Active Vision for Scene Understanding

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Markus Grotz

aus Freiburg

Tag der mündlichen Prüfung:	25. Juni 2020
Erster Gutachter:	Prof. Dr.-Ing. Tamim Asfour
Zweite Gutachterin:	Prof. Dr. rer. nat. Maren Bennewitz

Deutsche Zusammenfassung

Die visuelle Wahrnehmung ist eine der wichtigsten Informationsquellen sowohl für Mensch als auch für Roboter. Eine besondere Herausforderung dabei liegt in der Erfassung und Interpretation komplexer unstrukturierter Szenen, besonders dann, wenn Objekte verdeckt sind und die Szene aus einem einzigen Blickwinkel nicht vollständig erfasst werden kann.

Ein wesentliches Problem bei der visuellen Wahrnehmung ist daher die Auswahl der Blickrichtung. Diese kann sowohl durch Augen- oder Kamerabewegung, als auch durch Wechsel der Körper- oder Roboterposition geändert werden. Methoden zur aktiven visuellen Wahrnehmung (*Active Vision*) steuern selektiv und zielgerichtet die Blickrichtung, entweder zur Unterstützung einer aktuellen Aufgabe (*task-oriented*) oder als Reaktion auf einen Reiz (*stimulus-driven*). Die Auswahl der Blickrichtung in Abhängigkeit von der aktuellen Aufgabe ist der zentrale Gegenstand dieser Arbeit. Verdeckungsprobleme oder Inkonsistenzen in einem Umgebungsmodell sollen durch Änderung der Blickrichtung aufgelöst werden. Die Umgebung soll dabei mit einer minimalen Anzahl von Blickrichtungen, die für ein vollständiges Umgebungsmodell notwendig sind, erfasst werden. Die entwickelten Methoden in dieser Arbeit zur aktiven visuellen Wahrnehmung werden zudem durch eine Blickstabilisierung ergänzt, um eine zuverlässige Wahrnehmung während Roboter- oder Blickrichtungsbewegungen zu ermöglichen.

Insgesamt lassen sich die Beiträge der Arbeit in die folgenden drei Themengebiete unterteilen: (1) Semantisches Umgebungsmodell, (2) Aktive visuelle Wahrnehmung und (3) Blickstabilisierung. Im ersten Teil, dem semantischen Umgebungsmodell, wird aus der aktuellen Blickrichtung zunächst ein Umgebungsmodell bestehend aus geometrischen Primitiven erstellt und mit semantischer Information angereichert. Diese semantische Information umfasst Interaktionsmöglichkeiten des Roboters mit Objekten, sowie Relationen zwischen geometrischen Primitiven der Szene. Zu den Relationen gehören Nachbarschaftsrelationen, sowie physikalisch plausible Stützrelationen (*Support Relations*). Eine Stützrelation existiert zwischen zwei geometrischen Primitiven A und B, wenn das Entfernen von A dazu führt, dass B seinen bewegungslosen Zustand verliert, d.h. A stützt B. Basierend auf dem resultierenden semantischen Umgebungsmodell wird eine neue Blickrichtungen so gewählt, dass fehlende und aufgabenrelevante Informationen vervollständigt werden. Die Blickstabilisierung unterstützt dabei die visuelle Wahrnehmung.

Die aktuelle Blickrichtung wird mit einem RGB-D Sensor abgetastet. Aus der daraus resultierenden Punktwolke werden mit einem RANSAC-basierten Ansatz geometrische Primitive gefunden. Die geometrischen Primitive umfassen Ebenen, Zylinder oder Quader, welche Objekte abstrahieren. Das resultierende geometrische Modell wird abschließend mit semantischer Information angereichert, die aus (i) Interaktionsmöglichkeiten mit diesen geometrischen Primitiven, sowie (ii) Relationen zwischen diesen Primitiven besteht. Hier sind insbesondere die physikalischen Relationen zwischen Elementen des Umgebungsmodells wichtig. Dadurch entsteht ein semantisches Modell, das es erlaubt, mögliche Aktionen zu identifizieren. Mit den physikalischen Relationen können diese dann vor Ausführung der Aktion auf Plausibilität überprüft werden.

Basierend auf dem erstellten semantischen Modell, welches die Interaktionsmöglichkeiten und die Relationen zwischen den geometrischen Primitiven umfasst, wird die Blickrichtung bestimmt, die notwendig ist um aufgabenrelevante Informationen zu vervollständigen. Die extrahierte Information aus der nächsten Blickrichtung wird anschließend in das bestehende Modell der Szene integriert. Aus den semantischen Relationen zwischen den geometrischen Primitiven werden dann Salienzkarten erstellt, die der Identifikation und Bewertung möglicher Kandidaten für eine neue Blickrichtung dienen. Der Salizenwert, ein Maß für das „Hervorstehen“ eines Umgebungselements, erlaubt es, die Anzahl der möglichen Blickrichtungskandidaten zu filtern und somit die zeit- und rechenintensive Untersuchung von Blickrichtungskandidaten zu reduzieren. Die Evaluation der Blickrichtungskandidaten erfolgt über eine Kostenfunktion, welche den geschätzten Informationsgewinn sowie die zurückgelegte Distanz des Roboters berücksichtigt. Noch nicht explorierte Regionen werden bevorzugt exploriert. Dadurch wird die Anzahl der benötigten Blickrichtungen minimiert.

Sowohl beim Blickrichtungswechsel, als auch bei der Ausführung von mobilen Manipulationsaufgaben wird die visuelle Wahrnehmung beeinträchtigt. Daher ist es notwendig, Methoden zur Blickstabilisierung (*Gaze Stabilization*) zu entwickeln und in die gesamte Architektur der aktiven visuellen Wahrnehmung zu integrieren. Die Blickstabilisierung erfolgt dabei, angelehnt an menschliche Methoden zur Blickstabilisierung, über den Vestibulo-Ocular Reflex, den Optokinetischen Reflex und über ein internes Modell des Roboters. Die Blickstabilisierung über das interne Modell des Roboters verwendet dabei Methoden der inversen Kinematik, um eine vorgegebene Blickrichtung stabil zu halten.

Die vorliegende Arbeit leistet einen Beitrag zur aktiven visuellen Wahrnehmung für humanoide Roboter. Es wird ein semantisches Modell der Szene erstellt, welches durch sukzessive Änderung der Blickrichtung des Roboters erweitert wird, um Manipulations- und Interaktionsmöglichkeiten mit Objekten der Szene zu explorieren. Weiterhin wird ein biologisch motivierter Ansatz für die Blickstabilisierung während Roboterbewegungen oder Blickrichtungswechseln vorgestellt.

Abstract

Visual perception is one of the most important sources of information for both humans and robots. A particular challenge is the acquisition and interpretation of complex unstructured scenes, especially when objects are hidden and the scene cannot be captured from a single view.

A major issue in visual perception is therefore the selection of the gaze. This can be changed by eye or camera movement as well as by changing the body or robot position. Methods for active vision selectively and purposefully control the gaze, either to support a task (*task-oriented*) or to respond to a stimulus (*stimulus-driven*). The selection of the gaze depending on the current task is the central subject of this work. Problems of occlusion or inconsistency in an environment model shall be solved by changing the direction of gaze. The environment should be visually captured with a minimum number of movements, which are necessary for a complete environment model. The developed methods for active vision will be complemented by a gaze stabilization to enable a reliable perception during robot or gaze direction movements.

Altogether, the contributions can be divided into the following three topics: (1) Semantic scene representation, (2) Active vision, and (3) Gaze stabilization. In the first part, the semantic scene representation, an environment model consisting of geometric primitives is created from the current view aligned with the gaze direction and enriched with semantic information. This semantic information includes interaction possibilities of the robot with objects, as well as relations among geometric primitives of the scene. The relations include neighborhood relations as well as physically plausible support relations. A supporting relation exists among two geometric primitives A and B if the removal of A leads to B losing its motionless state, i.e. A supports B. Based on the resulting semantic environment model, the next gaze direction is chosen to complete missing and task-relevant information. The gaze stabilization supports the visual perception. The current view is captured with an RGB-D sensor. From the resulting point cloud geometric primitives are fitted using a

RANSAC-based approach. The geometric primitives include planes, cylinders or cuboids, which abstract objects. The resulting geometric model is finally enriched with semantic information, which consists of (i) interaction possibilities with these geometric primitives, as well as (ii) relations between these primitives.

The physical relations among elements of the environment model are especially important here. This results in a semantic model that allows to identify possible actions. With the physical relations these can then be checked for plausibility before the action is executed. Based on the created semantic model, which includes the interaction possibilities and the relations among the geometric primitives, the next-best-view is determined, which is necessary to complete task-relevant information. The extracted information from the next-best-view is then integrated into the existing model of the scene. The semantic relations among the geometric primitives are then used to create salience maps, which serve to identify and evaluate possible candidates for a new viewing direction. The salience value, a measure of the " interestingness " of an object, allows to filter the number of possible views and thus reduces the time and computationally intensive examination of the viewing direction candidates. The evaluation of the viewing direction candidates is carried out via a cost function, which takes into account the estimated information gain and the distance travelled by the robot. Not yet explored regions are preferred. This minimizes the number of required views.

Visual perception is impaired both when changing the gaze direction and when performing mobile manipulation tasks. Therefore, it is necessary to develop methods for gaze stabilization and to integrate them into the overall architecture of the active vision system. The gaze stabilization is based on human gaze stabilization methods, the vestibulo-ocular reflex, the optokinetic reflex and an internal model of the robot. The gaze stabilization via the internal model of the robot uses methods of inverse kinematics to keep a given gaze direction stable.

The present work contributes to active vision for humanoid robots. A semantic model of the scene is created, which is extended by successively changing the robot's view in order to explore possibilities of manipulation and interaction with objects of the scene. Furthermore, a biologically motivated approach for gaze stabilization during robot movements or changes of gaze direction is presented.

Contents

1. Introduction

A *humanoid robot*[1] is a robot that resembles a human. Typically, robots are tailored to a specific task, e. g., monotonous assembly tasks, and do not generalize well. What makes humanoid robots unique is that they are designed to perform human tasks in a variety of different areas, especially in human engineered environments (Kemp et al., 2008). This includes, but is not limited to, collaborative tasks in highly unstructured industrial environments (Asfour et al., 2018), daily kitchen activities, or personal assistance in household environments (Asfour et al., 2006). Other applications include emergency situations (Spenko et al., 2018) like rescue operations in destroyed power plants (Tsagarakis et al., 2017), or industrial disaster challenges (Radford et al., 2015).

Despite recent and enormous advances in humanoid robotics (Hoffman et al., 2019), (Asfour et al., 2019b), (Kaneko et al., 2019), and (Radford et al., 2015), the desire for autonomous humanoid robots performing non-trivial tasks and interacting smoothly with the environment is not yet satisfied. A major challenge to increase the autonomy of humanoid robots is the automatic visual perception of unstructured and cluttered environments, where scene elements and relevant objects are hidden in the current view.

1.1. Motivation and Problem Statement

Without doubt, visual perception is among the most powerful sense, for both humans and humanoid robots. Visual perception allows to intuitively interpret the current scene and further to infer an understanding of the scene, describing objects and their relations. In general, similar to humans, robots have to deal with partial information from a single view due to occlusions as well as limited sensor data. Active vision methods (Aloimonos et al., 1988; Bajcsy, 1988)

[1] The etymology of the word *humanoid* is a hybrid of the Latin word *humanus* and the Greek suffix *-oid* meaning *resembling*.

