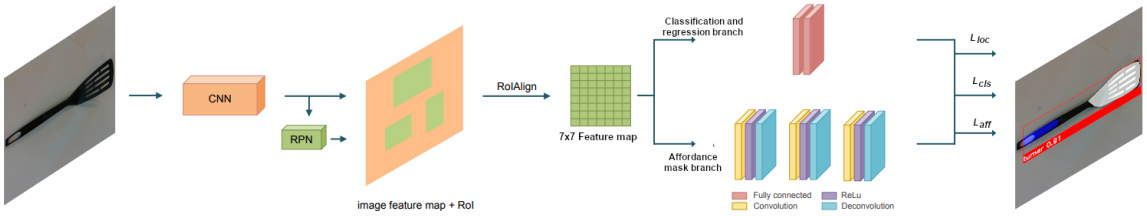


Figure 3.11: The architecture of AffNet-DR. The structure is identical to AffordanceNet [22], only some parameters of a few layers has changed and the backbone has been replaced with a ResNet-50 backbone. The image is adapted from [22].

Loss functions

The whole network is trained end-to-end with a multi loss function, which means that each task branch has a specific loss function. The system was trained on 30,245 synthetic images.

The classification task branch outputs $C + 1$ object categories, the categories are background and C is the number of object categories present in the dataset that the network is trained on. The loss function of the classification task branch L_{cls} is the cross entropy loss, calculated on the output, which is normalized with a softmax function, see equation (3.6).

$$L_{cls} = -\sum_{c=1}^{C} log \frac{exp(x_{n,c})}{exp(\sum_{i=1}^{C} x_{n,i})} y_{n,c} \tag{3.6}$$

where x is the prediction and y is the binary target.

Cross-entropy loss also serves as the loss function for the affordance mask task branch L_{aff}, but instead the cross-entropy is calculated for each pixel.

The regression task branch predicts four bounding box coordinates for each object. Therefore, the output is $N \times 4$, where N is the number of predicted objects in the RGB input image. The smooth L1 loss function was chosen as the function for the the regression

loss L_{loc} as is commonly done for bounding box regression. The regression task branch loss is computed as in equation (3.7):

$$L_{loc} = \sum_{i \in x,y,w,h} Smooth_{L1}(t_i^u - v_i) \tag{3.7}$$

where

$$Smooth_{L1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1. \\ |x - 0,5|, & \text{otherwise.} \end{cases} \tag{3.8}$$

The overall loss is computed as the sum of all branch specific losses, see equation (3.9).

$$L = L_{loc} + L_{cls} + L_{aff} \tag{3.9}$$

Post-processing

The segmented affordance masks and object detection, as computed by AffNet-DR, sometimes require post-processing to work with the rest of the system. One of the most common occurring issues is that a part of an object gets segmented with several different affordances. A post-processing procedure to alleviate this issue has, therefore, been programmed, see Figure 3.12.

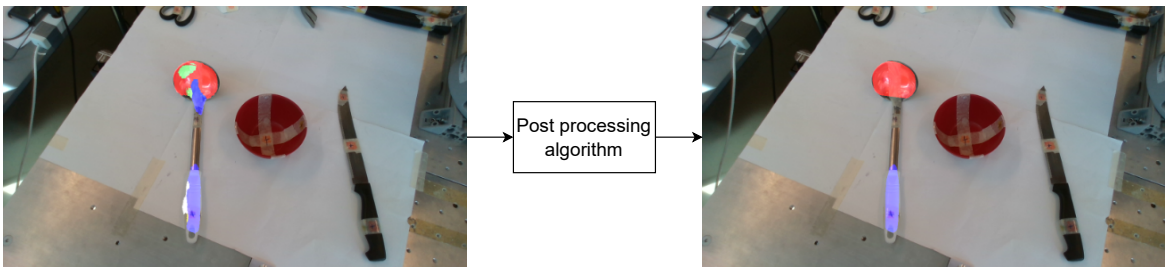


Figure 3.12: Left: Faulty affordance segmentation of a ladle as predicted by AffNet-DR. Note how there are too many affordances present in the object. According to the ground truth, only the grasp and contain affordances should be present. Right: Result after the post-processing algorithm. Red denotes contain affordance, blue denotes grasp affordance.

At first, the mask is treated with an erode operation with a kernel of size 3x3 to remove small patches of affordance predictions. From there, blob analysis is performed for each affordance mask separately. The largest blob is identified and stored, while the rest are discarded. Blobs are found for each affordance mask using the OpenCV library. Blobs are thresholded based on their size in relation to the bounding box of the object. All blobs less than 5% of the total bounding box size are discarded. Small blobs are essentially treated as uncertain predictions. A convex shape is found for each blob using the OpenCV contours function. Due to the post-processing operations, some blobs in the various affordances might overlap. All places where affordance masks overlap are discarded.

The complete pipeline of the *affordance analyzer* module is visualized in Figure 3.13

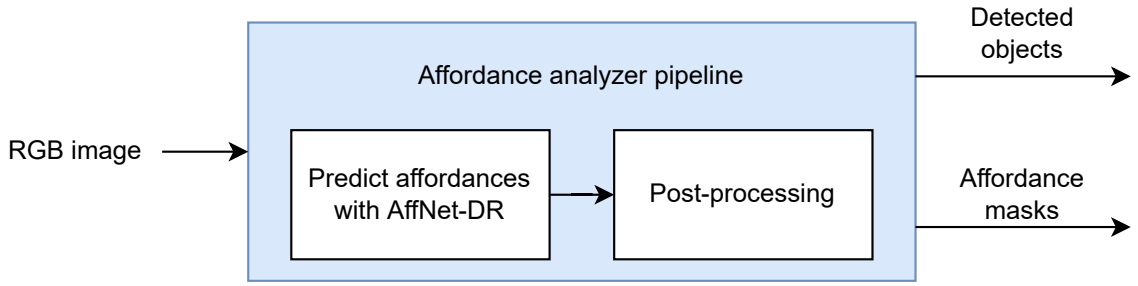


Figure 3.13: The pipeline of the *affordance analyzer* module.

3.3 The object handover orientation pipeline

As noted by Cakmak et al. [42], pre-defined object handover orientations are not appropriate since they fail to consider the object affordances. Instead, they should be learned from observing human-to-human handovers. Recently, however, a rule-based method that did consider object affordances was proposed by Ardon et al. [7]. Their method would always point the handle of an object towards the receiver. This harmonizes well with observations from human-to-human handovers, where it was observed that humans handover objects with the handle oriented towards the receiver [43, 44].

With this report, two methods for solving the problem of object handover orientation are proposed. Both methods are object independent and only consider object affordances in order to generate an object handover orientation. The first method learns handover orientations from human-to-human handover observations, while the second method uses pre-defined rules for orienting the affordances of an object.

3.3.1 A dataset of handover orientations

To learn object handover orientations from observing human-to-human handovers, an annotated dataset of observations is needed. However, the available datasets of human-to-human handovers by Chan et al. [10] or Ye et al. [80] either overlap poorly with the objects that can be detected by AffNet-DR or the dataset is not available to download. Therefore, an object handover orientation dataset was collected and annotated by the authors. The dataset consists of twelve object categories transferred from the giver to the receiver in human-to-human handovers while being recorded.

Data collection

A dataset with eight participants was collected. The participants had the role of the giver, while one of the authors acted as the receiver. The participants were asked to handover 12 different object categories, which are all present in the UMD dataset as well as in the synthetic dataset used to train AffNet-DR, see Section 3.2.1. Since humans might handover

objects in a configuration which does not consider the comfort of the receiver, as noted by Chan et al. [10], participants were asked to perform the handover with the perceived comfort of the receiver in mind. The giver is also asked to hand over the object with their right arm as not to introduce unwanted bias and variation between the observations.

The handovers were recorded with a Kinect V2 sensor, which provides RGB, depth, and skeleton tracking data, see Figure 3.14. A single-camera setup is used, with both the giver and receiver in the camera's field of view.

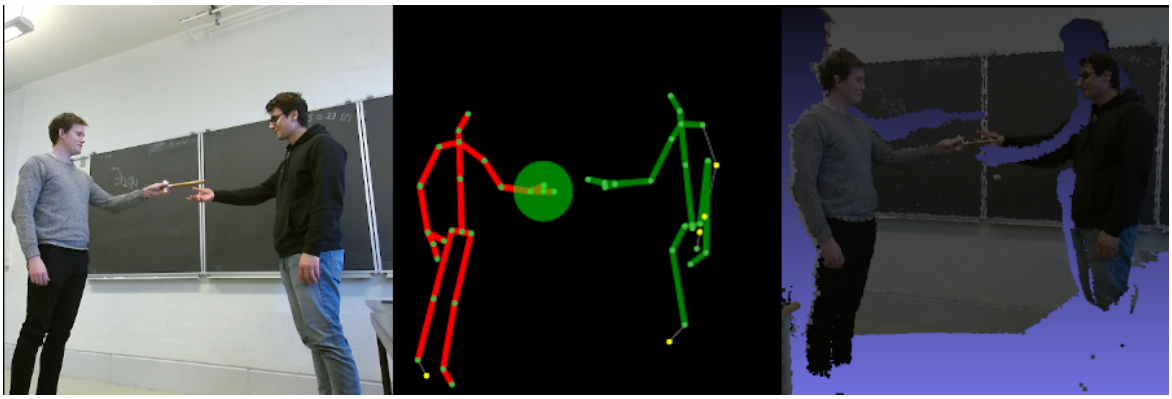


Figure 3.14: Relevant information such as RGB images (left), pose estimation and tracking of people (middle) and colored point clouds (right) can be captured using the Kinect V2 sensor.

Annotating the human-to-human handover dataset

For each observation recorded in the dataset, an object orientation must be annotated. At first, a coordinate frame for each of the 12 objects was manually defined, see Figure 3.15.

During the data collection, each handover was recorded as a single clip, consisting of a set of frames with RGB, depth, point cloud, and human tracking information present for each frame. Only one object orientation was annotated for each clip. To identify the handover frame, this report makes use of the method presented by Chan et al. [10], where the handover frame is found as the point in time when the distance between the humans as tracked by the Kinect V2 sensor is minimized, see Figure 3.16.

Figure 3.15: Objects from the synthetic dataset with their assigned frames, rendered in the Unity engine. The x-axis is red, the y-axis is green, and the z-axis is blue.

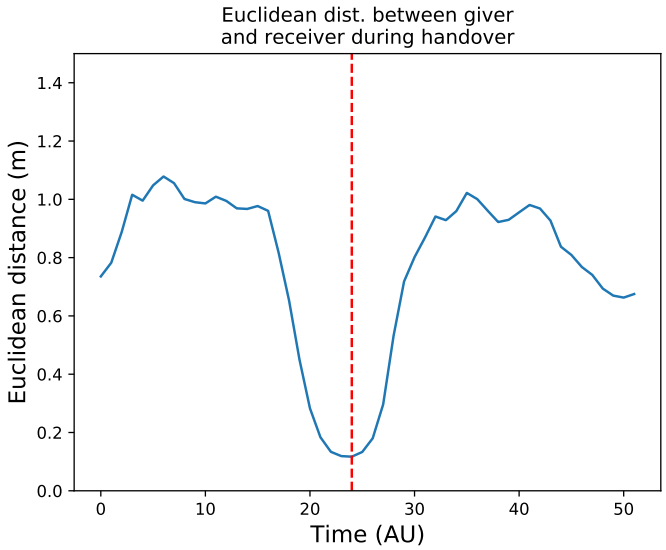


Figure 3.16: The handover frame is identified as the frame where the Euclidean distance between the tracked joints of the giver and receiver is minimized, marked in red.

The handover orientation was manually annotated using the software named labelCloud [2], which allows the user to annotate a 9-DOF transformation consisting of a position, orientation and scale. The annotation process is done by fitting a bounding box to the object being handed over. The labelCloud software was slightly modified to show the orientation of the three axes in 3D space, see Figure 3.17.

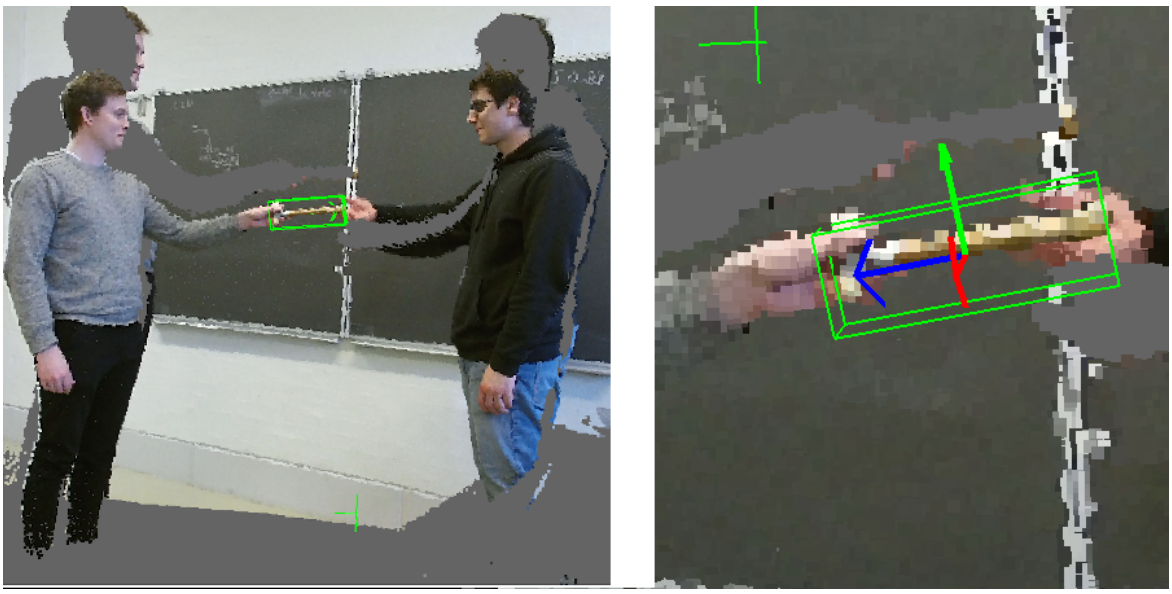


Figure 3.17: The position, orientation, and scale of a bounding box are fitted to the handover object at the moment of handover. An object is represented by a 9-DOF bounding box in 3D.