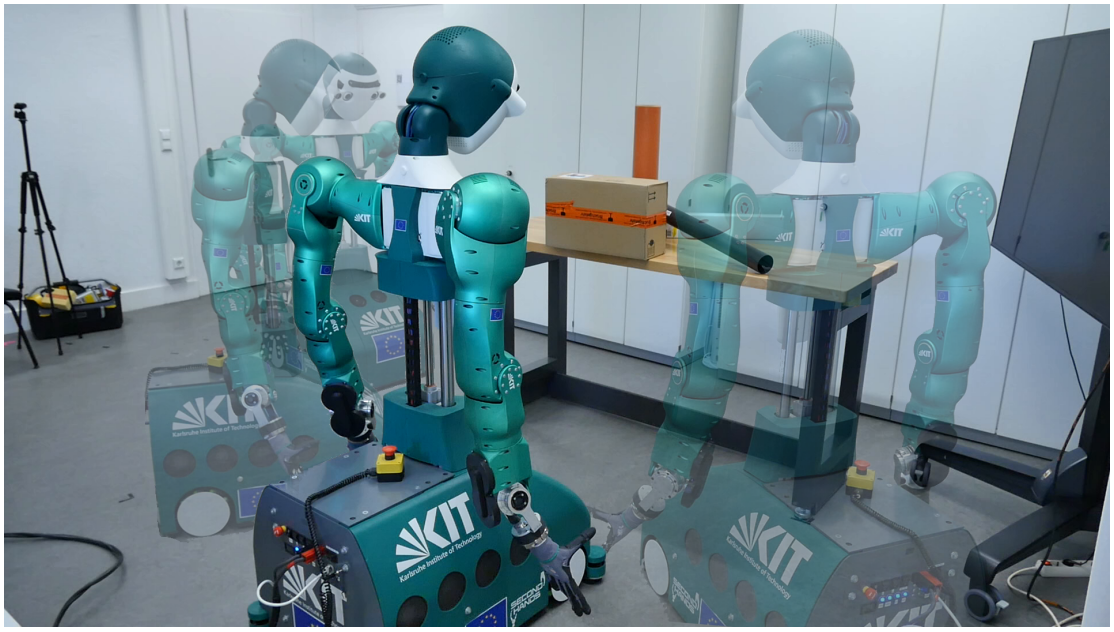


Figure 1.1.: ARMAR-6 (Asfour et al., 2019b) in front of a cluttered table-top scenario. In order to generate a complete scene model, the robot has to change the view, i. e., the gaze direction and the platform position. Here, the question arises which view the robot should choose. Possible views not yet visited by the robot are illustrated transparently.

selectively and purposefully control the gaze direction and the robot's position, either to support a current task (task-oriented) or to react to a stimulus (stimulus-driven). Visual perception is *active* in humans (Findlay and Gilchrist, 2003) and thus should be *active* for humanoid robots as well.

Figure 1.1 depicts an example of a cluttered scenario and the issue that arises with occlusion because the relevant part of the table-top scenario is hidden from the current view. Imagine the robot wants to interact with the scene and grasp the black pipe from the table-top. Since the scene is unknown to the robot, the robot first needs to autonomously create a scene model and infer an understanding. Due to the fact that relevant parts are hidden from the current view, it is impossible for the robot to create a complete semantic scene model from a single view only. In this example, transparent robot silhouettes are overlaid over the image to indicate possible robot views. These views are not yet visited by the robot and thus offer the potential to provide additional information. Selecting a suitable view for the robot allows to perceive relevant parts of the scene and finally to reason over them. Moreover, when moving the robot induces noise to the camera images.

Ultimately, for the automatic perception of unknown environments, robots require the following abilities. First, robots need to recognize which objects are present in the scene. This is done by mapping the sensor data from the current view to a meaningful representation and then to identify objects. Second, possible actions associated with the objects need to be derived. For example, whether an object is graspable at all, and if so, which grasp should the robot select. Third, relations among the objects have to be inferred for further reasoning. This includes, for example, inference of stability and support relations between objects that are in contact. This knowledge allows to understand the effects of manipulation actions. Fourth, robots need to change their gaze direction and their position purposefully for the automatic perception. Changing the gaze additionally induces noise into the perception which needs to be compensated. And finally, the extracted information needs to be available as soon as possible. This aspect covers all previous steps equally.

Visual perception is not only limited to geometric information. In addition to geometric information, semantic knowledge, for example, allows to interpret and understand the scene. Especially in cluttered and unstructured scenes, semantic and support knowledge is central for safe action execution. Due to occlusions, however, support relations among objects and scene elements cannot be reliably inferred from a single view only.
Overall, the major challenges for a humanoid robot to perceive a scene autonomously can be summarized as follows.

(1) **Incomplete Knowledge**: The scene is completely unknown or only partially known to the robot.

(2) **Occlusion**: Relevant parts of the scene are invisible or only partially visible in the current view. This is a special case of the first challenge.

(3) **Visual Sensor Constraints**: The sensor has a limited field of view, and sensor measurements are incomplete and noisy.

(4) **Self-induced noise**: The robot induces noise to the visual perception by moving the head and the body.

(5) **Time Constraints**: Relevant information needs to be available as soon as possible.

Similar challenges for view planning of object modeling tasks have been presented in Vasquez-Gomez et al. (2014). Indeed, the list of constraints is not complete, and depending on the current task or the scenario, other issues can

be more prominent. The subsequent section will answer how an active approach addresses these challenges for perceiving a scene autonomously.

1.2. Active Scene Understanding

In humans, visual perception is *active*. Humans constantly change their gaze to attend regions of interest. Thereby, humans can perceive a scene seamlessly. For example, a human would simply take a step to the side and look behind the occluding object depicted in the table-top scenario of Figure 1.1 to validate the unconscious inference of physical support among the objects. An *active* approach for visual perception addresses the challenges (1) - (5) stated in the previous section by changing the robot's position and gaze direction. In order to do this autonomously, the question arises, which among the endless possibilities of views is the best. This issue is addressed by planning the Next-Best-View (NBV). Given the described challenges for a humanoid that arise when perceiving the scene, planning the NBV is subject to many constraints. For example, planning all the required views in advance is not possible because the scene is unknown or changing. In addition, an exhaustive search would consume too much time. Hence, the view of the robot needs to be selective and include task relevant information. By allowing the robot to selectively change the current view, occlusion can be mitigated by shifting the position and observing the previously occluded region. Furthermore, a robot can *actively* search for and discover possible actions in the scene.
This thesis presents an active vision system that mitigates occlusion and explores the scene for object support relations. With respect to the previously raised issues, the goal of the presented *active vision* system is to:

(1) **Explore** the scene for missing and relevant information.

(2) **Mitigate** the effect of occlusions.

(3) **Resolve** inconsistencies or ambiguities in the extracted scene model and **validate** uncertainties of the support relations and actions.

(4) **Stabilize** the gaze and compensate for self-induced motion.

(5) **Minimize** the number of views necessary to complete the current task.

Overall, a humanoid robot's perception should be *active*, similar to visual perception in humans.

1.3. Contributions of the Thesis

This thesis presents a novel approach for humanoid robots to create a semantic scene model autonomously. Such a model is constructed given the sensor data from the current view. Next, the thesis presents an active vision method that iteratively determines the NBV and thus updates the scene model. Finally, a gaze stabilization controller is integrated to allow for perception during motion. Overall, the goals (1) to (5) of this thesis are formulated to answer the following questions:

(i) What objects are present in the scene?

(ii) What are the relations among the objects?

(iii) What is the NBV to improve the current perception?

(iv) How to allow for perception during motion?

(v) How can results be made available as soon as possible to the robot?

Figure 1.2 visualizes the structure of this thesis, which can be divided into the following parts (1) Semantic Scene Representation, (2) Next-Best-View Planning, (3) View Selection and Gaze Stabilization. The major contributions to these topics are highlighted in the following.

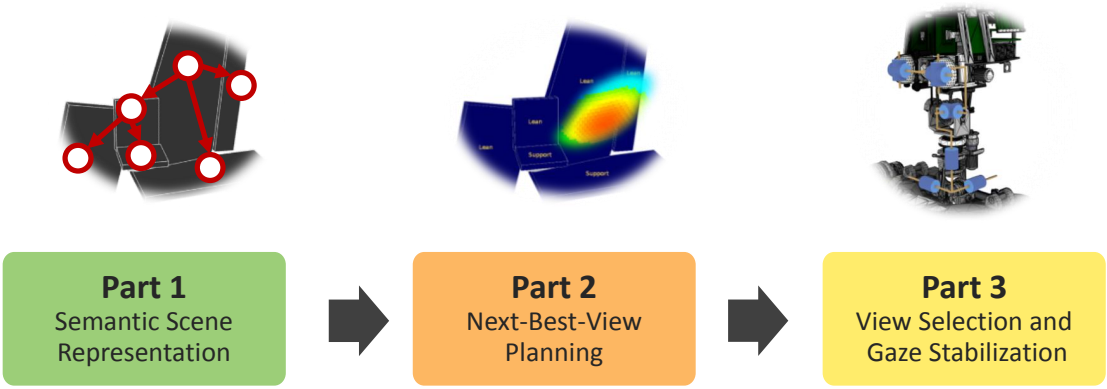


Figure 1.2.: The structure of the automatic scene perception approach. The approach starts by describing a semantic scene model built from the current view. The subsequent part determines the NBV to improve the scene model. Finally, a gaze stabilization controller is linked to allow perception during motion. The outline of this thesis follows the structure of the approach.

Semantic Scene Representation The robot captures the current view with an RGB-D sensor, and basic geometric shapes are fitted against the point cloud obtained from the current view. These geometric shapes, such as cuboids or cylinders, are abstracting objects and scene elements. The geometric model is then enriched with semantic information. Inferred spatial and semantic relationships among objects and scene elements allow for an understanding of the scene. The semantic scene model is iteratively improved by merging results from consecutive views into a global consistent scene model.

Next-Best-View Planning The presented scene model, is extended with an active vision method that deals with the Next-Best-View (NBV) problem. Planning the NBV allows completing the scene model. Therefore, possible views are sampled based on the semantic information already available in the previously presented scene representation. A view comprises the robot's position and gaze direction. In a subsequent step, the sampled views are then evaluated and ranked. The evaluation considers tasks aspects that a relevant for the automatic perception, such as the traveled distance and the volumetric information gain. By choosing the view which maximizes the utility function as the next view, the number of views necessary to interpret the scene is minimized.

View Selection and Gaze Stabilization Changing the gaze as well as executing manipulation actions impairs the visual perception. Thus, it is necessary to stabilize the gaze during motion. To this end, gaze stabilization methods are implemented on the ARMAR humanoid robots and further coupled with an active vision method that determines the next gaze direction for grasping and manipulations tasks. This part of the thesis investigates the link between active vision and gaze stabilization and evaluates the benefit of using gaze stabilization in real world experiments.

1.4. Outline of the Thesis

This thesis is structured into six chapters. Chapter 1 gives an introduction to the problem and is followed by an overview of related work in Chapter 2. The Chapters 3 to 5 describe the core of a technical system towards an autonomous system for visual scene perception in unstructured and unknown environments. These chapters share a similar general structure and begin with

an introduction to particular problems involved with the automatic perception of unknown environments. The chapters continue with the approach as well as an evaluation and presentation of the results. Each approach contributes to the stated goals (1) to (5) achieved contributions of the presented approach.

Chapter 2 gives an overview of relevant work and reviews related methods with respect to the presented approach. Related vision paradigms are introduced first. In particular, the term Active Vision is defined and differentiated from other definitions as it is key to perceive a scene autonomously. Next, the chapter proceeds with an in-depth review of existing approaches, which are relevant for this thesis, and a discussion of their advantages and limitations.

Chapter 3 defines the scene representation for unknown environments comprising geometric as well as semantic information. An approach to fuse results from consecutive views is also presented and evaluated. The chapter concludes with a brief summary of the semantic scene perception method and discusses the impact of this approach.

Chapter 4 presents a novel active vision method to autonomously determine the Next-Best-View, which completes the scene model presented in the previous chapter. Therefore, the chapter describes the extensions to the system architecture of the semantic scene representation. Finally, the chapter summarizes the results and discusses the major advantages as well as limitations of the presented NBV approach.

Chapter 5 links the presented active vision method with a gaze stabilization approach that is required to allow visual perception during motion. After an introduction to the problem, modalities for gaze stabilization and their dependency on a view target are presented. The chapter continues with an integration into the system architecture and a task-oriented evaluation. The chapter concludes with a summary and discussion on the importance of gaze stabilization and active vision for visual perception.

Chapter 6 concludes this work with a general discussion on the contributions of the thesis. It further gives an overview of future work on autonomous visual perception of unknown scenes.

2. Related Work

This chapter starts with a brief introduction of vision paradigms and surveys relevant work and alternative approaches with respect to this thesis. Since this thesis covers different robotic research areas, the state-of-the-art can be roughly grouped into the following three major categories: (i) Active Vision, (ii) Scene Understanding, and (iii) Gaze Stabilization. The structure is visualized in Figure 2.1.
To begin with, this chapter starts with definitions and ideas of visual perception paradigms relevant to this thesis in Section 2.1. Next, Section 2.1.2 classifies the approach of this thesis in terms of the given definitions as active vision method. Selected seminal contributions are also addressed in the following. The chapter continues in Section 2.2 with an overview of the relevant work on scene representation and understanding. Followed by active vision methods in Section 2.3. In particular, the section reviews active vision methods dealing with the Next-Best-View (NBV) problem. Described works are put into relation with the approach of this thesis and differences are highlighted. Subsequently, Section 2.4 details methods for gaze stabilization and gaze control. Finally, Section 2.5 summarizes and discusses the content of this chapter briefly.

2.1. Vision Paradigms

Vision is the most important information source in humans and a broad research area not only in robotics but also in neuro- and cognitive science. The seminal work *Vision* of Marr (1982) defines vision as the reconstruction of a description, and his theory has since influenced not only computational neuroscience but also other vision research. His work presents perception as the reconstruction of a description and presents vision as a processing chain with clear separation. This is known as Marr's paradigm. Marr argues that in order

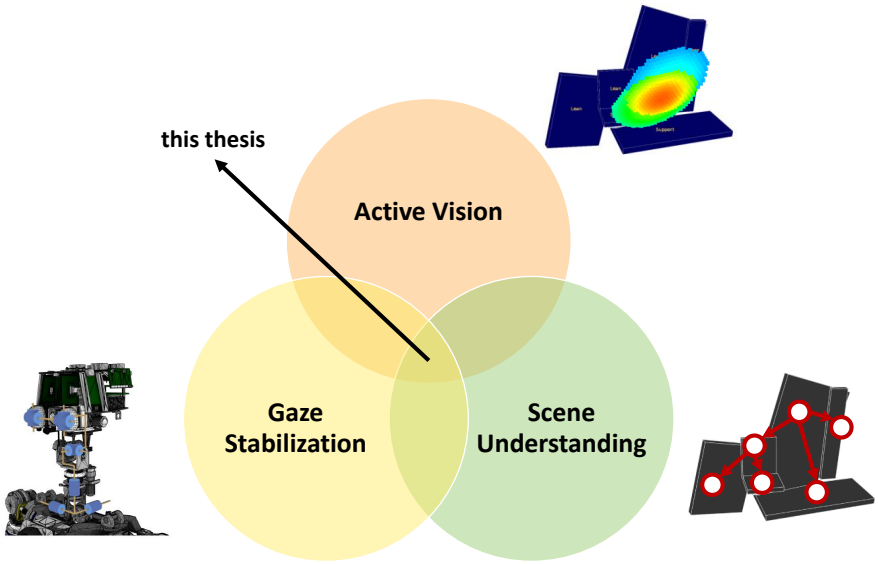


Figure 2.1.: The different research areas related to this work. Active vision is highly context and task-dependent. Therefore, this approach overlaps different research areas. The approach, as presented in this thesis, comprises of an active vision system to determine the Next-Best-View (NBV). The NBV improves a semantic scene model and allows to reason over stability and support among objects. Finally, the system architecture is extended by required gaze stabilization modalities to enable visual perception during motion.

to understand the visual process, different levels of abstraction have to be considered. To separate the visual process, he distinguishes between the following three abstraction levels:

(1) Computational theory level: This level defines the mathematical analysis and mapping from one kind of information to another.

(2) Representation and algorithm level: Here, the implementation of the computational theory is done.

(3) Hardware implementation level: This level describes the physical realization of the previous level.

The succinct description and the clear classification of the levels made Marr's paradigm popular, which can also be mapped to this thesis. The first level corresponds to the description of the method, i. e., the thesis itself. The second level equates to the implementation available in the Robot Development Environment (RDE) ArmarX (Vahrenkamp et al., 2015). The last level confers to the humanoid robot system, where the method is executed and evaluated. The separation is important when designing a perception algorithm. In his work,

Marr delineates the representation to extract a shape model from images into a process of

(1) first drafting a primal sketch from an image,

(2) then a 2.5D sketch of the scene, and

(3) finally, a 3D model representation.

As one can see, the model of the representation is built iteratively. Methods to build such a representation are addressed in Chapter 3. In Marr's work, the representation is constructed using viewer centered coordinate frames (*egocentric*). In robotics, especially in humanoid robotics, vision is embodied, active, and goal-directed. Hence, many robotic researchers advocated that vision is *active*, leading to new perspectives, which are discussed in the following.

2.1.1. An Active Approach to Vision

The assumption of a passive observer, the disembodied approach, and the lack of feedback are the main criticisms directed at Marr's work. Notably, the active perception paradigm (Bajcsy, 1988) and active vision paradigm (Aloimonos et al., 1988) in robotics have been proposed to acknowledge that vision is an active process and not a passive process. Definitions of vision paradigms that deal with vision as an active process include

(1) active vision (Aloimonos et al., 1988),
(2) active perception (Bajcsy, 1988) and (Bajcsy et al., 2018),
(3) animate vision (Ballard, 1991), and
(4) interactive perception (Bohg et al., 2017).

Since humanoid robotics has a strong link to the human being, the idea of foveal vision in humans is briefly introduced in the following.

Foveal Vision

Vision in humans is also active. This is due to the *fovea centralis*. The fovea centralis is a small area in the human retina which covers around two percent of the retinal surface. In this small area, vision has the highest acuity and allows for color perception (foveal vision). Visual input projected on areas of the retina other than the fovea centralis are perceived with lower resolution, and the color

Figure 2.2.: Recorded eye movements given a task. *Left:* Eye movements with the task "give the ages of the people." *Right:* Eye movements with the task "remember the clothes worn by the people." The experiment is described in Yarbus (1967). Figures taken from Archibald (2008) (© 2008 Cabinet).

information cannot be distinguished. The remaining part of the retina is hence used to monitor the scene (peripheral vision). Therefore, humans have to shift the gaze to perceive a view of the scene with high resolution.
In Yarbus (1967), subjects were asked to perceive a picture given a specific task while their eye movements were recorded. Depending on the tasks, the recordings show different movements and different regions of interest in the picture. Figure 2.2 shows the recorded eye movements overlaid of the picture.

In robotics, foveal vision is inspired by biological systems and mimics the human *fovea centralis*. In robotics, foveal vision is usually modeled by adding additional cameras with higher resolution (Fiala et al., 1994) or by using a pair of foveated wide angle lenses (Kuniyoshi et al., 1996). Other robotics systems with foveal vision, such as the Karlsruhe Humanoid Head (Asfour et al., 2008), are described in Appendix B. Relevant work on visual attention is discussed in Section 2.3.2. Shifting the attention is of utmost importance for foveal vision due to the narrow area with high visual acuity.

Active Vision and Active Perception

The *active vision* paradigm in robotics was coined by Aloimonos et al. in the late 1980s. The basic idea of active vision methods is the purposeful manipulation

of the camera pose and the camera parameters to improve visual perception. The authors wrote in their work

> *"An observer is called active when engaged in some kind of activity whose purpose is to control the geometric parameters of the sensory apparatus."*
> (Aloimonos et al., 1988, p. 333)

Similar to the active vision concept by Aloimonos et al., a broader definition was given by Bajcsy with *active perception* (Bajcsy, 1988). The author explains the term as follows:

> *"Active Perception (Active Vision specifically) is defined as a study of Modeling and Control strategies for perception. By modeling we mean models of sensors, processing modules and their interaction."*
> (Bajcsy, 1988, p. 996)

Therefore, the definition encompasses active vision as well and is not only limited to cameras. However, the focus of active perception is more on modeling and control strategies for perception according to the author.

Indeed, both definitions of an active observer are in contrast to the assumption of Marr's approach (Marr, 1982), where the observer is assumed to be passive. However, the purposeful manipulation of the camera pose is required to mitigate occlusions in the scene and to overcome the limited sensor's field of view. Both occlusion and sensor limitations, make active vision challenging even if a priori information is available. Despite these difficulties, an active observer is also necessary for many robotic applications and improves a robot's visual perception significantly. The early work of Aloimonos et al. (1988) further shows the superiority of an active observer vs. a passive one. For example, many vision problems, such as shape from shading, where an object's surface normals are estimated under different lighting conditions, are ill-posed problems when formulated with a static observer. By knowing the camera motion, the shape from shading problem can convert to a well-posed problem with an active observer. In addition, the early work *active vision* (Aloimonos et al., 1988) makes strong claims about the advantages of an active observer, i. e., one that can actively control the camera. One of the major claims is that active vision can also deal with noise in perception by controlling the camera.

The control of the camera is either to support the execution of the current task (*task-oriented*) or to respond to a perceived sensor cue (*stimulus-driven*). The link to perception and the current tasks was described by Aloimonos (1990) a few years later. Therefore, vision should not only be active but should be designed to solve a particular task. For example, active vision improves visual segmentation (Mishra et al., 2012), Simultaneous Localization and Mapping (SLAM) (Frintrop and Jensfelt, 2008), or visual object search (Welke, 2011). The approach presented in this thesis also has a strong link to the current task, i. e., creating a semantic representation of the environment.

Animate Vision

A more refined definition of active vision, called *animate vision*, was given by Ballard (1991). The animate vision paradigm is inspired by biological systems and anthropomorphic features. As an extension to active vision, the animate vision definition explicitly takes foveated vision into account and emphasizes gaze control of a vision system. Gaze control is the umbrella term for different mechanisms to keep a target centered. This includes, for example, gaze stabilization behavior or saccadic movements to attend a new region of interest. The importance of gaze control, in conjunction with gaze stabilization, is highlighted in Section 2.4 with respect to relevant work. In Section 5.2, a gaze stabilization controller is integrated into an active vision system.
The definition of animate vision also emphasizes the usage of the coordinate system, which acts as a reference for the scene model. In contrast to Marr, the coordinate system should be a world coordinate system (*exocentric*) and not viewer centered (*egocentric*). Animate vision assumes that the observer moves with a known motion, and therefore, the coordinate system can be exocentric and aligned with respect to a fixed position and orientation in the scene. An exocentric coordinate frame allows to merge information from multiple views and to facilitate a spatio-temporal memory structure to store processed results. A memory-based structure is nowadays widely used in robotic systems. For example, with the knowledge service openEASE (Beetz et al., 2015), robots can recall memorized manipulation episodes. A different action matching and retrieval mechanism to recall episodes were presented in (Rothfuss et al., 2018). Animate vision, however, utilizes foveal vision, and not every robotic system is equipped with one. Additionally, mapping the viewer-centered coordinate system to a world coordinate system can be inaccurate, since the transformation

is often imprecise due to kinematics inaccuracies or errors in the localization of the agent. Besides that, the core idea of an active observer is similar.

Revisiting Active Perception

In a broader context, the active perception paradigm has been revisited recently (Bajcsy et al., 2018). The authors Bajcsy, Aloimonos, and Tsotsos extend the active perception definition to more sensor modalities and give an overview of the development and view on active perception. Among other things, the new definition takes up the concepts of animate vision and generalizes them to other sensor modalities. The definition of active perception is thus distilled to

> *"An agent is an active perceiver if it knows why it wishes to sense, and then chooses what to perceive, and determines how, when and where to achieve that perception."* (Bajcsy et al., 2018, p. 178)

The components *why*, *what*, *how*, *when*, *where* are used to distinguish the term from other definitions, and the *why* component is identified as key distinguishing factor. Figure 2.3 visualizes the elements of the active perception definition. For this thesis, the *where* and *how* components are important. The *where* includes the particular view and the *how* includes the alignment of the camera. Altogether, the authors point out that an agent has to be active in order to perceive.