The orientation as annotated with labelCloud is defined in the coordinate system of the Kinect sensor. The annotated orientation needs to be transformed into a coordinate system that is easily transferred to a robotic system. The annotated orientation is, therefore, redefined in the coordinate system of the giver. The giver's coordinate frame was computed similarly to Chan et al. [10]. The center of the giver's coordinate frame is located in the torso of the giver, with the x-axis defined as the unit vector pointing from the torso of the giver to the torso of the receiver. The z-axis is found as the unit vector pointing from the torso to the head of the giver, the y-axis is the unit vector that completes a right-handed 3D coordinate system, see Figure 3.18. In some cases, the skeleton tracker of the Kinect V2 sensor might fail. In those cases, coordinates of the giver's head and torso as well as the receiver's torso coordinate can be manually defined.

[2] `https://github.com/ch-sa/labelCloud`

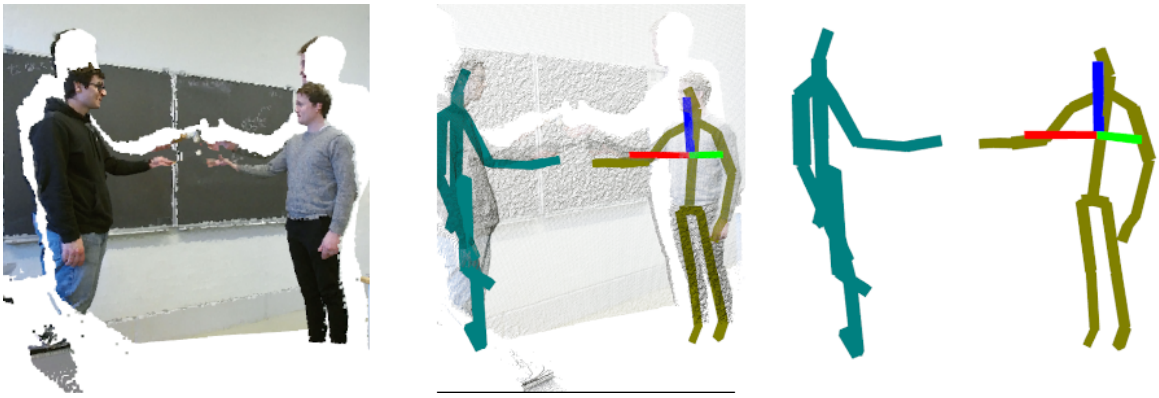


Figure 3.18: A reference frame for the handover orientation is computed by making the x-axis (in red) point from the giver's torso towards the receiver. The z-axis (in blue) points from the torso to the head of the giver. The y-axis (in green) is found as the axis that completes the right-handed coordinate system.

At this point, the dataset consists of a set of observed handover orientations for each of the 12 object categories, but a single goal orientation is needed for each item. Finding the goal orientation associated with each object is done by computing the mean handover object orientation for each object respectively.

There exist a plethora of methods for computing the mean orientation, but no single best method for doing so [81, 11]. Intuitively, one might compute the mean of rotation matrices, but the result is no longer a valid rotation matrix [82]. Computing the arithmetic mean of either quaternions or rotations vectors is another solution that has shown to work well when differences in observed orientations are small [83]. However, this performs poorly if the orientation differences are large. Furthermore, these methods also suffer from issues of discontinuities [11].

Chan et al. [11] argue that the mean $\bar{x}$ of a set, considered from a statistical point of view, is the value that minimizes the sum of distances from $\bar{x}$ to all observations x_i in the set. Therefore, Chan et al. suggest that the task of finding the mean orientation can be cast as a minimization problem, see equation (3.10).

$$\bar{x} = \arg\min_{x'} \sum_i dist(x' - x_i) \tag{3.10}$$

where $dist(x' - x_i)$ is a distance function that needs to be defined.

Chan et al. found that their method *DistMinRPY*, where the orientations are represented as roll, pitch, and yaw values, is suitable for computing the mean object handover orientations. Unfortunately, the distance function used for computing the mean orientations was left ambiguous, which renders their method *DistMinRPY* unimplementable. Therefore, this report will instead implement a similar method but based on quaternions called *DistMinQuat*. The method for computing the mean handover orientation, represented in quaternions, from a set of observations can therefore be computed as follows:

$$\bar{q} = \arg\min_{q'} \sum_{i} dist(q' - q_i) \tag{3.11}$$

where

$$dist(q' - q_i) = min\{||q' - q_i||_2, ||q' + q_i||_2\}$$

The distance function was selected as suggested by Hartley et al. [81]. The distance function by Hartley et al. considers the fact that the positive quaternion q is equal to the negative quaternion $-q$.

The *MinDistQuat* method was implemented in Python. The algorithm is initialized with the initial solution $q_0 = [x, y, z, w]$, where values $\{x, y, z, w\}$ are randomly generated and lie within the range from -1 to 1. In order to solve the minimization problem, the *scipy* library is used[3]. To optimize for $\bar{q}$ in $\mathbb{R}^4$, the function is restarted 50 times, each time with a new initial solution q_0. After 50 iterations, the $\bar{q}$ that provides the minimum sum of distances is considered to be the mean handover orientation.

The computed mean object handover orientation for all 12 objects can be seen in Table 3.3 and Figure 3.19.

Object	Quaternion				RPY		
Bottle	-0.66	-0.05	-0.68	0.30	-100.62	-6.89	-103.73
Bowl	0.67	0.16	0.71	-0.12	-112.53	-27.01	-118.87
Cup	-0.10	0.70	-0.19	-0.68	12.69	-90.57	24.05
Hammer	0.05	0.60	-0.32	-0.73	-5.69	-75.74	40.83
Knife	0.05	-0.73	0.16	0.66	6.81	-94.47	20.60
Ladle	-0.04	0.72	-0.08	-0.68	4.71	-93.03	10.81
Mallet	-0.73	-0.05	0.68	-0.07	125.22	9.36	-116.73
Mug	0.25	0.70	0.32	-0.59	-33.63	-93.44	-43.25
Scissors	0.00	-0.83	0.26	0.50	0.48	-114.49	35.93
Scoop	0.09	-0.72	0.15	0.67	11.80	-92.79	19.73
Spatula	0.18	-0.72	0.14	0.65	23.23	-94.10	18.86
Spoon	0.14	-0.73	0.23	0.63	18.40	-95.37	29.80

Table 3.3: Computed mean handover orientations.

[3]https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html

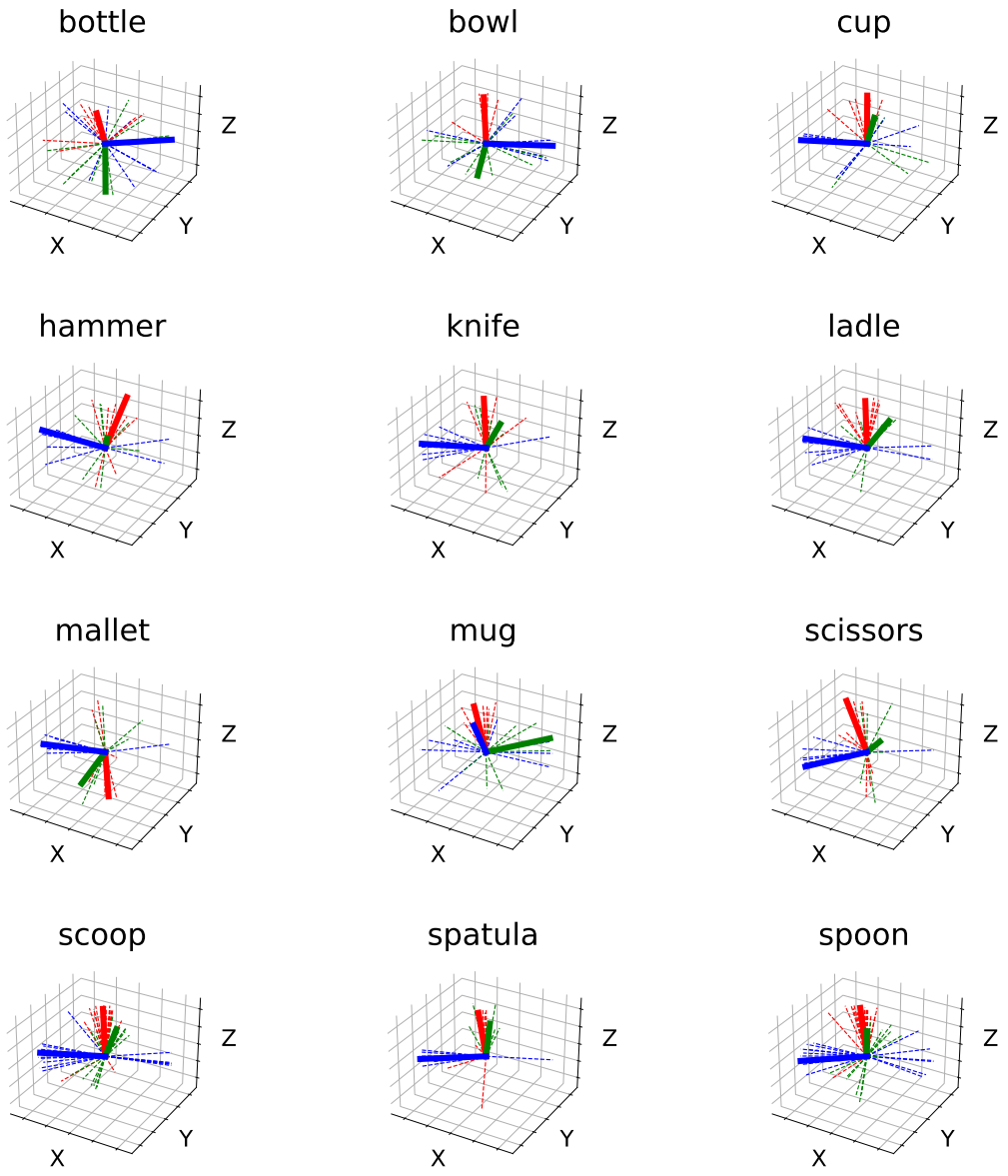


Figure 3.19: Computed mean handover orientations. The dotted axes are observations. The solid thick axes are the computed means.

3.3.2 Handover object orientation pipeline

As mentioned in section 2.4, this report presents two methods for computing object handover orientations. One method relies on the human-to-human handover dataset to generate handover orientations, the other is a rule-based method with pre-defined handover orientations. Both methods for computing proper object handover orientations presented in this report rely solely on object affordances without a need for considering the object class. Both the rule-based and the observation-based method should, therefore, be able to generalize to all objects that share the same affordances. Several of the steps in the algorithm are the same in both methods. At first the observation-based method is presented, followed by the rule-based method.

The observation-based method

The observation-based method learns proper handover orientations from the dataset of mean handover orientations as computed in section 3.3.1, Table 3.3. The handover orientation pipeline produces both a goal object handover orientation as well as a pose estimate of the object in the world. The overall pipeline is depicted in Figure 3.20

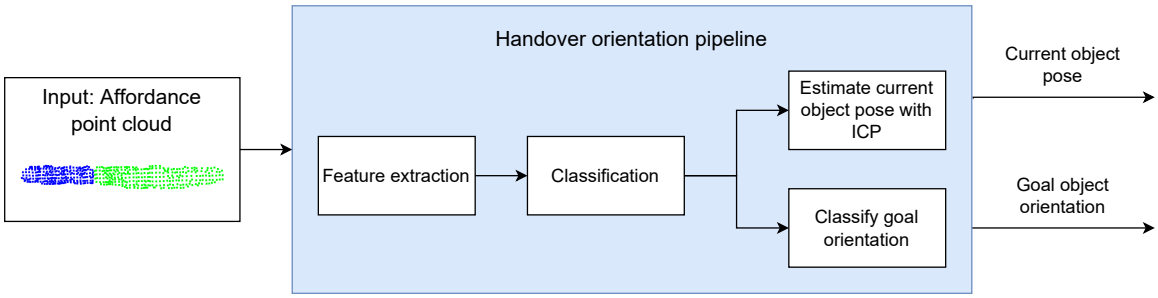


Figure 3.20: Implemented pipeline for the observation-based object handover orientation.

The input to the system is a point cloud where each point has a predicted affordance, which will be referred to as affordance point clouds hereinafter. Affordance point clouds are found by first performing affordance segmentation with AffNet-DR in the RGB image, captured by the Intel RealSense D435i sensor, and then projecting the affordance masks found in the RGB image into a point cloud. Projection into a point cloud is straight forward as the RGB and depth sensors of the Intel RealSense D435i are calibrated.

Feature extraction and classification

The pose estimation is found using a modified iterative closest point (ICP) algorithm, therefore, a source point cloud is needed. The source point cloud is selected based solely on the affordances present in the affordance point cloud as segmented by AffNet-DR. The identification of the correct source point cloud is treated as a classification problem.

A feature vector is extracted and used for the classification task. The feature vector is computed as the one-hot-encoded vector that represents the affordances present in the affordance point cloud. Since AffNet-DR can segment 7 different affordance classes, the one-hot-encoded vector is a row vector with 7 binary values, where 0 means that the given affordance is not present, and 1 means that the given affordance is present in the affordance point cloud. A k-nearest algorithm is then used for classification.