Interactive Perception

The definition of active perception can also include interaction with the environment. Interaction is key for robotic systems and therefore a major focus for the definition of *interactive perception* (Bohg et al., 2017). Interactive perception is defined as any kind of forceful interaction with the environment to improve perception. Formally, the definition is restricted to interaction that affects the space $S \times A \times t$, where S denotes the sensor information, A the action parameters, and t the time. The action space A is further divided into forceful interaction and interaction that only affects the sensory apparatus and not the environment. Thereby, the authors distinguish their definition from the active

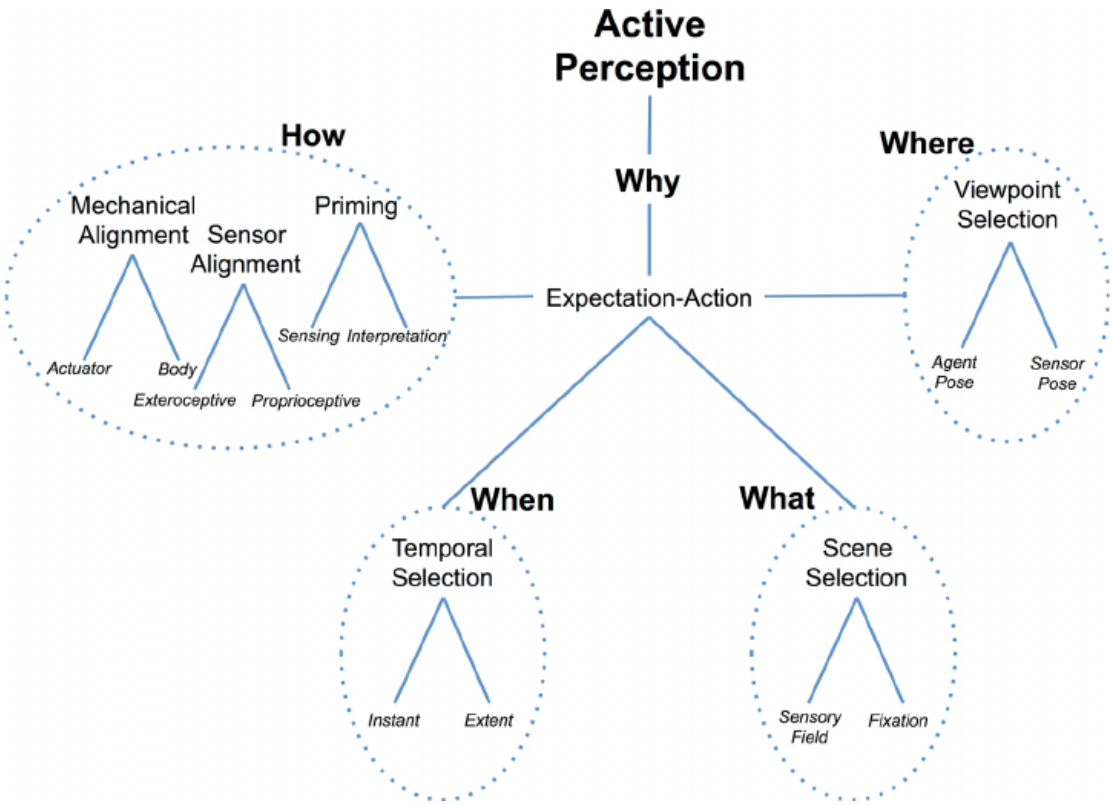


Figure 2.3.: The definition of active perception decomposed into basic elements, *why*, *what*, *how*, *when*, *where*. Depending on the *why* an active observer controls the other elements *what*, *how*, *when*, and *where*. Figure taken from Bajcsy et al. (2018) (© 2018 IEEE).

perception definition. Interactive perception supports grasping actions or the segmentation of unknown objects (Tsikos and Bajcsy, 1991; Schiebener, 2017).

2.1.2. Classification of the Approach

So far, this chapter presented several vision paradigms for an active observer, which often have several features in common, and the definitions are overlapping. In the following, features of the approach, as presented in this work, are described. Figure 2.4 distinguishes the vision paradigms by their main characteristic.

Altogether, this thesis builds an active vision system due to the following reasons. First and foremost, the observer is active. An active observer constitutes the key element of active vision. Although the scope of this approach is consistent with other definitions, such as active perception, the definition of active vision is more suitable since only unimodal sensory input is considered. Further,

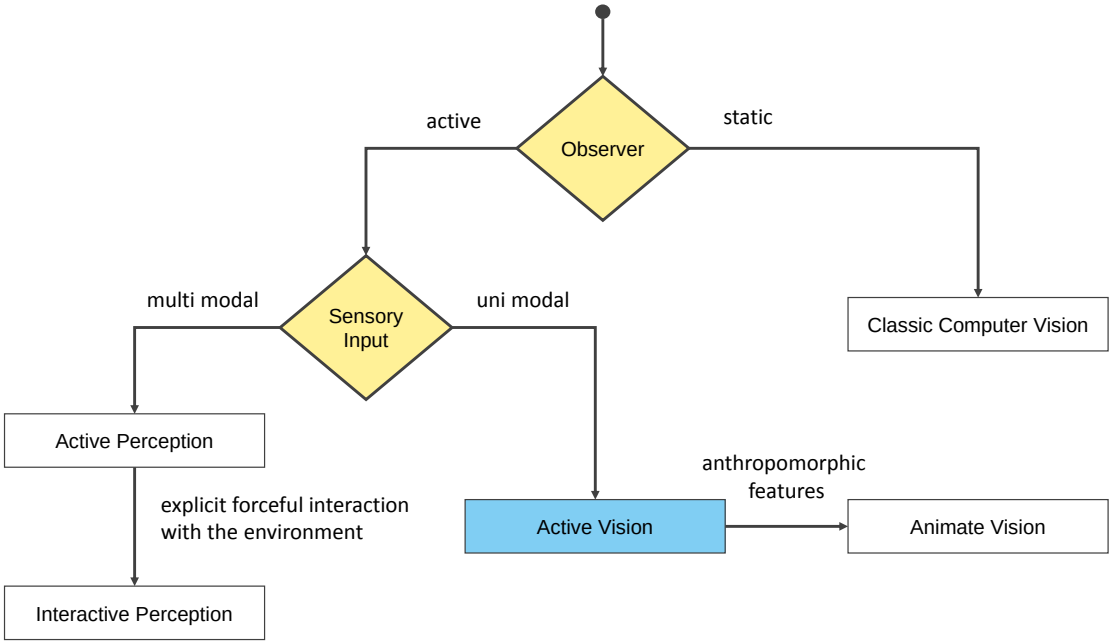


Figure 2.4.: Differentiation of vision paradigms as used in the thesis. Definitions share common design features. Here key design features are identified and used to distinguish the definitions.

the focus is more on the autonomous perception of unknown environments than on manipulation actions. Animate vision emphasizes the advantage of anthropomorphic features, which are also used by the methods. However, another focus of animate vision is behavior-based control, and the focus of this work is more on planning. Therefore, the approach, as presented in this thesis, can be seen as active vision method.

2.1.3. Summary

This chapter provided an overview of relevant vision paradigms that have influenced and shaped the research areas in robotic vision. In particular, this includes the definition of active vision (Aloimonos et al., 1988), active perception (Bajcsy, 1988) and (Bajcsy et al., 2018), animate vision (Ballard, 1991), and interactive perception (Bohg et al., 2017). The approach, as presented in this thesis, can be seen as an active vision approach.

Robotic vision differs from classic computer vision in the way that the embodiment of the robot is a key aspect. A robot is an active observer. Having an active observer resolves many issues, which are difficult for a passive observer.

	Vision Paradigm				
	Classical CV	*Active Vision*	*Animate Vision*	*Active Perception*	**This approach**
Image processing	✓	✓	✓	✓	✓
View selection	-	✓	✓	✓	✓
Embodiment	-	✓	✓	✓	✓
Anthropomorphic features	-	-	✓	-	-
Multi-modal sensory input	-	-	-	✓	-
Change environment	-	-	-	✓	-

Table 2.1.: Definitions of vision paradigms. A checkmark ✓ indicates that the feature is explicitly taken into account by the definition. Table adapted from Asfour (2019).

Chapter 1 formulated the challenges (1) to (5) for an autonomous visual perception. This includes, for example, the effect of occlusion, which can be resolved by changing the view. In particular, for humanoid robotics being active is even necessary due to the anthropomorphic design. Robots have to focus on what's relevant for their current task. Hence, the *active* aspect is an essential part of this thesis and constitutes the core of the presented approaches. Table 2.1 summarizes the different definitions with the main focus on visual perception. With respect to the presented vision paradigms, the approach of this work was classified as an active vision method.

2.2. Scene Modeling and Scene Understanding

As pointed out by Marr, the representation of the perceived sensor stimuli plays a crucial role in perception. A scene model is required to map the real world to an internal representation for the robot. Therefore, this section describes different approaches to build a scene model and its automatic interpretation.
The major aspect of a scene model is the spatial representation of the scene. Indeed, a scene model is not limited to the spatial representation, but can also incorporate different aspects, such as semantic or temporal information. Scene understanding is the automatic interpretation of the current scene model or the image. Both, a scene model as well as semantic scene understanding are essential for a robot when interacting with the scene. In addition to the embodiment, robotic vision differs further from classic computer vision or machine vision: While classic computer vision focuses more on 2D vision and the interpretation of camera images or videos. Robotic vision, instead, relies more on 3D information of the scene and considers the robot's embodiment and its ability to interact with the scene. Due to the necessity to control a robot in real-time, robotic vision requires a fast computational time, if not real-time, to process information required for the robot's control (Corke, 2011). The focus in this section is therefore on 3D scene modeling and 3D scene understanding with respect to robotic vision.
The section is organized as follows. Section 2.2.1 overviews scene modeling approaches in general. Section 2.2.2 details methods that decompose a scene using geometric primitives. Section 2.2.3 reports on scene understanding methods. This section concludes with a summary.

2.2.1. Scene Modeling

Choosing a scene model is also an important aspect when designing an active vision system since the model has to support the current task. Therefore, different design goals have to be considered for the scene representation. After the robot's current view has been captured with a sensor, the data needs to be mapped to a meaningful representation. The extracted scene model is then used for further processing and allows for view independence and further reasoning. This step of building a scene model comes with a loss of geometric information. Choosing the model for the scene representation is therefore an

important step to communicate and analyze the content of the scene.
Since a scene can be represented in several ways, a good choice of a scene model is one that contains all the necessary information and details required for the current task. At the same time, the scene model should also be memory and access efficient, since robotic vision is time constrained and a robot's resources are limited. The choice of the scene model often depends on the current task. For example, mesh based representations are popular whenever a model of an object is created automatically. These approaches often utilize a polygon mesh or a function to approximate a surface or the shape of an object. The advantage of a polygon mesh is that this approach can be mapped to any kind of surface. By increasing the number of polygons in the mesh, the precision of the scene model can be increased. Implicit surface models (Bloomenthal and Bajaj, 1997) are another way to model the surface of an object. These models map geometric surface models to the sensor data. In general, these methods have high accuracy and can also model uncertainty. However, implicit surface models are often limited to the geometry of the object. Hence they cannot be used for a complex room layout.

For scene modeling, different approaches are used. In the following, it is assumed that the data is available either as point cloud or RGB-D image. In robotics, the most common methods are to store the data as an accumulated point cloud or in a voxel grid. Here, mapping a scene with methods like KinectFusion (Newcombe et al., 2011; Izadi et al., 2011) are very popular due to online capabilities. KinectFusion runs on a Graphics Processing Unit (GPU) and represents the volumetric 3D map as implicit surface model using the Truncated Signed Distance Function (TSDF). Recent methods are surveyed in Zollhöfer et al. (2018).

Henry et al. (2012) utilize a surfel-based map for dense 3D modeling of indoor environments. *surfels* are point primitives without explicit connectivity (Pfister et al., 2000). A surfel also comprises normal, color, and other information. Therefore, multiple views from an RGB-D sensor are registered with respect to each other using a modified version of the Iterative Closest Point (ICP) algorithm, called RGB-D ICP. The representation is to enable compact representations and visualizations of 3D maps. Wagner et al. (2013) focus on dense mapping of the environment. To register multiple views, the authors utilize the KinectFusion algorithm to represent the environment in real-time for DLR's humanoid robot Justin. The method utilizes the robot's forward kinematics model to improve the registration process. Further, the work also includes a

multi-scale approach to deal with the large map volumes to reduce storage and keep the required accuracy.
Approaches presented in this thesis make use of ElasticFusion as presented in Whelan et al. (2017). The core idea of ElasticFusion is similar to KinectFusion and features loop closure to improve the registration result. It uses a surfel-based representation and is able to detect the light source. To improve accuracy small local model-to-model loop closures combined with larger scale global loop closures are implemented. The loop closure ability is one of the major differences to KinectFusion. Similar to KinectFusion the approach assumes a low latency data throughput and requires a GPU. Noteworthy, some extensions to dynamic environments have been proposed. However, they have not been proven practically on a real humanoid robot since they are computationally expensive and sacrifice accuracy of the registration for the sake of having a dynamic scene model.

Input data can be reduced by employing a voxel grid. Voxelized representations approximate a scene by discretizing the spatial data with a 3D grid. The major disadvantage is the memory requirement. An octree-based representation can be used to reduce the required storage size of the scene. This data representation is more memory efficient since it allows for a flexible and multi-resolution. The more compacter representation is an advantage of voxelization over surface-based approaches. It contains the necessary trade-off between accuracy and efficiency. In addition, they allow to easily model the occupancy state of a voxel in a probabilistic manner. Therefore, voxelized representation can aggregate data from multiple sensors. For an overview of probabilistic scene models the reader is referred to Thrun et al. (2005). Due to the discretization, voxelized representations are advantageous in terms of free space reasoning. This is a limitation of geometric primitive based representation since it is difficult to reason over unknown space.
The work in Hornung et al. (2013) presents OctoMap, a popular framework based on Octrees. The octree data structure is used to represent the scene with a probabilistic 3D occupancy grid. The octree data structure allows for a memory efficient data representation. Each voxel has a loglikelihood probability of being occupied to represent free space. The loglikelihood is used to make an update easier. Knowing the current pose of the sensor, the map is updated by casting rays to the sensor measurements. Thereby, the framework allows combining different sensor modalities. Another framework, GPU-Voxels (Hermann et al., 2014), optimize update and query speeds with a GPU implementation to speed-up collision detection and path planning.

2.2.2. Geometric Primitive Detection

To interpret the data, a common approach is to decompose it into geometric primitives, such as planes, cylinder, or other parameterized surface types. Early work in Roberts (1963) on 3D representation already had the assumption of geometrical primitives. The survey of Kaiser et al. (2018) gives an overview of 72 geometric primitive detection methods. The work divides the different approaches into seven categories: RANSAC, Hough transform, Primitive growing, Local statistics, Clustering parameter, Automatic clustering, and Segmentation then fitting. Theoretical methods are then compared qualitatively. For robotics, the *RANSAC* as well as the *Segmentation then fitting* categories are particularly relevant. Segmentation then fitting approaches apply first a segmentation and then reason over each segment. Methods belonging to the RANSAC category fit the geometric primitive models using Random Sample Consensus (RANSAC) based approaches (Schnabel et al., 2007, 2008; Li et al., 2011). Due to the RANSAC based nature, these approaches are also robust against outliers. Figure 2.5 displays an example of geometric primitives.

Schnabel et al. (2007) use registered LIDAR scans to obtain a point cloud based representation, which is then automatically decomposed into geometric primitives. An octree is used to accelerate the model fitting. The advantage of such representation is that it already classifies the surface into geometric elements and therefore reduces redundancy. Further, the representation requires less storage compared to voxelized representations since large surfaces can be represented with a few parameters. The work of Schnabel et al. (2007) has also been utilized for robotics applications in Berner et al. (2013).
The idea of approximating objects with geometric primitives has been used for

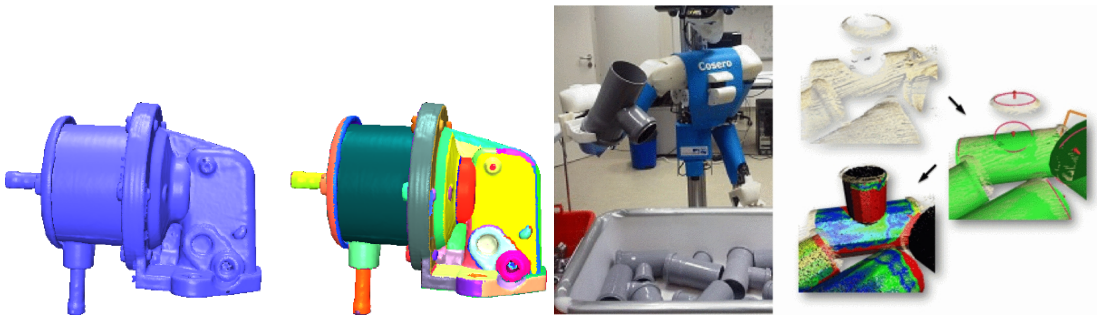


Figure 2.5.: *Left:* A LIDAR scan decomposed into parts. A RANSAC based geometric primitive approach is used. *Right:* This approach has also been utilized for robotic applications. Figure taken from Schnabel et al. (2007) and Berner et al. (2013) (© 2007 and 2013 IEEE).

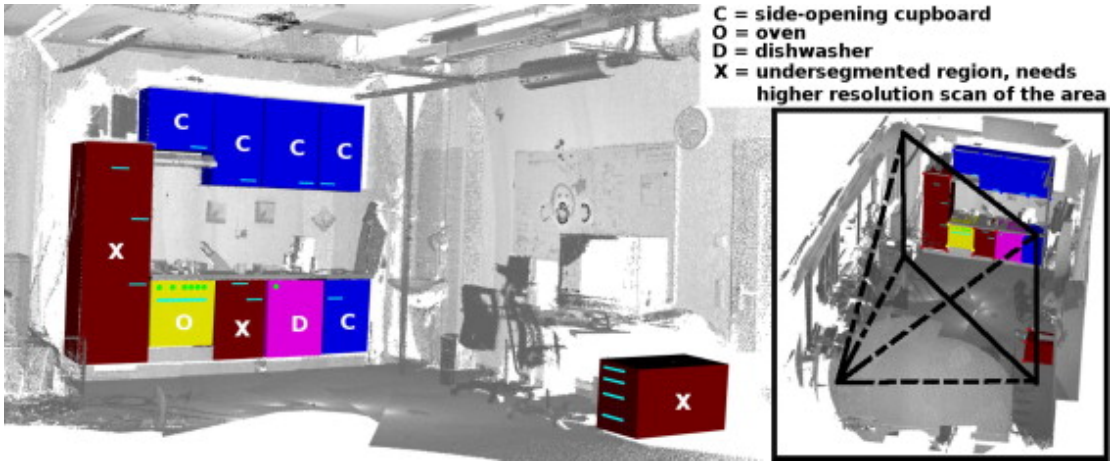


Figure 2.6.: A kitchen environment and detected elements. Figure taken from Rusu et al. (2008) (© 2008 Elsevier). After a LIDAR scan of the scene is taken, the given point cloud is filtered, segmented and object hypothesis are fitted. Finally, a functional reasoning step infers classified objects.

indoor environments. Rusu et al. (2008) build a 3D scene model for indoor environments based on point cloud data. After noise filtering, the point cloud of the current view is registered into a single consistent point cloud using the ICP algorithm. The coordinate system is a predefined world coordinate system. After registration, the point cloud is segmented into plausible parts, and against each segment object hypotheses are fitted. The representation includes a functional reasoning process for kitchen environments. The approach was designed for mobile robots working in kitchen scenarios. Therefore, the scene is represented as cuboids to approximate the structure. High-level features, such as knobs, are extracted as well. Figure 2.6 visualize a point cloud based map for a kitchen scenario. While cuboids might work well for kitchen environments, real world scenes are often more complex, and thus other geometric shapes have to be considered. This work was also extended with a NBV planning approach (Blodow et al., 2011). A more complex algorithm is presented in Hager and Wegbreit (2011), which handles dynamic environments as well. The approach uses a priori information about the 3D model of the objects and is therefore limited to parametric models, such as cuboids or cylinders. To be computationally efficient, the scene parsing algorithm consists of approximation algorithms derived from a maximum a posteriori (MAP) formulation of the scene. The algorithm is able to infer support relations and detect changes in the scenes. Changes are molded as Markov dynamical system model. The view, however, is static. The scene parsing algorithm is evaluated in several complex

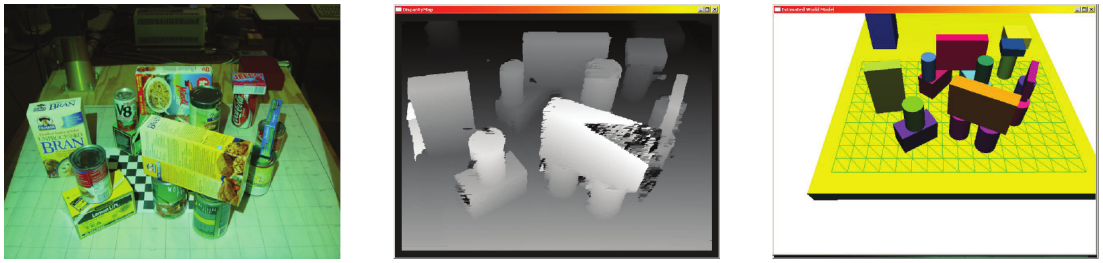


Figure 2.7.: Experimental results of Hager and Wegbreit (2011) (© 2011 Sage). The scene is captured with a stereo camera system and decomposed into parametric geometric models. *Left:* the scene comprising of 17 objects. *Middle:* the disparity image. *Right:* the resulting scene model.

scenes both in simulation as well as with a stereo camera system. Experimental results are depicted in Figure 2.7.

Geometric primitives can also be used to improve other robotic tasks. The approach of Richtsfeld et al. (2012) fits planes and non-uniform rational B-splines against a segmented point cloud in order to detect objects. Relations between patches are modeled in a graph structure and used as object hypotheses. Other application includes SLAM, which can be improved by fitting planes to the scene (Biswas and Veloso, 2012). Here, the work detects planes from an RGB-D sensor. The plane detection has several advantages as it already decreases the input size of the data being processed. Next, planes are projected to 2D and used for localization within a 2D map of the environment. For humanoid robots, a 2D environment representation is not sufficient as it does not allow for planning manipulation actions with the environment. Geometric primitives can also be used for staircase detection (Westfechtel et al., 2016), segmentation (Pham et al., 2016), and footstep planning (Wahrmann et al., 2019). A major disadvantage of RANSAC based methods is that they have many parameters to tune. Therefore, Fang et al. (2018) presented a parameter-free plane extraction method.

2.2.3. Scene Understanding

Having a scene model is not sufficient for most robotics tasks. To support the current task, the robot needs to infer relationships and semantic information as well. Semantic information can be derived from a 2D segmentation of the current view and then combined with a simultaneously created 3D map of the en-

vironment (McCormac et al., 2017). Another application field that benefits from semantic information is object detection, where especially spatial relationships are helpful. Meißner (2020) uses spatial relations that are encoded as an implicit shape model. These are then used to choose a Next-Best-View (NBV) for object search tasks. More details on the NBV problem is given in Section 2.3.1. Spatial relationships, such as neighborhood relationships, are required to match shapes in the scene. Neighborhood relations between the geometric primitives are represented within a graph structure (Schnabel et al., 2008). These neighborhood relations can then be used to redetect distinct shapes in the scene. The set of geometric primitives matched to the point cloud represents the vertices in the topology graph. An edge is added if two primitives are adjacent to capture neighborhood relation between the primitives. This graph structure can then be used to identify structures in the environment. The authors additionally present an algorithm for querying the scene graph.