Since the method does not rely on object classes but solely on the object affordances, the different objects are grouped together based on their affordances. As such, the k-nearest algorithm will only output a single goal object handover orientation and source point cloud based on the object affordances, see Table 3.4 where members of the same group are color-coded together. The k-nearest algorithm outputs 1 of 7 different predictions. The novelty is that the learned orientations from a knife handover can be applied to all objects that share the combination of grasp and cut affordances.

	Grasp	**Cut**	**Scoop**	**Contain**	**Pound**	**Support**	**Wide-grasp**
Spatula	✓					✓	
Knife	✓	✓					
Scissors	✓	✓					
Hammer	✓				✓		
Mallet	✓				✓		
Spoon	✓		✓				
Scoop	✓		✓				
Bottle							✓
Ladle	✓			✓			
Bowl				✓			✓
Cup				✓			✓
Mug				✓			✓

Table 3.4: The 12 objects present in handover orientation dataset. The colors indicate how they have been grouped together based on their object affordances. Best viewed in color.

Since one object from each of the 7 groups should be able to represent the other objects in the same group, only one source point cloud and one goal object handover orientation is needed per group. The authors chose to use the source point clouds for the following objects: Spatula, hammer, scoop, bottle, ladle and cup.

The associated goal orientation can be read in Table 3.3 on page 50. The associated source point clouds can be seen in Figure 3.21.

Estimation of current pose

The goal object handover orientation for the robot is known, once the registered affordance point cloud has been classified with the k-nearest neighbour classifier. The next step is to

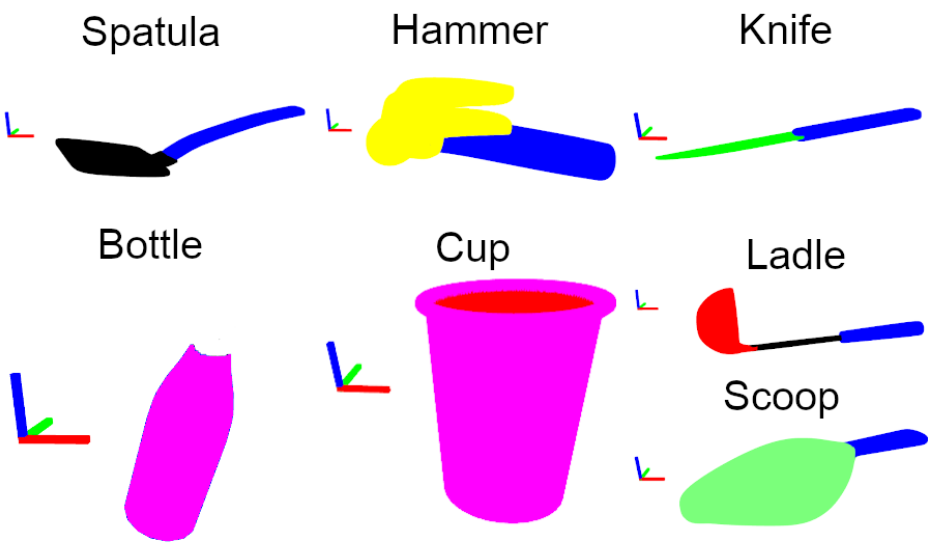


Figure 3.21: Source point clouds used for the ICP algorithm with their respective affordances. All point clouds are sampled from the object meshes used to train AffNet-DR. The point clouds has also been rotated with the goal object handover orientation. All orientations are shown in the giver's frame.

estimate the current pose of the object in the world coordinate frame, such that the goal end-effector pose can be computed as described in section 3.1.1 . To this end, a modified ICP alorithm is used.

The approach is most similar to the work done by Akizuki et al. [84], who proposed to perform pose-estimation using affordance cues. In a point cloud where each point is labelled with an affordance, a source point cloud is aligned using a particle swam optimization method.

For this report, the particle swam optimization method is replaced with a modified ICP algorithm. Traditionally, the ICP algorithm aligns a source point cloud to a target point cloud using x, y and z information only. This method extends the traditional ICP algorithm to also consider affordance information. The affordance information is available in the affordance point cloud, which is an input to the object handover orientation pipeline. The method makes use of the ICP algorithm outlined below.

```
while i < maxIterations || meanError < convergenceCriteria do
    Compute nearest neighbour for all points
    Compute rotation and translation
    Compute mean error
end while
```

In the base ICP algorithm [85], the nearest neighbour is computed for each point in the source point cloud as the closest point in the target point cloud measured with the Euclidean distance. This is changed in this implementation in order to take advantage of the object affordances found during the affordance segmentation. Instead, the nearest neighbours can only be identified for points which share the same affordance. The nearest neighbour algorithm was implemented using the *scikit-learn* library [4].

The transformation is found computing the least-squares fit of the source point cloud to the target point cloud. First, the source and target point clouds are centered on their respective centroids. Then, the rotation is found by using singular value decomposition, see equation (3.12).

$$\begin{aligned} H &= source^{T}_{centered} \cdot target_{centered} \\ U, S, V &= SVD(H) \\ R &= V^{T} \cdot U^{T} \end{aligned} \tag{3.12}$$

where R is the rotation. Finally, the translation is found as the dot product of the source and target centroids.

Using the semantic understanding of the objects allows the ICP algorithm to provide robust pose estimations, but it requires a sound strategy for choosing a source point cloud to align the target point cloud with. An output example of the modified ICP algorithm can be seen in Figure 3.22

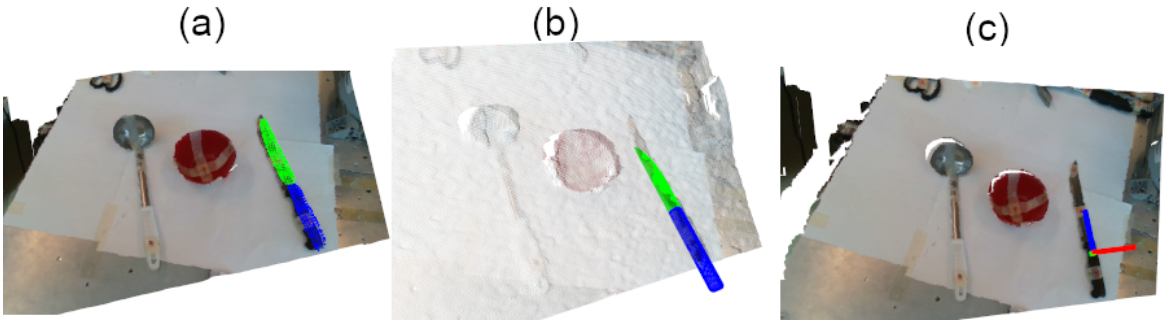


Figure 3.22: An overview of the ICP algorithm performing pose estimation on the affordances of a knife. (a) The affordance point cloud is marked in green and blue. (b) The transformation as computed by the ICP algorithm applied to the predicted source point cloud. (c) Estimated object pose.

The rule-based method

Rule-based methods that consider the affordances of objects might, as previously argued, be suitable for assigning goal object handover orientations. Therefore, this section proposes an affordance-aware rule-based method. The overall pipeline can be seen in Figure 3.23.

[4] `https://scikit-learn.org/stable/modules/neighbors.html`

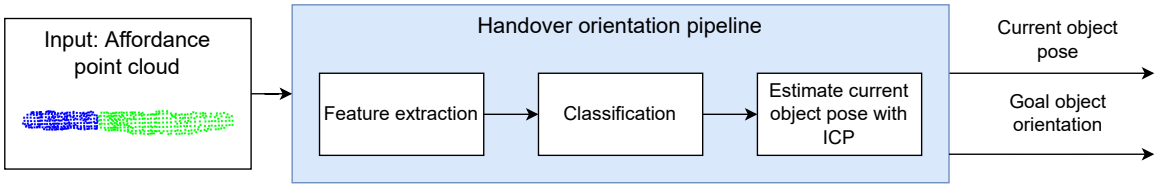


Figure 3.23: Implemented pipeline for the rule-based object handover orientation.

The overall pipeline is similar to the observation-based method but heavily simplified. The rule-based method needs to produce orientations based on affordances. As such, a set of rules must be defined. Similar to Ardon et al. [7], one of the rules is to orient the handle of a tool towards the receiver. This rule alone covers 14 out of the 19 objects that AffNet-DR can detect, where they all share the grasp affordance. The last five object classes are a cup, a bowl, a mug, a bottle, and a pot. Common to these five object classes is that they have the wide-grasp as well as contain affordances. It is, therefore, decided that the cup, bowl, mug, bottle and pot object classes should always be oriented vertically in a manner that does not spill or empty the content of the objects.

In order to confirm that these two rules indeed simulates the manner in which humans handover the above mentioned objects, an analysis of the collected handover orientation dataset was conducted. The point of this analysis was to confirm that items with the handle are usually oriented with the handle pointed towards the receiver, and the items with contain affordance are oriented such that a spillage is prevented or minimized. To achieve this, a visual analysis of the observations in the collected dataset was conducted.

As was discussed in section 2.3, identifying the way people tend to orient various objects is not straightforward. For some objects, different axes of rotations have varying degrees of importance. Certain axes of rotations might be equivariant, which means that rotations around those axes are of little importance. Other axes might be bi-polar or epi-polar, which means that people tend to agree on one or two orientations for those axes. It is, therefore, important to identify which axes matter.

The important axes were found by visualizing the x-, y-, and z-axes at the moment of handover for each object. If an axis appeared to be oriented similarly between the collected observations for the object, the axis was appointed the axis of importance. For example, see Figure 3.24. This is an example of visual analysis of a cup. From this figure, it can be seen that x-axes seem to be in agreement compared to z- and y-axes, for which there is a little consensus on their orientations. From this, it can be concluded that for a cup, the x-axis is the axis of importance. It should be noted here, that if an axis of importance appears to be bipolar, it does not disregard its relevancy.

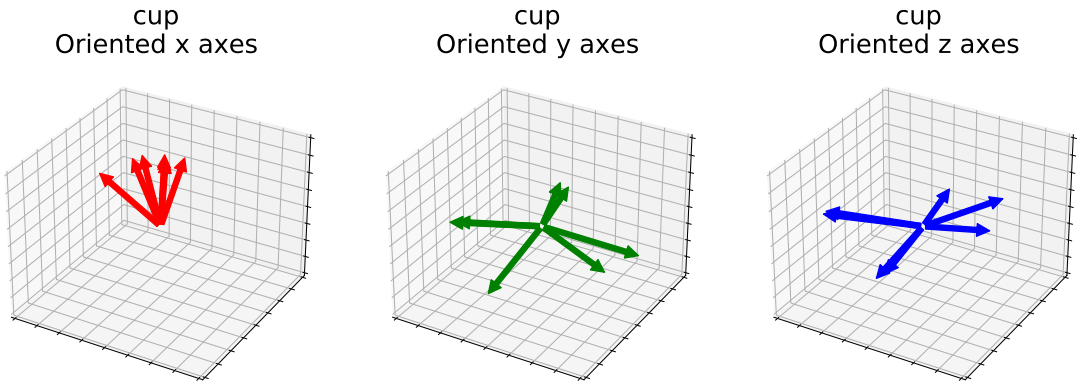


Figure 3.24: Observed orientations of individual axes for a cup. Note how there is a strong agreement on the orientations of the x-axes but none for the y- and z-axes.

After each object has been investigated following the visual analysis described above, which result can be reviewed in appendix A, it has been found that there are three categories. These are:

- **Group A** - People tend to agree on the orientation of the z-axis, but the z-axis may be bi-polar
  - Here belongs a spatula, a knife, a hammer, and a mallet.
- **Group B** - People tend to agree on the orientation of all three axes, with a strong agreement on the x-axis pointing upwards and the y- and z-axes being bi-polar.
  - Here belongs scissors, a scoop, a ladle, and a spoon.
- **Group C** - People tend to agree on the orientation of the x-axis, with the x-axis pointing upwards.
  - Here belongs a cup, a mug, a bowl, and a bottle.

From the categories listed above, it can be seen that groups A and B share the z-axis as the axis of importance. To remind the reader, the z-axis was set to point from the grasp affordance towards the functional affordance of the the object i.e, from the handle towards the utility part of the object. This means that objects of groups A and B can be collected under a single group AB. For objects of group C, which all share the contain affordance, people agree that the orientation of the x-axis is of importance. For objects of group C, the x-axis was defined as the vector that is orthogonal to the bottom of the inside of the contain affordance, thereby, pointing straight out of the cup, mug, and etc.

However, grouping object based on their axes of importance is not enough to confirm that the two proposed rules described the handover orientations of these objects. In order

to do that, it must be shown that the axes of importance for groups AB and C also point in the overall same direction. This was done by taking the computed mean handover orientations for each object of the corresponding group, rotating a unit frame by it, and then computing the Euclidean distance between the rotated axes of importance. To visualise this better, see Figure 3.25. In here, rotated z-axes for objects of group AB are visualised. It can be observed that while each z-axis points towards slightly different point, they do generally agree on the orientation. This is further confirmed by computing the Euclidean distances between the axes, see Table 3.5.

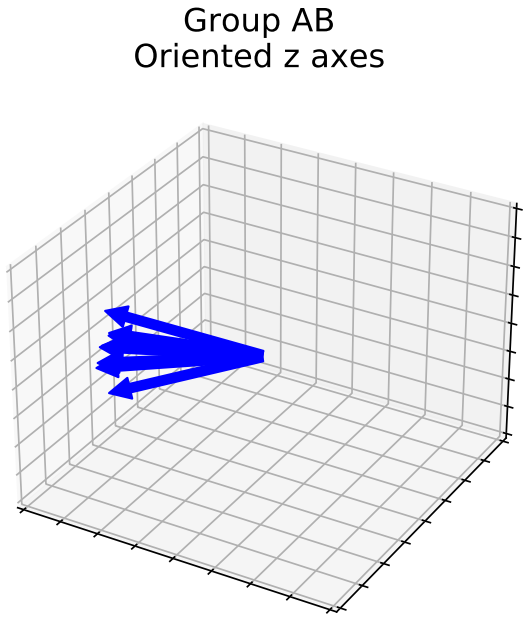


Figure 3.25: The oriented z-axes of the group AB objects.