Spatial information is also important to improve the accuracy of a scene model. Gupta et al. (2010) use relationships to refine a model based on cuboids. These blocks are pairwise enriched with additional simple relationships. Based on estimated depth relationships, split and merge proposals for the extracted blocks are created. To join two primitives that are separated by an occluding obstacle, the authors use volumetric constraints as a hint. Rosman and Ramamoorthy (2011) model the scene with respect to spatial relationships from a segmented point cloud. A graph-based structure is derived from a contact point network to abstract the objects. Spatial relationships are extracted from a minimum weighted spanning tree. The tree is based on support vectors, trained by a Support Vector Machine (SVM). Neighborhood relations are then described with *on* and *adjacent* relations. Such relationships can also be extracted from video streams and not only single images. Zampogiannis et al. (2015) learn the spatial semantics of manipulation actions. During manipulation actions, the objects are tracked in an RGB-D video. They extract directional relative space relations between involved objects, such as *in*, *left*, *right*, *front*, *behind*, *below*, or *above*. The approach is validated on the Baxter humanoid robot.
Semantic reasoning can also be used to parse large scale point clouds into segments forming semantically meaningful spaces to extract structural and building elements (Armeni et al., 2016). Parsing the scene into semantic objects with an active observer was studied by Zheng et al. (2019).

Other relations are physically plausible relations, which are often derived from spatial relations. Such physically plausible relationships are important for ma-

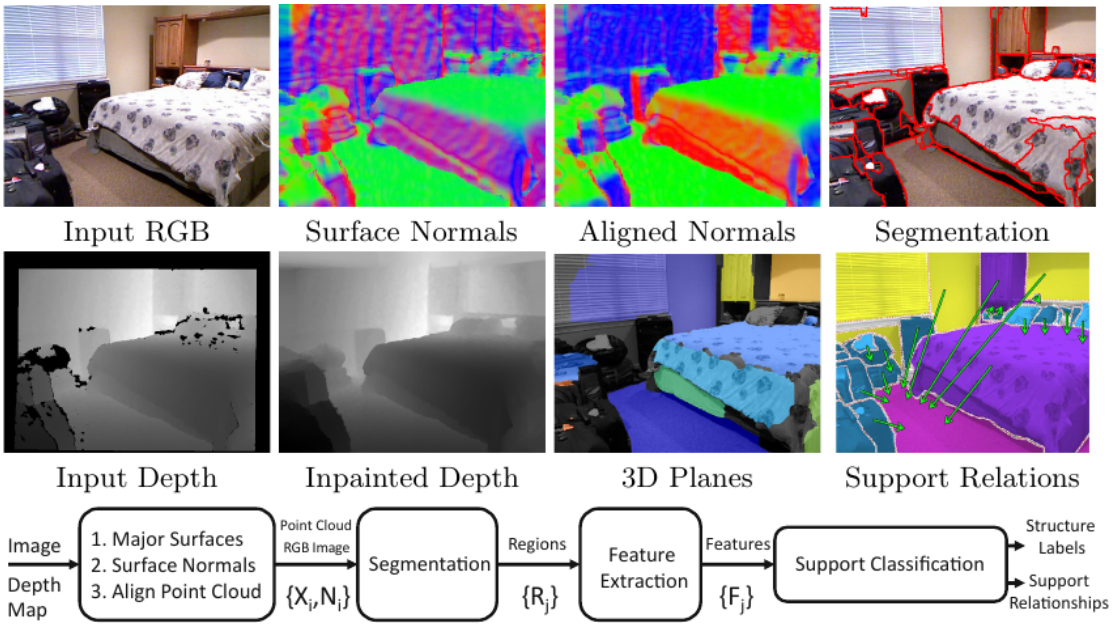


Figure 2.8.: Processing pipeline for support classification as presented in Silberman et al. (2012) (© 2012 Springer). Based on RGB-D images planes are fitted using RANSAC and finally physical support is inferred.

nipulation actions. These relationships can then be used to define a manipulation order for cluttered environments (Panda et al., 2013). For example, given a sufficient understanding of the scene, a robot can manipulate objects in a meaningful way or utilize the structure of the scene.
Silberman et al. (2012) identify support relations in indoor scenes by a maximum a-posteriori (MAP) inference, interpreting major surfaces and objects from RGB-D images. First, the scene is segmented into objects and surfaces. Supporting planes, as well as the floor and ceiling planes, are found using a RANSAC approach. In an additional step, physical relations are parsed from the model. The dataset is published as the NYUv2 dataset. The algorithm is outlined in Figure 2.8.

Jia et al. (2013) present a global stability criterion by averaging over the center of mass and volume. The approach fits 3D cuboids against an over-segmented RGB-D image. The fitted cuboids are then used to refine the initial segmentation. Boxes are considered as unstable if the projected support area does not include the center of mass. The work then infers three different support relations: on top, partially on-top, or side support. The reasoning process is then used to extract features to propose a splitting and merging approach of the boxes. Mojtahedzadeh et al. (2015) identify contact points and differentiate between contact types. The result of their method is a Support Graph (SG), which

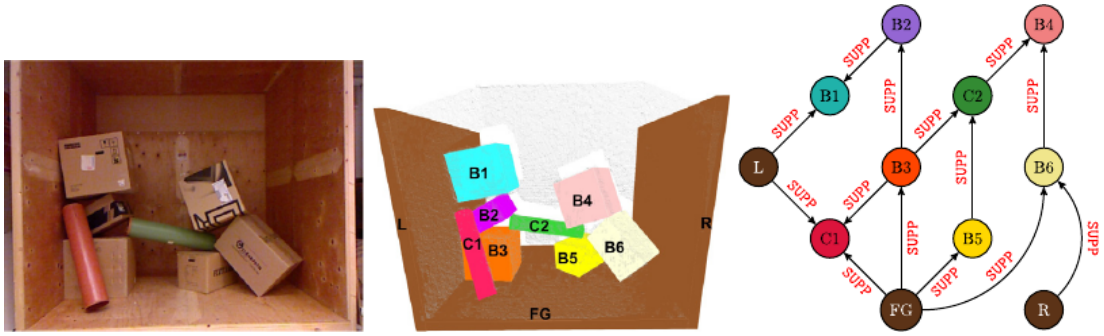


Figure 2.9.: Experimental results of Mojtahedzadeh et al. (2015) (© 2015 Elsevier). The approach extracts support relations by geometrical and static analysis. *Left:* the scene. *Middle:* detected objects. *Right:* The inferred physical support visualized as support graph spawning among the objects.

models the support among objects. The contact points are then aggregated into a network to determine support relations between the objects. The network is called Contact Point set network (CPSN). The goal is to reason which object can be removed from the stack without affecting the static equilibrium. To this end, the work distinguishes between complete (CSO) and incomplete set of objects (ICSO), i. e., not all objects in the scene are captured. For the first category, a non-linear system is solved to test if an object can be removed from a CSO. A machine learning approach is proposed for the incomplete case, i. e., the ICSO. The feature vector for an object is composed of geometrical features. These include the axis-aligned bounding box, the centroid, six distinct points of interest. However, the approach assumes that the scene is perfectly segmented, and the authors therefore manually label the data. Figure 2.9 shows the results of an experiment and extracted support relations.
Kartmann et al. (2018) build on the work of Mojtahedzadeh et al. (2015) with the focus on safe bimanual manipulation actions with a humanoid robot. Real world experiments with the ARMAR-III robot evaluate the methods. Section 3.2.2 details the method further. Both in (Kartmann et al., 2018) and (Mojtahedzadeh et al., 2015) the scene has to be manually labeled, and views are chosen manually as well.

2.2.4. Summary

This section gave an overview of relevant work for 3D modeling and interpreting a scene. A scene model is essential for real-world robotic applications.

The presented methods on scene modeling differ in terms of representing and storing the data. Here, efficiency is a major design criterion since a robot's computational resources are limited. Another crucial aspect is the interpretation of the 3D model. Therefore, the scene model must have a suitable spatial representation. Geometric primitives are particularly suitable here. With few exceptions, views are selected manually and not automatically. Also, the data needs to be interpreted with additional reasoning steps. Spatial and physical reasoning allow improving many robotic applications. Table 2.2 overviews the most important scene representation approaches. Further, inferring physically plausible support relations among objects is required for a humanoid robot when interacting with the scene. Finally, with a few exceptions, the presented works focus on the semantic scene representation and thus not consider an active agent.

2.3. Active Vision

In the survey of active vision system in robotics of Chen et al. (2011) it is stated that,

> *"high-level representation and reasoning depend on, but also affect the low level vision perception."* (Chen et al., 2011, p. 1370)

It follows that geometric and semantic information, as well as knowledge of spatial relationships, needs to be combined to fully utilize a robotic vision system. Hence in the following an overview of active vision systems is given. The survey of Chen et al. (2011) covers industrial and mobile robotics as well. Due to the wide application in robotics, the authors distinguish active vision systems in its purpose/task and method. Such purposes of active vision systems include, but not limited to, *grasping*, *exploration*, *object modeling*, and *site modeling*. These purposes are especially relevant for humanoid robots, and thus in the following presented active vision methods are categorized with respect to these purposes.
The section is organized as follows. First, early works on active vision methods are reported, followed by active vision methods with the focus on grasping. Next, Section 2.3.1 overviews NBV methods focusing on object modeling or exploration. Furthermore, deep learning approaches are presented. Due to the

Publication		Geometric Primitives				Scene Understanding		
Author	Year	Planes	Cuboids	Spheres	Cylinders	Spatial Reasoning	Physical Reasoning	Affordances
Schnabel et al.	2008	✓	-	✓	✓	✓	-	-
Rusu et al.	2008	✓	✓	-	-	✓	-	(✓[a])
Hager and Wegbreit	2011	✓	✓	✓	✓	✓	✓	-
Silberman et al.	2012	✓	-	-	-	✓	✓	-
Jia et al.	2013	-	✓	-	-	✓	✓	-
Mojtahedzadeh et al.	2015	-	✓	-	✓	✓	✓	-
Kartmann et al.	2018	-	✓	✓	✓	✓	✓	-
This approach		✓	✓	✓	✓	✓	✓	✓

Table 2.2.: Comparison of relevant approaches for semantic scene representation of unknown 3D environments

[a] Functional reasoning

complexity of the NBV problem, deep learning approaches are only of limited use for a robotic system but are becoming more and more popular. Section 2.3.2 covers attention based methods. These methods are often biologically inspired. Finally, results are summarized.

An active vision system can have different aspects. Therefore, Aloimonos (1990) presents an active vision system, called *Medusa*, to consider different aspects of perception, such as object tracking. The idea behind this system is that the perception process is decomposed into simple processes, which are then considered by a master controller to plan the gaze. By this layer of abstraction, the individual process can be very simple, but overall the system can be powerful due to many processes.

To obtain a scene representation, for example, various approaches have been suggested. Marchand and Chaumette (1999) developed a scene exploration algorithm for static scenes composed of cylinders and polyhedral objects. Their approach features three stages: exploration, primitive reconstruction, incremental reconstruction. Their active vision system reflects different aspects, such as exploration.

Besides exploration, another particular application of active vision is object grasping. To this end, Rasolzadeh et al. (2010) present a visual attention system for the Karlsruhe Humanoid Head (Asfour et al., 2008). The cognitive system leverages peripheral and foveal cameras in order to segment, detect, and grasp objects. The focus of attention is determined by combining dynamically bottom-up and top-down saliency. Therefore, an artificial neural network approach is used to learn the optimal bias of the top-down saliency map. A stochastic Winner-Takes-All (WTA) approach is used to shift the view to the regions with the highest saliency value. The system also models the Inhibition of Return (IOR) mechanism, which is used to promote the exploration of unattended areas. Attention mechanisms can also include task-relevant information, such as grasping and manipulation actions. Bohg et al. (2012) present a complete grasping pipeline for object grasping. The system includes an attention mechanism to fixate the object. The attention mechanism uses geometric information and determines fixation points which are then used as initial seed points for the object segmentation. The system is evaluated on the humanoid robot ARMAR-III (Asfour et al., 2006).
An important aspect for grasping and manipulation actions is the uncertainty of object localization results (Eidenberger and Scharinger, 2010). Welke et al. (2013) also propose a view selection strategy based on the uncertainty of object

localization results. The approach is tailored to known objects, but works in dynamic scenes. In addition to the requirement that the environment is known a priori the work neither considers occlusion and only moves the cameras of the robot and not the robot itself.
Other aspects include reliable grasping, i. e., to detect failure or grasp success. Arruda et al. (2016) present a view selection strategy for grasping unknown objects. Here, the active vision system is clearly task-driven. The camera is mounted on the wrist of a dexterous Schunk hand. The active vision system is designed to maximize surface reconstruction quality around contact points and to refine grasp gaze direction planning to improve safety. A 3D occupancy map is used to model the scene.
Kahn et al. (2015) use a frontier-based approach to grasp hidden objects. The camera planning is optimized for the grasping trajectory. The approach includes the NBV problem, which is highlighted in Chapter 4 of this thesis. Gualtieri and Platt (2017) investigated the choice of a view pose to increase the accuracy of a grasp detection method for the Baxter robot.