Group AB	spatula	knife	hammer	scissors	scoop	mallet	ladle	spoon
spatula	0.00	0.16	0.40	0.26	0.13	0.28	0.29	0.08
knife	0.16	0.00	0.35	0.35	0.05	0.12	0.14	0.23
hammer	0.40	0.35	0.00	0.65	0.33	0.38	0.37	0.42
scissors	0.26	0.35	0.65	0.00	0.35	0.42	0.44	0.28
scoop	0.13	0.05	0.33	0.35	0.00	0.17	0.18	0.19
mallet	0.28	0.12	0.38	0.42	0.17	0.00	0.02	0.35
ladle	0.29	0.14	0.37	0.44	0.18	0.02	0.00	0.36
spoon	0.08	0.23	0.42	0.28	0.19	0.35	0.36	0.00

Table 3.5: Computed distances of the oriented z-axes for group AB. Distance is given as the Euclidean distance in the unit sphere. Each number represents the distance between the object written in the corresponding row and the object written in corresponding column. Hence, the matrix is diagonally symmetric.

Similarly, Figure 3.26 visualizes the rotated x-axes for objects of group C. Table 3.6 shows computed distances.

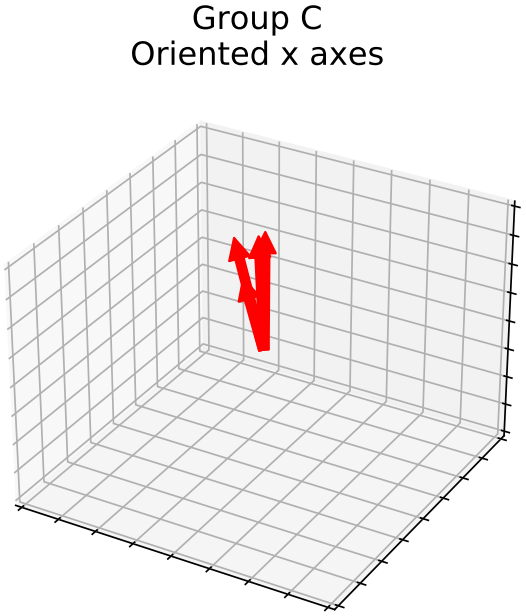


Figure 3.26: The oriented x-axes of the group C objects.

Group C	cup	mug	bowl	bottle
cup	0.00	0.19	0.07	0.49
mug	0.19	0.00	0.14	0.41
bowl	0.07	0.14	0.00	0.42
bottle	0.49	0.41	0.42	0.00

Table 3.6: Computed distances of the oriented x-axes for group C. Distance is given as the Euclidean distance in the unit sphere. Each number represents the distance between the object written in the corresponding row and the object written in corresponding column. Hence, the matrix is diagonally symmetric.

From Tables 3.5 and 3.6, it can be concluded that objects not only share the same axis of importance, but they also agree on the orientation of this axis. Therefore, it is confirmed that the two proposed rules, orient a handle towards a receiver if an object has the grasp affordance and orient an object vertically if an object has the wide-grasp affordance, sufficiently describe the way people orient the objects of our dataset. The system, therefore, only needs to distinguish between whether an object has the grasp or wide-grasp affordance. A simple `if/else` statement, therefore, covers all the objects.

The pose estimation uses the same modified ICP algorithm explained in the observation-based method, see Section 3.3.2, a source point cloud is, therefore, needed. Two source

point clouds are manually constructed, one for the grasp class and one for the wide-grasp class. The grasp source point cloud is constructed such that the handle is oriented along the x-axis in the giver's frame i.e., towards the receiver, and the wide-grasp source point cloud is constructed such that the cup, mug, etc. does not spill the content, see Figure 3.27.

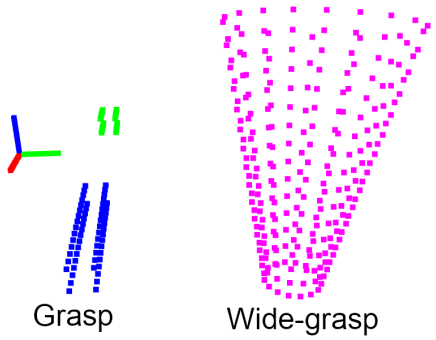


Figure 3.27: The two constructed source point clouds for the rule-based method. Both point clouds are visualized in the givers frame, where the x-axis is pointing towards the receiver. Blue indicates the grasp affordance, pink is the wide-grasp affordance and green represents all other affordances.

4 - Testing

This section describes tests and test procedures carried out to verify the performance of the proposed system.

4.1 Affordance segmentation

The test described in this section evaluates the performance of the AffNet-DR network, described in section 3.2, in order to confirm that the requirement of a F_β^w score of more than 0.546 has been achieved. For this test, the post-processing step is omitted. This is done in order to evaluate unmodified output of the AffNet-DR network.

4.1.1 Information flow

AffNet-DR is evaluated on the real-world UMD dataset to confirm that the network, trained on synthetic data only, generalizes to real world data. The test uses the validation set of the UMD dataset category split. Each RGB image in the validation set is processed with AffNet-DR in order to perform affordance segmentation. The affordance segmentation prediction is then evaluated against the ground truth of the UMD dataset with the F_β^w score. The overall information flow can be seen in Figure 4.1

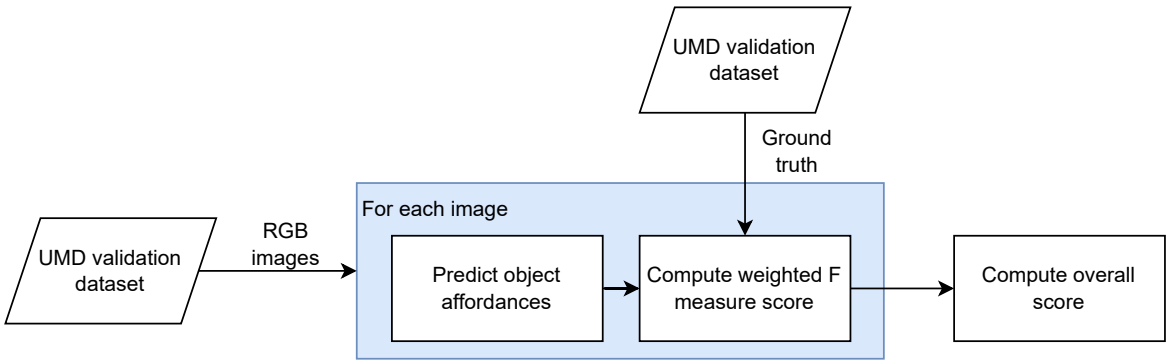


Figure 4.1: The flow of information during the testing of the AffNet-DR method.

4.1.2 Metrics

The affordance segmentation capabilities of AffNet-DR is evaluated with the following metric.

- F_β^w as defined by Margolin et al. [50], described in section 2.2.

4.1.3 Procedure

AffNet-DR was evaluated on the category split validation set of the UMD dataset [19].

1. For all images in the UMD validation set:
   (a) Predict affordances with AffNet-DR.
   (b) Compute a F_β^w score for each affordance mask ground truth.
2. Compute average F_β^w score on a per affordance class basis.
3. Compute the overall F_β^w score as the average of the averages of each affordance class.

In order to evaluate how well AffNet-DR overcomes the sim2real gap, the same procedure is perfomed with a baseline network which has the same architecture as AffNet-DR, but it is trained on the real UMD dataset. In addition to the baseline, results of AffNet-DR will also be compared to the reported results of AffordanceNet [22] and the reported AffNet-DA results [16]. The results can be found in section 5.1.

4.2 Object handover orientation

The effectiveness of the two proposed object handover orientations methods, the rule-based and the observation-based, are evaluated with the user study described in this section. A random-based method, with randomly sampled orientations, will server as the baseline.

4.2.1 Information flow

Both the rule-based and observation-based methods make use of object affordances as segmented by the *affordance analyzer* module. Object affordances found in RGB images are used as the input to the *handover orientation* module. The *handover orientation* module then outputs an orientation for the object based solely on the segmented object affordances. As described in section 3.1.1, information such as the current pose of the object and the grasp pose is needed. The information flow is visualized in Figure 4.2.

An object set used in this user study consists of five objects - a cup, a bowl, a mallet, a spoon, and a spatula. During an experiment, the objects are placed into their pre-defined pose on the worktable.

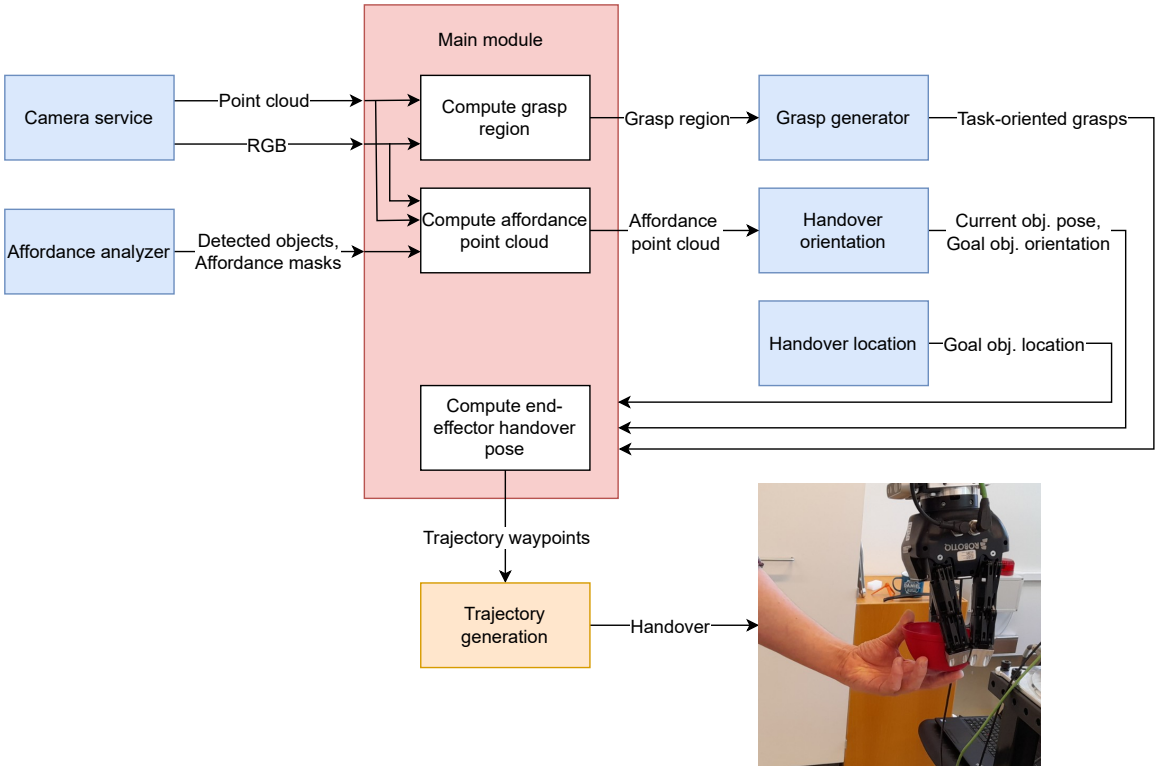


Figure 4.2: The information flow between the various sub-modules during the handover object orientation test.

In order to facilitate a smooth human study, grasp poses, affordance segmentation, and object pose estimations are performed offline and prestored, which is possible as the objects are always located at their pre-defined poses at the beginning of the test. Using pre-computed information effectively eliminates any errors that might occur from faulty affordance segmentation, grasp generation, or pose estimation, which allows the user to fully concentrate on the orientation aspect of the study.

4.2.2 Metrics

After experiencing each method, users will be asked to fill out a questionnaire, which consists of five questions evaluated on 5-point Likert scale. As suggested by Mangold et al. [86] and Hankinson et al. [87], a 5-point format was chosen in order to reduce the frustration levels of the users while answering the questionnaire, which could increase the response rate and the quality of the responses. The questions of the questionnaire are:

- How safe did you feel during the handover?
- The objects were oriented in a way that was comfortable for me when grasping.

- I would orient the objects similarly when performing handovers.
- I agree with the way the objects were oriented.
- How well would you say you understood the handover process?
- How natural did you find the object orientations when you grasped the items?
- How appropriate did you find the object orientations when you grasped the items?

After experiencing all three methods, the user will be given a final questionnaire in which the user will indicate their preferred method.

4.2.3 Procedure

Each participant will experience all three handover orientation methods. The order in which these methods are presented to the participant is randomized between the participants. The participant does not know which method is applied. One trial for each method consist of receiving all objects of the object set. The procedure for a trial is as follows:

1. The receiver is asked to stand at a pre-defined spot.
2. For each of the five items:
   (a) One of the authors executes the handover program with pre-computed information.
   (b) The robot grasps the object by its functional affordance.
   (c) The system hands over the object with the computed orientation.
3. The user is asked to evaluate the handover experience by answering the aforementioned questionnaire.

The questionnaires used for the user study can be found at the link in the footnote[1]. The results of the test can be found in section 5.2

4.3 Full system test

This section describes the full system test with all of the modules working together. The test is conducted to evaluate the performance of the whole system. As mentioned in the requirements section 2.5, the minimum required success rate is set to be 75 %.

[1] `https://docs.google.com/forms/d/1WKT7d5Ixx4REIlR6rnA_LYZn9pRlOXT4g4KX_hxU9SU/edit`

4.3.1 Information flow

For the full system test, information flow between the various modules of the system is visualized in Figure 3.2. Both the rule-based and observation-based methods are tested.

4.3.2 Metrics

The full handover pipeline is evaluated with the the following metrics:

- Success: Binary, the robot is able to grasp the object by its functional affordance and transport it to the computed end-effector handover pose, where it is handed over to the receiver without any failures.
- Failure to generate grasp: Binary, the *grasp generator* module could not produce any task-oriented grasps.
- Failure to grasp: Binary, the robot failed to grasp the functional affordance or failed to hold the object.
- Failure to generate trajectory: Binary, grasps were generated, but no feasible trajectory could be generated for the best-scoring grasp of the *grasp generator* module.
- Failure to handover: Binary, any failure after the robot has moved the object from the initial position to the handover location resulting in the object not being transferred from the giver to the receiver.

A test run is considered successful even if the object is not oriented correctly at the handover location. However, such occurrences will be noted down nonetheless and reported as part of the results. The reason why test runs with inappropriate orientations are still considered a success is that inappropriate orientations should not decrease the success rate of handover, only their perceived quality.

While affordance detection is part of the full system, and as such it should be tested along with it, AffNet-DR is evaluated separately on the UMD dataset. Still, even if AffNet-DR manages to achieve impressive performance on the UMD dataset, the network may not able to generalize to real-world manipulation task. For this reason, failure to detect or segment is not considered sufficient to declare a test run as failed. Instead, if the system fails to segment an object, the object is moved to a different position until the position at which the object can be sufficiently segmented is found. Nevertheless, the number of test runs when AffNet-DR failed to classify or segment will be provided along with the other full system results.

4.3.3 Procedure

The object set for this test consists of a mallet, a ladle, a spatula, a knife, a bowl, and a mug. The authors of this report will play the role of the receivers during the testing.

The following procedure was repeated for each test run and for both handover orientation methods:

1. A test object is placed fully in the view of the camera.
2. Start the program.
3. If AffNet-DR fails to produce a usable segmentation, move the object, and restart a current test run. If AffNet-DR correctly classifies and segments the test object, or it fails to classify the test object but provides sufficient segmentation, continue with the test run.
4. If the robot manages to deliver the test object from the worktable to the receiver, the test run is considered successful. If it fails to do so, note down the failure case.
5. Repeat the test procedure 10 times for each object of the testing object set.

For both the rule-based and observation-based methods, slight pertubations around goal end-effector pose are allowed. This is done as it may happen that the computed end-effector pose that achieves object handover pose as computed by the *handover location* and *handover orientation* modules is not feasible. However, slight pertubation of this pose may be feasible, which allows the robot to move there. For the rule-based method, the perturbation of the end-effector orientation around the axis of importance is allowed along with the perturbation of the end-effector position. For the end-effector pose of the observation-based method, only the position is allowed to be perturbated.

Following pertubations are allowed:

- Rule-based method
  - Pertubations of the x coordinate of the handover position in the range from -20 cm to 20 cm discretized into 5 bins.
  - Pertubations of the the handover orientation around the axis of importance, which means around the handle for objects with grasp affordance and around the vertical axis for objects with wide-grasp and contain affordances, discretized into 7 bins.
  - No pertubations were allowed for the z and y coordinates.
  - All together, 35 pertubations of the handover pose are allowed.
- Observation-based method
  - Pertubations of the x coordinate of the handover position in the range from -20 cm to 20 cm discretized into 6 bins.
  - Pertubations of the y coordinate of the handover pose position in 0 cm to 10 cm discretized into 6 bins.

- No pertubations were allowed for the z coordinate and the orientation.
- All together, 36 pertubations of the handover pose are allowed.

Overall, 120 test runs are performed, 60 test runs per method, with 10 test run for each object of the testing object. The results can be found in section 5.3.

5 - Results

This section presents the results of the tests as described in section 4.

5.1 Affordance segmentation

	Real-world data		Synthetic data	
	AffordanceNet [22]	**Baseline**	**AffNet-DA [29]**	**AffNet-DR (Ours)**
Grasp	**0.731**	0.482	0.473	**0.611**
Cut	**0.762**	0.575	0.599	**0.604**
Scoop	**0.793**	0.647	0.332	**0.639**
Contain	**0.833**	0.859	**0.83**	0.710
Pound	**0.836**	0.655	0.224	**0.804**
Support	**0.821**	0.519	0.541	**0.578**
W-grasp	0.814	**0.848**	**0.821**	0.785
Average	**0.799**	0.655	0.546	**0.676**

Table 5.1: Performance on the UMD dataset. Results in bold indicates best performance. Methods trained on real-world data and synthetic data are compared seperately.

5.2 Object handover orientation

Question	Random-based	Rule-based	Observations-based
How safe did you feel during the handover?	2	**1.83**	2.67
The objects were oriented in a way that was comfortable for me when grasping.	2.5	**2**	**2**
I would orient the objects similarly when performing handovers.	3.33	**2**	**2**
I agree with the way the objects were oriented.	3.5	**1.83**	**1.83**
How well would you say you understood the handover process?	1.5	**1.33**	1.67
How natural did you find the object orientations when you grasped the objects?	3.17	**1.83**	**1.83**
How appropriate did you find the object orientations when you grasped the objects?	3.5	**1.83**	**1.83**

Table 5.2: Results of the user study. Questions were evaluated on a 5-point Likert scale, with 1 corresponding to "Strong agree" and 5 corresponding to "Strong disagree". Bold indicate preference.

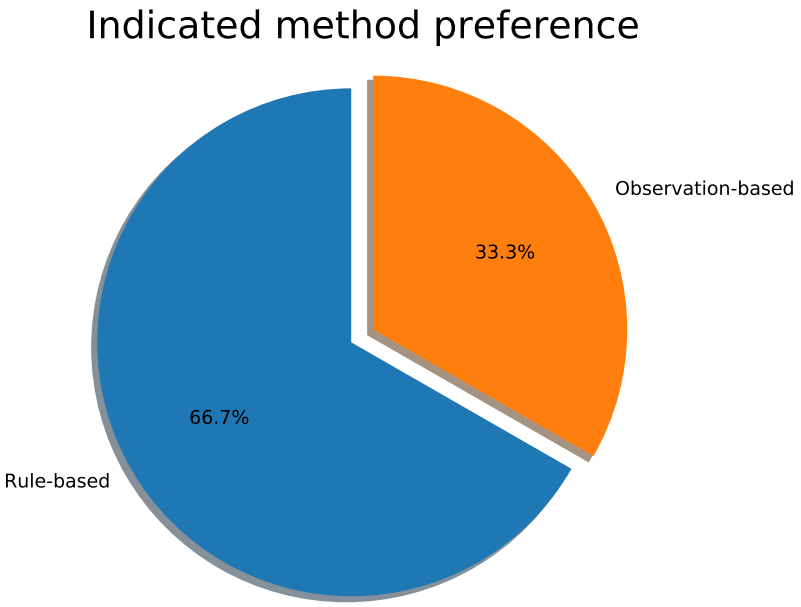


Figure 5.1: Method preferences as indicated by the participants after experiencing each method.

5.3 Full system test

5.3.1 Rule-based method

	Success	Failure to generate grasps	Failure to grasp	Failure to generate trajectory	Failure to handover
Mallet	90.00 %	0.00 %	10.00 %	0.00 %	0.00 %
Ladle	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Spatula	80.00 %	0.00 %	20.00 %	0.00 %	0.00 %
Knife	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Bowl	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Mug	80.00 %	0.00 %	20.00 %	0.00 %	0.00 %
Average	**91.67 %**	**0.00 %**	**8.33 %**	**0.00 %**	**0.00 %**

Table 5.3: Results of the full system test for the rule-based method.

5.3.2 Observation-based method

	Success	Failure to generate grasps	Failure to grasp	Failure to generate trajectory	Failure to handover
Mallet	90.00 %	0.00 %	10.00 %	0.00 %	0.00 %
Ladle	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Spatula	60.00 %	0.00 %	30.00 %	10.00 %	0.00 %
Knife	100.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Bowl	40.00 %	0.00 %	0.00 %	60.00 %	0.00 %
Mug	80.00 %	0.00 %	0.00 %	20.00 %	0.00 %
Average	**78.33 %**	**0.00 %**	**6.67 %**	**15.00 %**	**0.00 %**

Table 5.4: Results of the full system test for the observation-based method.

5.3.3 Failure to classify or segment

	No. of test runs with failure to classify	No. of test runs with failure to segment
Mallet	1	3
Ladle	0	3
Spatula	6	3
Knife	0	6
Bowl	1	2
Mug	1	0
Sum	**9/137**	**17/137**

Table 5.5: Number of test runs when AffNet-DR either failed to classify or segment the object correctly. The number is computed across both methods. The overall number of test runs across both methods, including successfull and restarted test runs, is 134.

5.3.4 Object handover orientations

Rule-based method								
Overall	**Object**	**Mallet**	**Ladle**	**Spatula**	**Knife**	**Bowl**	**Mug**	**Average**
	Success rate	90.00 %	100.00 %	80.00 %	100.00 %	100.00 %	80.00 %	**91.67 %**
Handover orientation	**Correct**	90.00 %	50.00 %	80.00 %	100.00 %	90.00 %	40.00 %	**75.00 %**
	Incorrect	0.00 %	50.00 %	0.00 %	0.00 %	10.00 %	40.00 %	**16.67 %**
	Pose est. error	0.00 %	40.00 %	0.00 %	0.00 %	10.00 %	30.00 %	**13.33 %**
	Error from grasp	0.00 %	10.00 %	0.00 %	0.00 %	0.00 %	10.00 %	**3.33 %**

Table 5.6: Occurrences and causes for incorrect handover object orientations observed during the testing of the rule-based method.

Observation-based method								
Overall	**Object**	**Mallet**	**Ladle**	**Spatula**	**Knife**	**Bowl**	**Mug**	**Average**
	Success rate	90.00 %	100.00 %	60.00 %	100.00 %	40.00 %	80.00 %	**78.33 %**
Handover orientation	**Correct**	70.00 %	70.00 %	30.00 %	90.00 %	0.00 %	40.00 %	**50.00 %**
	Incorrect	20.00 %	30.00 %	30.00 %	10.00 %	40.00 %	40.00 %	**28.33 %**
	Pose est. error	10.00 %	20.00 %	10.00 %	10.00 %	40.00 %	40.00 %	**21.67 %**
	Error from grasp	10.00 %	10.00 %	20.00 %	0.00 %	0.00 %	0.00 %	**6.67 %**

Table 5.7: Occurrences and causes for incorrect handover object orientations observed during the testing of the observation-based method.

5.3.5 Comparative results between the methods

	Rule-based method	**Observation-based method**
Success rate	**91.67 %**	78.33 %
Failure to grasp	8.33 %	**6.67 %**
Failure to generate trajectory	**0.00 %**	15.00 %
Correct handover orientation	**75.00 %**	50.00 %
Incorrect handover orientation	**16.67 %**	28.33 %

Table 5.8: Comparative results between the two proposed methods for object handover orientations. Bold indicate best performance.

6 - Discussion

6.1 Affordance segmentation

Table 5.1 shows that our AffNet-DR outperforms AffNet-DA, the current state-of-the-art method for affordance detection using synthetic data, by a meaningful margin. The requirement of a F_β^w score higher than 0.546, set up for the performance of the affordance detection network, is, therefore, exceeded.

This result shows that domain randomization is a promising approach to generating synthetic data for affordance segmentation tasks. Unlike AffNet-DA, our method does not require the neural network to train or adapt to real-world data. AffNet-DR generalizes to the real UMD dataset despite being trained on solely synthetic data. This is advantageous as synthetic data is considerably cheaper to obtain than annotated real-world data. Synthetic data generated with domain randomization can, therefore, alleviate some of the issues with current object affordance datasets, such as the limited size and scope.

AffNet-DR has succeeded in narrowing, if not closing, the sim2real gap, as AffNet-DR slightly outperforms the baseline trained solely on the real-world UMD dataset. Nevertheless, AffordanceNet [22] by Do et al. still outperforms both the baseline and AffNet-DR. All three neural networks are very similar architecture-wise, but both the baseline network and AffNet-DR have fewer parameters which might explain the performance drop compared to AffordanceNet. Future work should examine if better results can be obtained with synthetic data by experimenting with different network architectures.

6.2 User study of the object handover orientation

The user study confirmed the hypotheses presented in section 2.5. Firstly, the pie chart in Figure 5.1 shows that users, after experiencing all three methods of object handover orientations, prefer the rule-based and observation-based methods over the random-based method, with the majority being inclined towards the rule-based method. While some very unfavourable orientations were produced by the random-based method, it has also been observed that the random-based method would occasionally provide the user with the properly oriented object. Despite that, none of the participants preferred the random-based method.

Based on the collected results of the user study, presented in Table 5.2, users seem to find orientations produced by both the rule-based and observation-based methods to be equally comfortable, natural appropriate, and human-like. This confirms the second hypothesis of the user study i.e., both the rule-based and observation-based methods provide the user with appropriate object handover orientations.

This indicates that it is not necessary to learn object handover orientations from observing human-to-human handovers. Instead, proper object orientations can be achieved by considering the object affordances. This was further supported by the cluster analysis done in section 3.3.2, which found that the object handover orientations could be grouped into two groups, based on the object affordance and recorded object handover orientation. The first group consists of objects with a handle and, therefore, the grasp affordance. For this group, the handle should always be oriented towards the receiver. The second group of objects is the group with both the wide-grasp and contain affordances. The object of this group should be oriented such that they do not empty their contents, that is, normal to the ground plane.