With the research peak in deep learning, these approaches are also quite popular for application in active vision. Sünderhauf et al. (2017) give an overview of recent achievements and challenges in robotics using deep learning. The authors point out the difficulty with deep learning. A major issue is the embodiment of the robot that relates to active vision and active perception approaches. Especially the authors note that

> *"a more holistic approach to active scene understanding is still missing from current research."* (Sünderhauf et al., 2017, p. 408)

Approaches are mainly evaluated in simulation due to the lack of training data. In Ammirato et al. (2017), an annotated dataset for active vision is presented. The dataset is for object search. The approach is for mobile robots and thus steers only a mobile platform. The authors also present a reinforcement learning approach for predicting the next motion for object classification tasks. The next best motion consists of six direction commands. By executing these commands, the robot gets a new view. Although sophisticated learning approaches for active vision are limited due to the embodiment of the robot, there are some notable exceptions. Cheng et al. (2018) present a reinforcement learning approach, which keeps an object within the field of view during manipulation actions and is able to deal with occlusion. The approach is evaluated in simulation and the estimation is memoryless.

2.3.1. Next-Best-View Planning

As already mentioned, active vision should be goal-driven (Aloimonos, 1990). One particular problem is the generation of a complete scene model or object model. The NBV problem is defined as finding the next view to iteratively obtain a reasonably complete model of a scene or an object. Again, the camera should be controlled in a goal-directed manner. In general, the total number of views for the object or the scene model should be minimized.
The NBV was widely addressed for inspection tasks using a robotic arm with a priori knowledge about the object. For more details, the reader is referred to the survey of Scott et al. (2003) that distinguishes between model-based and model-free approaches. However, the problem is different if no a priori information is available or if solved with a humanoid robot. In the following this focus is on NBV designed for humanoid robots, but also present other relevant approach not related to humanoid robotics.

The NBV problem was pioneered by Connolly (1985). The early work presents two algorithms to address the NBV problem, namely (1) the Planetarium algorithm, and (2) the Normal algorithm. The first algorithm samples views on a sphere and uses ray casting to estimate the unknown space. The second algorithm uses surface normals to compute the NBV. Connolly acknowledges that the estimation takes too much time and therefore suggested the second approach. Both algorithms assume that the object needs to be inside a sphere. Further, the work does not consider occlusion or any kinematic constraints, such as the limited degrees of freedom of a robot.

Notably, the NBV problem has been improved by the work of Pito (1999) and Banta et al. (2000). Pito uses a polygon mesh as a data structure and surface normals to get an accurate 3D reconstruction for an object on a turntable. The work distinguishes between what must be scanned and how. To this end, the work introduces an intermediate positional space (PS) between object and workspace to facilitate what must be scanned. Pito uses surface patches to determine the NBV. The representation used by Pito, i. e., a polygon mesh, has some advantages over the volumetric model space for object modeling tasks. However, as pointed out by Torabi and Gupta (2011), the approach does not scale to robotic systems with many degrees of freedom. Here, presented works mainly consider volumetric approaches since they allow for probabilistic occupancy estimation and visibility checking.

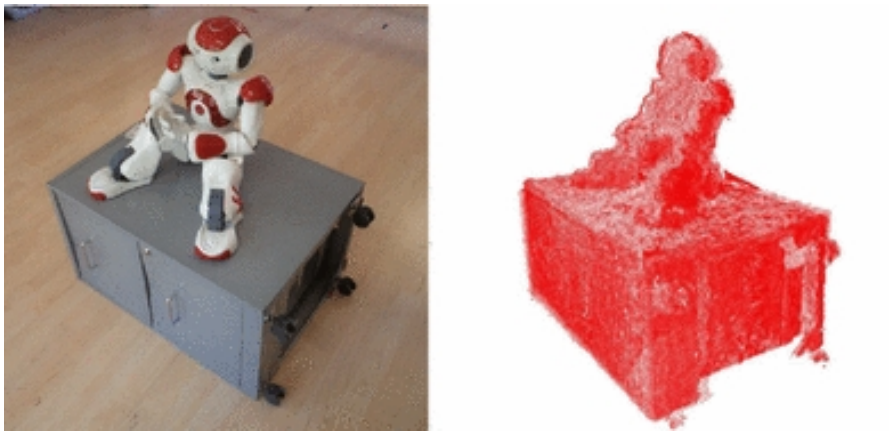


Figure 2.10.: Reconstruction of a NAO robot by a NBV approach which considers sensor inaccuracies. *Left:* the real object. *Right:* the reconstructed point cloud. Figure taken from Vasquez-Gomez et al. (2017) (© 2017 Springer).

In mobile robotics, building a model of the scene, i. e., the exploration, can be reduced to 2D. In the context of mobile robotics, frontier-based approaches are very popular. A frontier is defined as the border between unknown and known environment (Yamauchi, 1997). It is also crucial to take some safe space for the robot into account while exploring the area (Gonzalez-Banos and Latombe, 2002). However, an exploration strategy, which is limited to 2D, is not suitable for manipulation actions, since important elements of a scene can be missed. This includes, for example, a table-top scenario.

Another issue is that possible views are highly dependent on the robotic system. This can be due to the fact that the view is unreachable (Torabi and Gupta, 2011) or the position of the camera is inaccurate (Vasquez-Gomez et al., 2017). Other constraints for sensor planning have been addressed as well (Tarabanis et al., 1995; Yu and Gupta, 2004). For humanoids, the kinematic constraints are especially important since a humanoid needs to utilize the whole-body to be able to reach more view positions. Other works take explicitly the kinematic constraints of a humanoid robot into account when dealing with the NBV problem (Stasse et al., 2008; Foissotte et al., 2009). In humanoid and mobile robots, the NBV differs compared to static robotic arms, since the next view also needs to overlap the current view in order to allow for a registration. A reason for this is that, the camera position derived purely from forward kinematics is imprecise. The position error when moving the camera has been addressed by Vasquez-Gomez et al. (2017) for an industrial robot manipulator. Figure 2.10 displays experimental results of Vasquez-Gomez et al. (2017).

Suppa et al. (2004) present a physical space exploration for industrial eye-in-hand systems, namely the Kuka KR 16 robot. The exploration is sensor-based and maximizes knowledge about the configuration space while minimizing the number of views. The work also provides a comparison of update rules of the scene model.

For humanoids robots, not only an object model but also the scene is important. The goal of the NBV approach distinguishes between object modeling or scene exploration. Torabi and Gupta (2011) present a system for a 6 Degrees of Freedom (DoF) industrial robotic arm, that incorporates both modeling and exploration. After each scan the NBV is computed with a different goal to explore the scene and thus allow the robot to reach poses collision-free. Kriegel et al. (2013) use a utility function to balance between exploration and object modeling. In their work, the authors utilize both a volumetric as well as a surface based approach to represent the scene and the objects. Table 2.3 outlines a comparison of NBV approaches.

In general, most of the Next-Best-View approaches use a sampling strategy to determine possible views. These views are then evaluated in a second step, i. e., by computing the information gain. In general terms, the expected information gain for a view is defined as the change of information entropy

$$\mathcal{IG}(v) = H(v) - H(v^{t-1}) \ , \tag{2.1}$$

where v is the evaluated view and v^{t-1} the current view and $H(\cdot)$ the total information gain. $H(v)$ can be predicted by summing the changes in the entropy of each voxel. The idea of many NBV algorithms is to select the view which maximizes the function $\mathcal{IG}(v)$ and therefore add as much information to the model as possible. For the information gain, the work follows the notation of Isler et al. (2016). In many works, the basic idea to predict the information gain for a volumetric representation can be mainly reduced to count the number of newly observed voxels in a view. The number of unknown voxels can be obtained using a ray casting method for a possible view candidate. Thus, the predicted information gain for a view can be formulated as

$$\mathcal{IG}(v) = \sum_{\forall r \in \mathcal{R}_v} \sum_{\forall x \in \mathcal{X}_r} \mathcal{I} \ , \tag{2.2}$$

where $\mathcal{R}_v$ are all possible rays in view v and $\mathcal{X}_r$ the set of traversed voxels by the current ray r. Now the formulation of the information gain $I(x)$ can model

Publication				
Author	Year	Platform	Approach	Purpose/Task
Foissotte et al.	2009	HRP-2	Optimization algorithm	Object modeling
Kriegel et al.	2013	Kuka KR 16	Utility function	Object modeling and scene exploration
Potthast and Sukhatme	2014	PR2	Information gain	Scene exploration
Monica et al.	2016	Comau SMART SiX	Information gain and saliency	Spatial attention
Isler et al.	2016	Kuka youBot	Information gain	Object modeling
Xu et al.	2016	PR2	Attention	Object identification
Vasquez-Gomez et al.	2017	PatrolBot	Utility function	Object modeling
Daudelin and Campbell	2017	Kuka youBot	Information gain	Object modeling
Monica and Aleotti	2018a	Comau SMART SiX	Saliency and information gain	Object modeling
Oßwald and Bennewitz	2018	NAO (Simulated)	Information gain	Scene coverage
Liu et al.	2019	UR5	Attention	Affordance exploration
Monica et al.	2019	NAO	Information gain	Scene exploration

Table 2.3.: Comparison of Next-Best-View approaches based on their task, target platform and task.

different aspects to further improve the choice of the NBV. The choice of the NBV and therefore the formulation of the information gain depends on the tasks. Common tasks in robotics are object modeling or scene exploration.

Potthast and Sukhatme (2014) argue that the likelihood of observing an unknown voxel decreases as more unknown voxel are traversed. They introduce a more general approach for estimating the NBV. Their probabilistic framework is designed for cluttered environments and directly reasons about the unknown space. The unknown space is obtained by using a point cloud based representation of the scene. Both the laser scanner as well as the RGB-D camera of a PR2 robot are used to evaluate the approach. For humanoid robots other aspects have to be considered as well. This can be done by modeling the NBV with a utility function. Vasquez-Gomez et al. (2014) model a utility function, which also considers a required overlap of the view to support the registration of the view and penalizing movements of the robotic system. In robotic systems, it is important to also consider the costs to reach a view. Besides that, the estimated consumed power can be integrated over the whole movement of a humanoid robot (Oßwald et al., 2017).

In their work, Isler et al. (2016) present different information gain formulations for an object modeling task by a mobile robot. The authors evaluated their proposed formulation with respect to each other and as well as against previous work of Kriegel et al. (2015) and Vasquez-Gomez et al. (2014). The formulations vary with respect to the proximity and spatial location of voxels. Their system was evaluated using a synthetic model dataset and on a Kuka youBot with six DoF. For evaluation, the surface coverage, the normalized robot motion and the entropy were used. The accuracy of the reconstruction result was not qualitatively assessed. Figure 2.11 displays experimental results of the work published in Isler et al. (2016).
Based on the work of Isler et al. (2016), Daudelin and Campbell (2017) consider object probabilities by modeling the probability of a voxel belonging to the object being scanned. Their approach modifies the work of Isler et al. (2016) and limits the sampling to an area near frontier regions. Frontier regions are defined as the border between unknown and known space. Hence, the probability of a voxel belonging to an object is independent of the current view and can be computed once for each NBV algorithm iterations. The system was also evaluated in simulation as well as using a Kuka youBot. Figure 2.12 displays experimental results of the work published in Daudelin and Campbell (2017).