Our results, which are in line with findings by Chan et al [11], indicate that one only needs to consider object affordances in order to compute proper handover orientations. To the best of the authors knowledge, this is the first work to compute handover orientations based on object affordances, obtained with affordance segmentation.

Finally, future work should concentrate more in depth on the handover location and the robot configuration during the handovers. Several of the user study participants reported that the object was handed over at an uncomfortable height. For this work, the handover location only considered the midpoint between the giver and the receiver with a fixed handover location height of 1.2 meter. It was also noted that in order to achieve the desired object handover orientations, the robot would occasionally reach awkward configurations, which lessened the perceived quality of the handover and reduced the receiver's perceived safety.

6.3 Full system test

The results of the full system test, presented in section 5.3, show the validity of the proposed robot-to-human handover system. The full system test done with the rule-based handover orientation method achieved an overall success rate of 91.67 % calculated on 60 trials, 10 trials per object, thereby, exceeding the requirement of a success rate of 75 %. The success rate was only negatively affected by the *grasp generator* module, which in 11.67 % of the cases generated a grasp that failed to obtain the item. Out of the 55 successful trials, a correct handover orientation was achieved in 45 of the cases. This means that a correct handover orientation was achieved by the system in 81.81 % of the successful handovers. The pose estimation method, which failed 8 times, was the main reason for the incorrect handover orientations. It mainly failed on the ladle object with 4 failed pose estimations. While the modified ICP pose estimation method performs well on most of

the objects, it fails on the ladle. This is most likely because the source point cloud is too dissimilar from the target point cloud. Choosing the correct source point cloud might be a challenging problem for a diverse set of objects. Instead, more reliable pose estimation techniques could be applied to this problem. Future work could take advantage of the synthetic affordance dataset to train a pose estimator.

The full system with the observation-based handover orientation method also met the requirement of a success rate of 75 %, albeit this method achieved a success rate of only 78.33 %. Unlike the rule-based method, the failure cases do not only originate from a failure to grasp the object but also to plan a trajectory that would orient the object correctly. The authors argue that this is due to harsher constraints on the object handover orientation for the observation-based method as compared to the rule-based method. The observation-based method only allows for a single handover orientation, whereas the rule-based method is allowed to rotate around the axis of importance, which makes it harder to plan a trajectory for the observation-based method.

Both the system with the rule-based and observation-based methods are capable of surpassing the requirement, however, based on the results, the *grasp generator* module could be improved, as it is the main cause of failures. Furthermore, the grasps generated by the *grasp generator* module also affects the ability to plan trajectories that would obtain a correct object handover orientation, as described in section 3.1.1. Future work could, therefore, investigate how to better grasp an object in such a way that it would make it more likely that the handover orientation is achieved.

While both methods meet the requirement, the rule-based method seems the most promising as it achieves a better performance than the observation-based method, as can be seen in Table 5.8. The observation-based method only outperforms the rule-based method in terms of the ability to grasp the object, but this is likely because the observation-based method did not try to perform the same number of grasps as the rule-based method. The system does not attempt a grasp unless a trajectory that would achieve the object handover orientation can be generated.

Full system testing shows that real-world robotic manipulation is achievable using the proposed AffNet-DR trained solely on synthetic data. The system relies on affordance segmentation, as from the segmented affordances, object orientation, position, and task-oriented grasps are computed. However, the affordance segmentation performed by AffNet-DR sometimes fails as seen in Table 5.5. AffNet-DR was successful in segmenting affordances in 120 cases out of 137, which is promising, but further improvement is still required.

6.4 Future work

The rule-based system showed itself capable of performing task-oriented handovers while also considering the appropriate object handover orientations. Here, the authors provide a list of possible future improvements for the proposed robot-to-human handover system.

- AffNet-DR achieves state-of-the-art performance for methods trained on synthetic data, but it still lacks behind systems that are trained on real-world data. The system could be improved by improving the synthetic data generator such that the system can also tackle object occlusions or object only in partial view. The architecture could also be experimented with in order to obtain better results on the dataset that is already generated.
- Handover orientations can not be considered in a vacuum. The user study showed that the perceived quality of the handover experience also depends on a handover location, a robot trajectory and a robot configuration. Future work should improve upon these aspects.
- In order to achieve better robot configurations and legible trajectories, a more advanced trajectory control could be implemented.
- Most of the failure cases stems from the physical system trying to grasp the object and failing. Another approach to grasping could, therefore, be investigated.
- The system does not consider the physical handover phase, where both the giver and the receiver is in contact with the object. The system goes to the handover pose and then waits for a pre-defined amount of time before realising the gripper. More sophisticated methods could be implemented.

7 - Conclusion

This report presents a robotic system capable of performing robot-to-human handovers. The system utilizes object affordance detection to compute task-oriented grasps as well as object handover orientations. The two main contributions of this report are in the field of object affordance detection and the computation of object handover orientations.

The neural network AffNet-DR is proposed as a novel method for applying synthetic data generated with domain randomization principles to object affordance detection. AffNet-DR is capable of performing affordance segmentation on real-world data despite being trained solely on synthetic data. This was confirmed by achieving a F_β^w score of 0.676 on the UMD affordance dataset, which consist of real-world data. The ability to perform real-world manipulation using the output of AffNet-DR was validated by the full system test which achieved a handover success rate of 91.67 %.

We show that object handover orientations can be computed solely from pixel-wise segmentation of affordances in RGB-D images, which, to the best of the authors' knowledge, we are the first to do. Furthermore, a conducted user study suggests that a set of rules based on object affordances is sufficient for generating appropriate object handover orientations.

The system was developed within the ROS framework and integrated on a robotic system consisting of a KUKA LBR iiwa 7 R800 collaborative robot, with an Intel RealSense D435i RGB-D sensor, a Hokuyo URG-04LX-UG01 laser scanner, and a Robotiq 3-finger gripper.

The developed system satisfies the problem statement and is capable of performing task-oriented robot-to-human handovers with proper object handover orientation using object affordance detection.

Bibliography

[1] Sebastian Hjorth and Dimitrios Chrysostomou. "Human–robot collaboration in industrial environments: A literature review on non-destructive disassembly". In: *Robotics and Computer-Integrated Manufacturing* 73 (2022), p. 102208. DOI: 10.1016/j.rcim.2021.102208.

[2] Albert Wilcox, Justin Kerr, Brijen Thananjeyan, Jeffrey Ichnowski, Minho Hwang, Samuel Paradis, Danyal Fer, and Ken Goldberg. "Learning to Localize, Grasp, and Hand Over Unmodified Surgical Needles". In: *arXiv preprint arXiv:2112.04071* (2021).

[3] Jonathan Bohren, Radu Bogdan Rusu, E Gil Jones, Eitan Marder-Eppstein, Caroline Pantofaru, Melonee Wise, Lorenz Mösenlechner, Wim Meeussen, and Stefan Holzer. "Towards autonomous robotic butlers: Lessons learned with the PR2". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 5568–5575.

[4] Natalie Sebanz, Harold Bekkering, and Günther Knoblich. "Joint action: bodies and minds moving together". In: *Trends in Cognitive Sciences* 10.2 (2006), pp. 70–76. ISSN: 1364-6613. DOI: https://doi.org/10.1016/j.tics.2005.12.009. URL: https://www.sciencedirect.com/science/article/pii/S1364661305003566.

[5] Patrick Rosenberger, Akansel Cosgun, Rhys Newbury, Jun Kwan, Valerio Ortenzi, Peter Corke, and Manfred Grafinger. "Object-independent human-to-robot handovers using real time robotic vision". In: *IEEE Robotics and Automation Letters* 6.1 (2020), pp. 17–23. DOI: 10.1109/LRA.2020.3026970.

[6] Wei Yang, Chris Paxton, Maya Cakmak, and Dieter Fox. "Human grasp classification for reactive human-to-robot handovers". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 11123–11130. DOI: 10.1109/IROS45743.2020.9341004.

[7] Paola Ardón, Maria E Cabrera, Eric Pairet, Ronald PA Petrick, Subramanian Ramamoorthy, Katrin S Lohan, and Maya Cakmak. "Affordance-aware handovers with human arm mobility constraints". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 3136–3143. DOI: 10.1109/LRA.2021.3062808.

[8] Valerio Ortenzi, Akansel Cosgun, Tommaso Pardi, Wesley Chan, Elizabeth Croft, and Dana Kulic. *Object Handovers: a Review for Robotics*. 2020. arXiv: 2007.12952 [cs.RO].

[9] Emrah Akin Sisbot and Rachid Alami. "A Human-Aware Manipulation Planner". In: *IEEE Transactions on Robotics* 28.5 (2012), pp. 1045–1057. DOI: 10.1109/TRO.2012.2196303.

[10] Wesley P. Chan, Matthew K.X.J. Pan, Elizabeth A. Croft, and Masayuki Inaba. "Characterization of handover orientations used by humans for efficient robot to human handovers". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 1–6. DOI: 10.1109/IROS.2015.7353106.

[11] Wesley P. Chan, Matthew K. X. J. Pan, Elizabeth A. Croft, and Masayuki Inaba. "An Affordance and Distance Minimization Based Method for Computing Object Orientations for Robot Human Handovers". In: *International Journal of Social Robotics* 12 (2020), pp. 143–162.

[12] Renaud Detry, Jeremie Papon, and Larry Matthies. "Task-oriented grasping with semantic and geometric scene understanding". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 3266–3273. DOI: 10.1109/IROS.2017.8206162.

[13] Maya Cakmak, Siddhartha S. Srinivasa, Min Kyung Lee, Jodi Forlizzi, and Sara Kiesler. "Human preferences for robot-human hand-over configurations". In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2011, pp. 1986–1993. DOI: 10.1109/IROS.2011.6094735.

[14] Jacopo Aleotti, Vincenzo Micelli, and Stefano Caselli. "Comfortable robot to human object hand-over". In: *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. 2012, pp. 771–776. DOI: 10.1109/ROMAN.2012.6343845.

[15] Jacopo Aleotti, Vincenzo Micelli, and Stefano Caselli. "An Affordance Sensitive System for Robot to Human Object Handover". In: *International Journal of Social Robotics* 6 (2014), pp. 653–666.

[16] Wesley P. Chan, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. "Determining proper grasp configurations for handovers through observation of object movement patterns and inter-object interactions during usage". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014, pp. 1355–1360. DOI: 10.1109/IROS.2014.6942733.

[17] SH Creem and DR Proffitt. "Grasping objects by their handles: a necessary interaction between cognition and action". In: *Journal of experimental psychology. Human perception and performance* 27.1 (2001), 218—228. ISSN: 0096-1523. DOI: 10.1037//0096-1523.27.1.218. URL: https://doi.org/10.1037//0096-1523.27.1.218.

[18] James J Gibson. "The theory of affordances". In: *Hilldale, USA* 1.2 (1977), pp. 67–82.

[19] Austin Myers, Ching L. Teo, Cornelia Fermüller, and Yiannis Aloimonos. "Affordance detection of tool parts from geometric features". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 1374–1381. DOI: 10.1109/ICRA.2015.7139369.

[20] Anh Nguyen, Dimitrios Kanoulas, Darwin G. Caldwell, and Nikos G. Tsagarakis. "Detecting object affordances with Convolutional Neural Networks". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 2765–2770. DOI: 10.1109/IROS.2016.7759429.

[21] Anh Nguyen, Dimitrios Kanoulas, Darwin G. Caldwell, and Nikos G. Tsagarakis. "Object-based affordances detection with Convolutional Neural Networks and dense Conditional Random Fields". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 5908–5915. DOI: 10.1109/IROS.2017.8206484.

[22] Thanh-Toan Do, Anh Nguyen, and Ian Reid. "AffordanceNet: An End-to-End Deep Learning Approach for Object Affordance Detection". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 5882–5889. DOI: 10.1109/ICRA.2018.8460902.

[23] Fu-Jen Chu, Ruinian Xu, Landan Seguin, and Patricio A. Vela. "Toward Affordance Detection and Ranking on Novel Objects for Real-World Robotic Manipulation". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4070–4077. DOI: 10.1109/LRA.2019.2930364.

[24] Lorenzo Jamone, Emre Ugur, Angelo Cangelosi, Luciano Fadiga, Alexandre Bernardino, Justus Piater, and José Santos-Victor. "Affordances in Psychology, Neuroscience, and Robotics: A Survey". In: *IEEE Transactions on Cognitive and Developmental Systems* 10.1 (2018), pp. 4–25. DOI: 10.1109/TCDS.2016.2594134.

[25] Philipp Zech, Simon Haller, Safoura Rezapour Lakani, Barry Ridge, Emre Ugur, and Justus Piater. "Computational models of affordance in robotics: a taxonomy and systematic classification". In: *Adaptive Behavior* 25.5 (2017), pp. 235–271. DOI: 10.1177/1059712317726357.

[26] Celso M de Melo, Antonio Torralba, Leonidas Guibas, James DiCarlo, Rama Chellappa, and Jessica Hodgins. "Next-generation deep learning based on simulators and synthetic data". In: *Trends in cognitive sciences* (2021). DOI: 10.1016/j.tics.2021.11.008.

[27] Jean-Baptiste Weibel, Timothy Patten, and Markus Vincze. "Addressing the Sim2Real Gap in Robotic 3-D Object Classification". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 407–413. DOI: 10.1109/LRA.2019.2959497.

[28] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. "Domain randomization for transferring deep neural networks from simulation to the real world". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 23–30. DOI: 10.1109/IROS.2017.8202133.

[29] Fu-Jen Chu, Ruinian Xu, and Patricio A. Vela. "Learning Affordance Segmentation for Real-World Robotic Manipulation via Synthetic Images". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1140–1147. DOI: 10.1109/LRA.2019.2894439.

[30] Aleksi Hämäläinen, Karol Arndt, Ali Ghadirzadeh, and Ville Kyrki. "Affordance Learning for End-to-End Visuomotor Robot Control". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019), pp. 1781–1788.

[31] Albert Daugbjerg Christensen and Daniel Lehotský. *Main GitHub repository of this project*. URL: https://github.com/daniellehot/ROB10.

[32] Albert D. Christensen, Daniel Lehotsky, and Marius W. Jørgensen. *Task-oriented handover using task-agnostic grasping and affordance segmentation*. 2021. URL: https://drive.google.com/file/d/1eq1AgkKP_CL5ySzMgT30E7mqOxWRRoq-/view?usp=sharing.

[33] Andrea Mason and Christine MacKenzie. "Grip forces when passing an object to a partner". In: *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale* 163 (June 2005), pp. 173–87. DOI: 10.1007/s00221-004-2157-x.

[34] Valerio Ortenzi, Francesca Cini, Tommaso Pardi, Naresh Marturi, Rustam Stolkin, Peter Corke, and Marco Controzzi. "The grasp strategy of a robot passer influences performance and quality of the robot-human object handover". In: *Frontiers in Robotics and AI* (2020), p. 138. DOI: 10.3389/frobt.2020.542406.

[35] Mia Kokic, Johannes A Stork, Joshua A Haustein, and Danica Kragic. "Affordance detection for task-specific grasping using deep learning". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE. 2017, pp. 91–98. DOI: 10.1109/HUMANOIDS.2017.8239542.

[36] Weiyu Liu, Angel Andres Daruna, and S. Chernova. "CAGE: Context-Aware Grasping Engine". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), pp. 2550–2556. DOI: 10.1109/ICRA40945.2020.9197289.

[37] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. "GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11441–11450. DOI: 10.1109/CVPR42600.2020.01146.

[38] Clint Hansen, Paula Arambel, Khalil Ben Mansour, Véronique Perdereau, and Frédéric Marin. "Human–Human Handover Tasks and How Distance and Object Mass Matter". eng. In: *Perceptual and motor skills* 124.1 (2017), pp. 182–199. ISSN: 0031-5125.

[39] Patrizia Basili, Markus Huber, Thomas Brandt, Sandra Hirche, and Stefan Glasauer. "Investigating Human-Human Approach and Hand-Over". In: *Human Centered Robot Systems, Cognition, Interaction, Technology*. 2009.

[40] Heramb Nemlekar, Dharini Dutia, and Zhi Li. "Object Transfer Point Estimation for Fluent Human-Robot Handovers". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 2627–2633. DOI: 10.1109/ICRA.2019.8794008.

[41] Halit Bener Suay and Emrah Akin Sisbot. "A position generation algorithm utilizing a biomechanical model for robot-human object handover". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 3776–3781. DOI: 10.1109/ICRA.2015.7139724.

[42] Maya Cakmak, Siddhartha S. Srinivasa, Min Kyung Lee, Sara Kiesler, and Jodi Forlizzi. "Using spatial and temporal contrast for fluent robot-human hand-overs". In: *2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2011, pp. 489–496. DOI: 10.1145/1957656.1957823.

[43] Matthew Ray and Timothy N. Welsh. "Response Selection During a Joint Action Task". In: *Journal of Motor Behavior* 43 (2011), pp. 329 –332.

[44] F. Cini, Valerio Ortenzi, Peter Corke, and Marco Controzzi. "On the choice of grasp type and location when handing over an object". In: *Science Robotics* 4 (2019).

[45] Markus Huber, Markus Rickert, Alois Knoll, Thomas Brandt, and Stefan Glasauer. "Human-robot interaction in handing-over tasks". In: *RO-MAN 2008 - The 17th IEEE International Symposium on Robot and Human Interactive Communication*. 2008, pp. 107–112. DOI: 10.1109/ROMAN.2008.4600651.

[46] Cordula Vesper, Stephen Butterfill, Günther Knoblich, and Natalie Sebanz. "A minimal architecture for joint action". In: *Neural Networks* 23.8-9 (2010), pp. 998–1003.

[47] AJung Moon, Daniel M Troniak, Brian Gleeson, Matthew KXJ Pan, Minhua Zheng, Benjamin A Blumer, Karon MacLean, and Elizabeth A Croft. "Meet me where i'm gazing: how shared attention gaze affects human-robot handover timing". In: *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. 2014, pp. 334–341.

[48] Jean-David Boucher, Ugo Pattacini, Amelie Lelong, Gerard Bailly, Frederic Elisei, Sascha Fagel, Peter F Dominey, and Jocelyne Ventre-Dominey. "I reach faster when I see you look: gaze effects in human–human and human–robot face-to-face cooperation". In: *Frontiers in neurorobotics* 6 (2012), p. 3.

[49] Mamoun Gharbi, Pierre-Vincent Paubel, Aurélie Clodic, Ophélie Carreras, Rachid Alami, and Jean-Marie Cellier. "Toward a better understanding of the communication cues involved in a human-robot object transfer". In: *2015 24th IEEE international symposium on robot and human interactive communication (RO-MAN)*. IEEE. 2015, pp. 319–324.

[50] Ran Margolin, Lihi Zelnik-Manor, and Ayellet Tal. "How to Evaluate Foreground Maps". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 248–255. DOI: 10.1109/CVPR.2014.39.

[51] Johann Sawatzky, Abhilash Srikantha, and Juergen Gall. "Weakly Supervised Affordance Detection". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5197–5206. DOI: 10.1109/CVPR.2017.552.

[52] Shengheng Deng, Xun Xu, Chaozheng Wu, Ke Chen, and Kui Jia. *3D AffordanceNet: A Benchmark for Visual Object Affordance Understanding*. 2021. arXiv: 2103.16397 [cs.CV].

[53] Luis Montesano, Manuel Lopes, Alexandre Bernardino, and JosÉ Santos-Victor. "Learning Object Affordances: From Sensory–Motor Coordination to Imitation". In: *IEEE Transactions on Robotics* 24.1 (2008), pp. 15–26. DOI: 10.1109/TRO.2007.914848.

[54] Huaqing Min, Chang'an Yi, Ronghua Luo, Jinhui Zhu, and Sheng Bi. "Affordance Research in Developmental Robotics: A Survey". In: *IEEE Transactions on Cognitive and Developmental Systems* 8.4 (2016), pp. 237–255. DOI: 10.1109/TCDS.2016.2614992.

[55] Natsuki Yamanobe, Weiwei Wan, Ixchel G Ramirez-Alpizar, Damien Petit, Tokuo Tsuji, Shuichi Akizuki, Manabu Hashimoto, Kazuyuki Nagata, and Kensuke Harada. "A brief review of affordance in robotic manipulation research". In: *Advanced Robotics* 31.19-20 (2017), pp. 1086–1101. DOI: 10.1080/01691864.2017.1394912.

[56] Abdalwhab Abdalwhab and Huaping Liu. "Feature Fusion One-Stage Visual Affordance Detector". In: *2020 7th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS)*. 2020, pp. 102–107. DOI: 10.1109/ICCSS52145.2020.9336910.

[57] Xue Zhao, Yang Cao, and Yu Kang. "Object affordance detection with relationship-aware network". In: *Neural Computing and Applications* 32.18 (2020), pp. 14321–14333. DOI: 10.1007/s00521-019-04336-0.

[58] Chau Nguyen Duc Minh, Syed Zulqarnain Gilani, Syed Mohammed Shamsul Islam, and David Suter. "Learning Affordance Segmentation: An Investigative Study". In: *2020 Digital Image Computing: Techniques and Applications (DICTA)*. 2020, pp. 1–8. DOI: 10.1109/DICTA51227.2020.9363390.

[59] Kun Qian, Xingshuo Jing, Yanhui Duan, Bo Zhou, Fang Fang, Jing Xia, and Xudong Ma. "Grasp pose detection with affordance-based task constraint learning in single-view point clouds". In: *Journal of Intelligent & Robotic Systems* 100.1 (2020), pp. 145–163. DOI: 10.1007/s10846-020-01202-3.

[60] Ruinian Xu, Fu-Jen Chu, Chao Tang, Weiyu Liu, and Patricio A. Vela. "An Affordance Keypoint Detection Network for Robot Manipulation". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 2870–2877. DOI: 10.1109/LRA.2021.3062560.

[61] Qipeng Gu, Jianhua Su, and Lei Yuan. "Visual affordance detection using an efficient attention convolutional neural network". In: *Neurocomputing* 440 (2021), pp. 36–44. DOI: 10.1016/j.neucom.2021.01.018.

[62] Chi-Yi Tsai, Han-Po Lin, and Yu-Chen Chiu. "An ESP-Based Lightweight Model for Joint Object Detection and Affordance Segmentation". In: *2021 6th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*. 2021, pp. 89–93. DOI: 10.1109/ACIRS52449.2021.9519343.

[63] Edoardo Ragusa, Christian Gianoglio, Strahinja Dosen, and Paolo Gastaldo. "Hardware-Aware Affordance Detection for Application in Portable Embedded Systems". In: *IEEE Access* 9 (2021), pp. 123178–123193. DOI: 10.1109/ACCESS.2021.3109733.

[64] Congcong Yin, Qiuju Zhang, and Wenqiang Ren. "A New Semantic Edge Aware Network for Object Affordance Detection". In: *Journal of Intelligent & Robotic Systems* 104.1 (2022), pp. 1–16. DOI: 10.1007/s10846-021-01525-9.

[65] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

[66] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[67] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision meets Robotics: The KITTI Dataset". In: *International Journal of Robotics Research (IJRR)* (2013).

[68] Johann Sawatzky, Martin Garbade, and Juergen Gall. "Ex paucis plura: learning affordance segmentation from very few examples". In: *German Conference on Pattern Recognition*. Springer. 2018, pp. 169–184. DOI: 10.1007/978-3-030-12939-2_13.

[69] Juergen Gall and Johann Sawatzky. "Adaptive Binarization for Weakly Supervised Affordance Segmentation". In: *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2017, pp. 1383–1391. DOI: 10.1109/ICCVW.2017.164.

[70] Tommaso Apicella, Andrea Cavallaro, Riccardo Berta, Paolo Gastaldo, Francesco Bellotti, and Edoardo Ragusa. "An Affordance Detection Pipeline for Resource-Constrained Devices". In: *2021 28th IEEE International Conference on Electronics, Circuits, and Systems (ICECS)*. 2021, pp. 1–6. DOI: 10.1109/ICECS53924.2021.9665447.

[71] Fu-Jen Chu, Ruinian Xu, and Patricio A. Vela. "Detecting Robotic Affordances on Novel Objects with Regional Attention and Attributes". In: *ArXiv* abs/1909.05770 (2019).

[72] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. "Sim2Real in robotics and automation: Applications and challenges". In: *IEEE transactions on automation science and engineering* 18.2 (2021), pp. 398–400.

[73] Haziq Razali and Yiannis Demiris. "Multitask Variational Autoencoding of Human-to-Human Object Handover". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021, pp. 7315–7320. DOI: 10.1109/IROS51168.2021.9636221.

[74] Christoph Hennersperger, Bernhard Fuerst, Salvatore Virga, Oliver Zettinig, Benjamin Frisch, Thomas Neff, and Nassir Navab. "Towards MRI-based autonomous robotic US acquisitions: a first feasibility study". In: *IEEE transactions on medical imaging* 36.2 (2017), pp. 538–548.

[75] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. "The YCB object and Model set: Towards common benchmarks for manipulation research". In: *2015 International Conference on Advanced Robotics (ICAR)*. 2015, pp. 510–517. DOI: 10.1109/ICAR.2015.7251504.

[76] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. "Training Deep Networks with Synthetic Data: Bridging the Reality Gap by Domain Randomization". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2018, pp. 1082–10828. DOI: 10.1109/CVPRW.2018.00143.

[77] Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. "BlenSor: Blender sensor simulation toolbox". In: *International Symposium on Visual Computing*. Springer. 2011, pp. 199–208.

[78] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. "A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM". In: *2014 IEEE international conference on Robotics and automation (ICRA)*. IEEE. 2014, pp. 1524–1531.

[79] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. *Mask R-CNN*. 2018. arXiv: 1703.06870 [cs.CV].

[80] Ruolin Ye, Wenqiang Xu, Zhendong Xue, Tutian Tang, Yanfeng Wang, and Cewu Lu. "H2O: A Benchmark for Visual Human-human Object Handover Analysis". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 15742–15751.

[81] Richard I. Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. "Rotation Averaging". In: *International Journal of Computer Vision* 103 (2012), pp. 267–305.

[82] W.D. Curtis, A.L. Janin, and K. Zikan. "A note on averaging rotations". In: *Proceedings of IEEE Virtual Reality Annual International Symposium*. 1993, pp. 377–385. DOI: 10.1109/VRAIS.1993.380755.

[83] Inna Sharf, Alon Wolf, and M.B. Rubin. "Arithmetic and geometric solutions for average rigid-body rotation". In: *Mechanism and Machine Theory* 45.9 (2010), pp. 1239–1251. ISSN: 0094-114X. DOI: https://doi.org/10.1016/j.mechmachtheory.2010.05.002. URL: https://www.sciencedirect.com/science/article/pii/S0094114X10000790.

[84] Shuichi Akizuki and Yoshimitsu Aoki. "Pose alignment for different objects using affordance cues". In: *2018 International Workshop on Advanced Image Technology (IWAIT)*. 2018, pp. 1–3. DOI: 10.1109/IWAIT.2018.8369759.

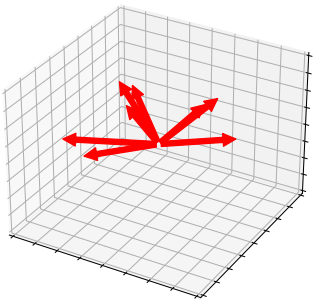
[85] P.J. Besl and Neil D. McKay. "A method for registration of 3-D shapes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 239–256. DOI: 10.1109/34.121791.

[86] Emin Babakus and W. Glynn Mangold. "Adapting the SERVQUAL scale to hospital services: an empirical investigation". In: *Health Services Research* 26 (1992), pp. 767–786. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1069855/.

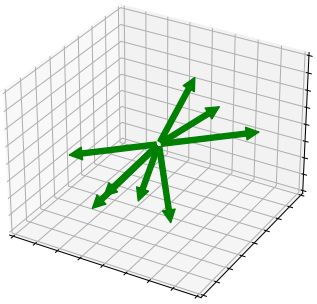
[87] Seung Youn Chyung, Katherine Roberts, Ieva Swanson, and Andrea Hankinson. "Evidence-Based Survey Design: The Use of a Midpoint on the Likert Scale". In: *Performance Improvement* 56 (2017), pp. 15–27. DOI: 10.1002. URL: https://doi.org/10.1002/pfi.21727.

A - Analysis of the axes orientations

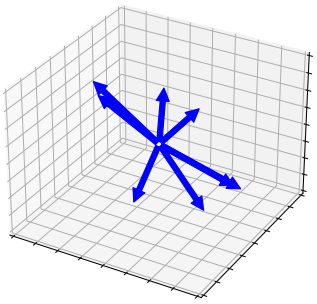
bottle
Oriented x axes



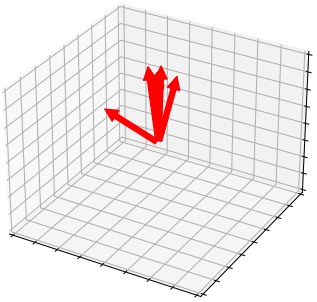
bottle
Oriented y axes



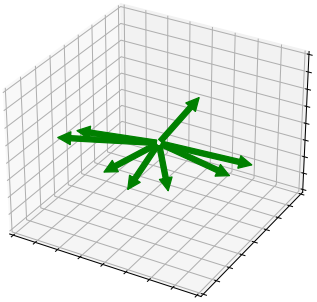
bottle
Oriented z axes



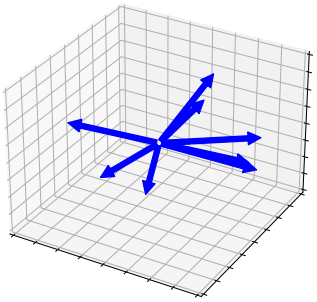
bowl
Oriented x axes



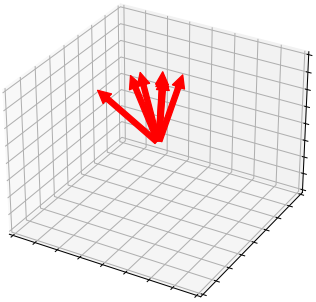
bowl
Oriented y axes



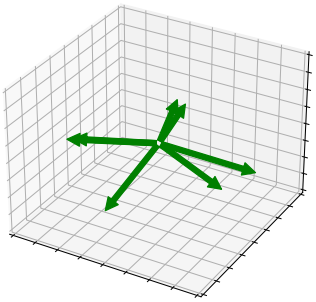
bowl
Oriented z axes



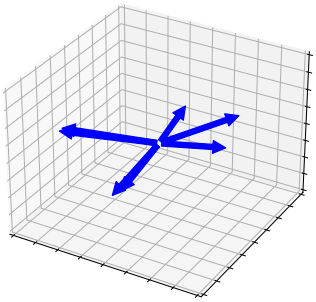
cup
Oriented x axes



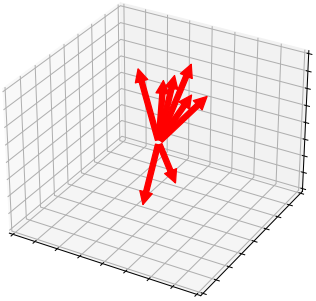
cup
Oriented y axes



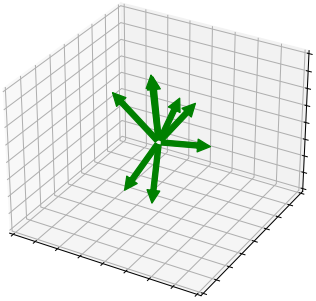
cup
Oriented z axes



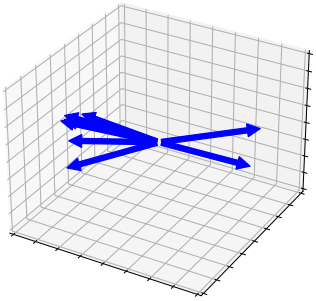
hammer
Oriented x axes



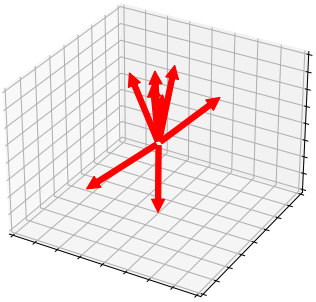
hammer
Oriented y axes



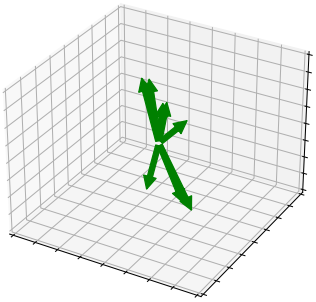
hammer
Oriented z axes



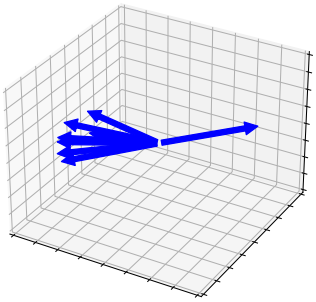
knife
Oriented x axes



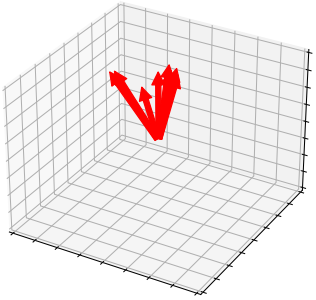
knife
Oriented y axes



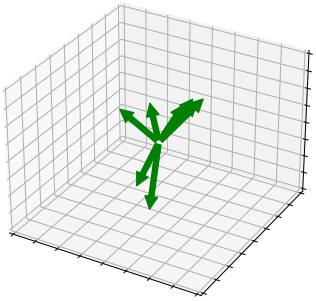
knife
Oriented z axes



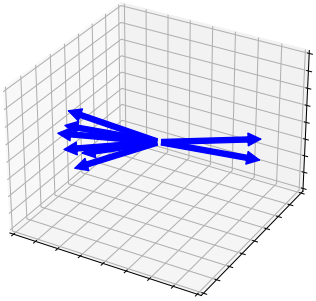
ladle
Oriented x axes



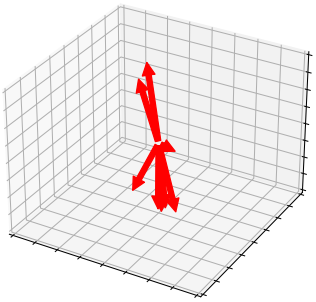
ladle
Oriented y axes



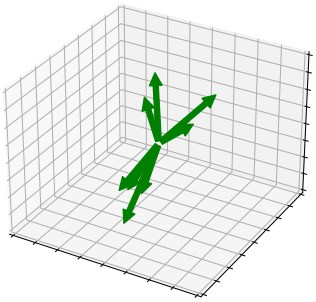
ladle
Oriented z axes



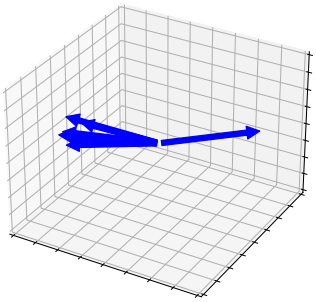
mallet
Oriented x axes



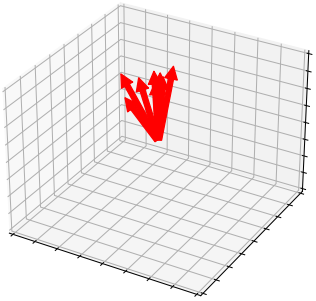
mallet
Oriented y axes



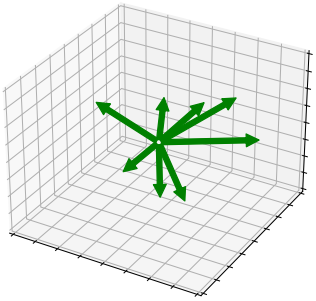
mallet
Oriented z axes



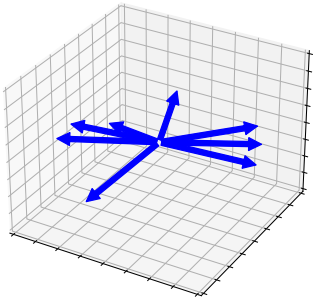
mug
Oriented x axes



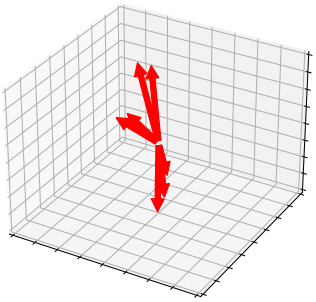
mug
Oriented y axes



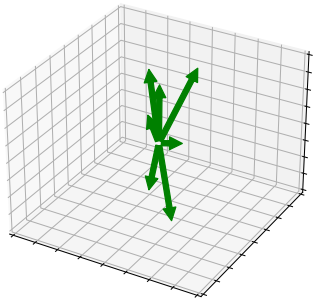
mug
Oriented z axes



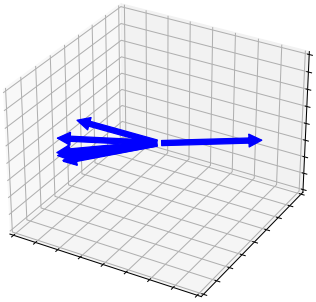
scissors
Oriented x axes



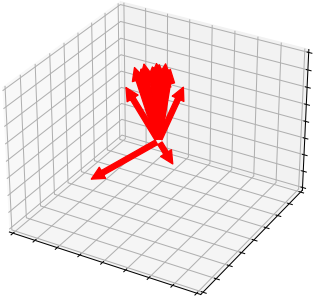
scissors
Oriented y axes



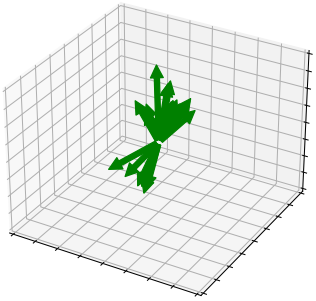
scissors
Oriented z axes



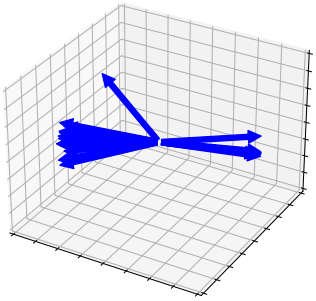
scoop
Oriented x axes



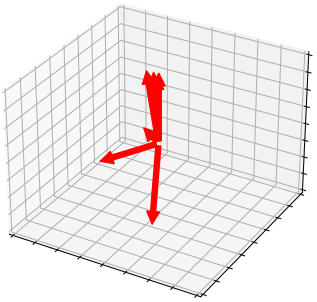
scoop
Oriented y axes



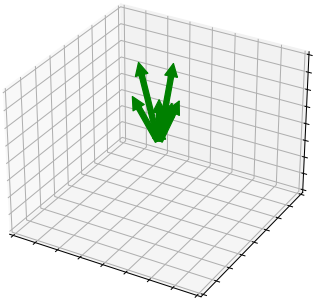
scoop
Oriented z axes



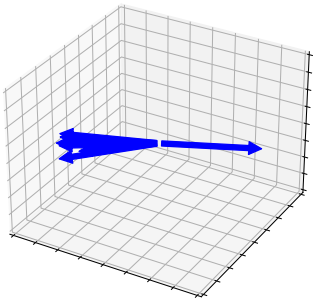
spatula
Oriented x axes



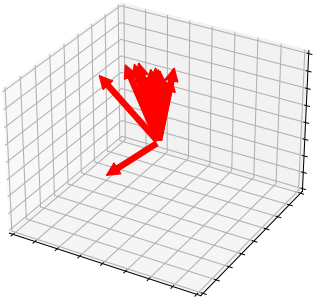
spatula
Oriented y axes



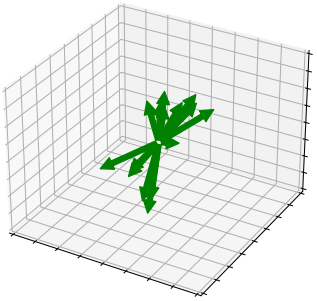
spatula
Oriented z axes



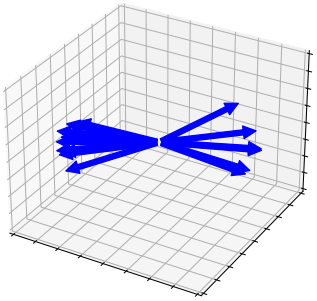
spoon
Oriented x axes



spoon
Oriented y axes



spoon
Oriented z axes



B - Links of interest

The main GitHub repository https://github.com/daniellehot/ROB10
Synthetic data generator https://github.com/HuchieWuchie/affordanceSynthetic
Synthetic dataset https://github.com/HuchieWuchie/affnetDRROS
The analysis of the human-to-human handover dataset https://github.com/daniellehot/handover_orientation_analysis
Demo video https://github.com/daniellehot/ROB10/blob/main/demo_video.mp4