Figure 2.11.: NBV approach for object modeling as presented in Isler et al. (2016) (© 2016 IEEE). *Left:* Scene of the experiment. *Middle:* Point clouds of the reconstruction result. *Right:* Voxelized representation of the object reconstruction result.

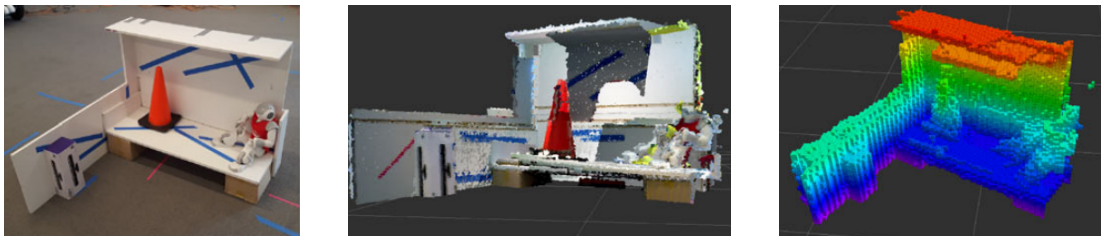


Figure 2.12.: Experiment results of Daudelin and Campbell (2017) (© 2017 IEEE). *Left:* Scene of the experiment. *Middle:* Point clouds of the reconstruction result. *Right:* Voxelized representation. The NBV system is based on the work of Isler et al. (2016).

Besides object modeling tasks, the NBV can also be used to cover a scene. Oßwald and Bennewitz (2018) present a GPU based NBV system for humanoid robots that attends user defined region of interests. Here, a crucial factor is how to balance the costs for reaching a view and the information gain. Evaluation has been performed in simulation. Figure 2.13 shows parts of the experiments. Similar work was done by Monica et al. (2019) where a humanoid robot NAO utilizes body movement primitives. For robot localization an external tracking system was used.

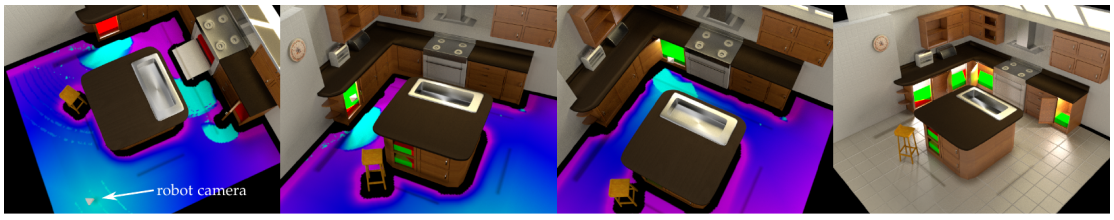
Another important aspect when designing a NBV system is the termination of the algorithms. The importance of the self-termination criteria has been stressed by Torabi and Gupta (2011). Typical approaches limit the total number of views. However, this is not compelling since the algorithm always has to visit the specified number of views. When dealing with object modeling the NBV planning terminates if the surface of the object does not contain any unknown patches.

In general, the evaluation is computationally expensive. The kinematic constraints of a humanoid robot can be exploited when pruning the number of

views before evaluation (Oßwald et al., 2017). Here, the inverse reachability maps are used to quickly check if the robot is able to reach the view pose. Isler et al. (2016) remove already visited view candidates. Other approaches (Oßwald and Bennewitz, 2018; Monica et al., 2016) implement a GPU version. Another way to deal with this issue is to utilize hierarchical ray casting (Vasquez-Gomez et al., 2014). The basic idea is that all view candidates are evaluated with fewer rays first and subsequently only the view candidates with the highest score are subjected to further evaluation while the number of rays is increased. Finally, the last remaining view candidates are evaluated with all possible rays. According to Vasquez-Gomez et al. (2014) this leads to a drastic speed-up, i. e., 20 times more performance to the standard case. Other approaches, such as Daudelin and Campbell (2017), use a look-up table to store the intermediate result for already computed positions.

Since then, several approaches for the NBV have been suggested. A comparison and benchmark of popular NBV approaches for object modeling is presented in Karaszewski et al. (2016). Interestingly, no NBV method outperformed the others. Other NBV algorithms focus on change detection. Monica et al. (2016) propose a NBV algorithm based on large scale point clouds. Their work is based on Connolly (1985) and Banta et al. (2000). After an initial scan of the environment, their saliency-based approach tracks relevant changes caused by human manipulation. These changes are detected by activity saliencies using a Gaussian mixture model to track a human subject. Points of Interest (PoIs) are computed from the possible changed regions. A NBV planning algorithm is then executed to determine the next position of the robot manipulator.

Monica and Aleotti (2018b) present a NBV planning system using a surfels-based representation. The space between unknown and known surfels is de-



(a) Example utility map at start (b) First exploration step (c) Second exploration step (d) Intermediate result after 7 steps

Figure 2.13.: The Next-Best-View approach explores user-defined regions of interest. The first image shows the simulated environment with an overlay of the utility map. User-defined regions of interest are visualized in red. The NBV approach is described in Oßwald and Bennewitz (2018) (© 2018 IEEE).

Figure 2.14.: Parts of an NBV experiments. *Left*: the scene. *Middle:* saliency map of point clouds segments. *Right:* 3D volumetric representation. Image taken from Monica and Aleotti (2018a) (© 2018a Springer).

noted as frontiers. View candidates are evaluated using the total area of visible frontier surfels. The authors argue that this representation has several advantages over a voxelized representation since the data is stored in a more compact way. PoIs are given as input to the algorithm. With respect to runtime the method shows a significant speed-up. However, this can be explained by the fact that, volumetric approaches are quite slow as they need to reason over all the unknown space if not limited to frontier regions as in Daudelin and Campbell (2017). Meißner (2020) uses spatial relations among objects to speed-up object search.

2.3.2. Visual Attention

Selecting the next view pose can also be attention driven. While the NBV problem is a top-down approach, i. e., clearly task driven, the view can be also selected based on visual stimuli as a bottom-up, i. e., purely driven by sensor input. In humans, visual saliency plays a crucial rule to control our gaze (Koch and Ullman, 1987). These aspects can be also transferred to robots, where it is important to focus on relevant regions as well. Bajcsy and Campos (1992) also pointed out that

> *"the perceptual system must be selective or it will suffer from an overflow of information."* (Bajcsy and Campos, 1992, p. 32)

In robotics, attention systems are often inspired by attention mechanisms discovered in humans, such as Itti et al. (1998) or Treisman and Gelade (1980). For an extensive review of visual attention approaches, the reader is referred

to the surveys in Frintrop et al. (2010) and Borji and Itti (2013) and more recently, Potapova et al. (2017) and Nguyen et al. (2018). The survey by Frintrop et al. (2010) stems more from a cognitive perspective, whereas the survey by Potapova et al. (2017) reviews work not considered in previous studies (Frintrop et al., 2010; Chen et al., 2011) with the focus on 3D visual attention in both human and robot vision. The survey further covers different areas of expertise in biological vision and neurophysiology, computer vision, as well as robotic vision. Many visual cues can be considered when shifting the attention of the robot. However, attention can also include multimodal cues, such as acoustic sensory processing capabilities (Schauerte, 2016). Despite that visual information is the most important sensory cue. In robotics, 3D information is central when interacting with the environment. Therefore, this section mainly focuses on 3D visual attention and foveated attention for robotic systems since this is especially important for humanoid robots. The section will also include relevant 2D based approaches.

Interestingly, humans can shift the attention without moving the eye. Attention shifts by moving the eye are called overt attention. In contrast to that are covert attention systems. An early visual attention system for humanoid robots was developed by Vijayakumar et al. (2001). The overt visual attention system is based on the visual flow and was developed for the humanoid robot DB with 30 DoF. The attention system utilizes both peripheral and foveal vision and is biologically inspired. A saliency map is generated by observing the optical flow and a WTA network is used to determine the next view direction. The approach uses only 2D features to shift the attention and during robot movements the image processing of the cameras is stopped. Simply discarding the images during motion is quite a common approach. Interestingly, the authors acknowledge the link to gaze stabilization, which is discussed in Chapter 5 of this thesis. Walther et al. (2005) show the effects of attention in object recognition tasks. Notably, that spatial attention improves the performance of object learning and recognition in cluttered scenes. Regions are selected using a WTA approach based on saliency values, which are computed with a multi-scale feature extraction process. An Inhibition of Return (IOR) is used to attend new regions. Welke (2011) proposes an inhibition of return mechanism that allows to generate a sequence of gaze directions.

Frintrop (2006) presents an attention system, called VOCUS, which uses both top-down and bottom-up cues to select regions of interest in images. Bottom-up cues are based on visual features such as color, orientation or contrast. Com-

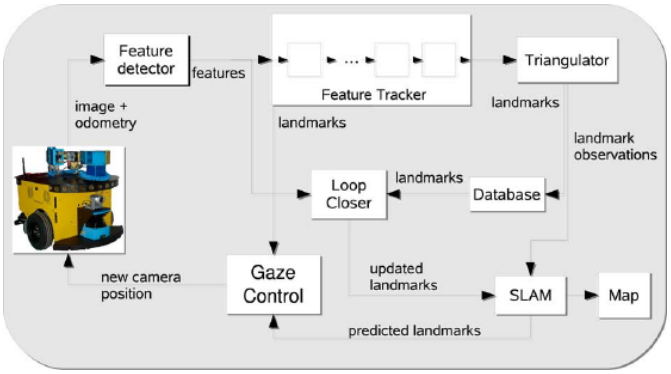


Figure 2.15.: A gaze control mechanism to support SLAM. The system support exploration of unknown areas and the redetection of landmarks to improve localization. Figure taken from Frintrop and Jensfelt (2008) (© 2008 IEEE).

pared to the iNVT by Itti et al. a different color space is used. These features are similar to the ones developed by Itti and Koch. Top-down cues include the current tasks. Both cues are aggregated into a single saliency map and the highest saliency is selected, i. e., a WTA approach. The regions of interest can then be used to support an object recognition system. In Frintrop and Jensfelt (2008), the VOCUS system has been used in combination with a behavior driven SLAM system for mobile robots. The system implements an attention driven approach to redetect landmarks for loop-closure while also exploring the scene. Regions of interest are either based on salient regions or on prediction landmarks in the scene. The method is outlined in Figure 2.15.

Other visual attention approaches are designed for object discovery (Horbert et al., 2015; Garcia et al., 2015).

Attention can also be used in combination with a NBV approach. The work of Monica et al. (2016) detects hand motions to shift the camera view. The work requires an initial scan of the environment. Xu et al. (2016) present a 3D attention approach and a NBV system. The system is based on 3-D recurrent attention model. The system allows for online identification of objects and consists of two levels of attention. The system is evaluated in on the PR2 robot with an eye-in-hand RGB-D camera. When addressing the NBV problem, the saliency can also be used when determining the next view. Monica and Aleotti (2018a) use saliency cues to filter the candidates for the NBV evaluation. The point cloud is segmented using Locally Convex Connected Patches (LCCP) and a saliency value is computed for each segment. View candidates are then ranked by the given saliency value and only the highest scores are considered for further eval-

Publication		Utility Function		
Author	Year	Information Gain	Path Costs	Task
Potthast and Sukhatme	2014	✓	-	-
Xu et al.	2016	✓	-	-
Oßwald et al.	2017	✓	✓	-
Oßwald and Bennewitz	2018	✓	✓	✓
Monica et al.	2019	✓	-	-
This approach		✓	✓	✓

Table 2.4.: Comparison of relevant Next-Best-View approaches for humanoid robots.

uation. Thereby, the number of candidates that are considered for evaluation can be drastically reduced. Figure 2.14 shows experimental results from the proposed method.

2.3.3. Summary

This section overviewed relevant active vision approaches. Active vision is controlling the camera purposefully to improve the current perception and is required for many robotic applications. An important application for active vision is the determination of the Next-Best-View (NBV) for scene exploration. An overview of the most important next-best-view approaches is provided in Table 2.4.

Methods addressing the NBV problem iteratively determine a view using a utility function, which evaluates views. For object modeling and scene exploration tasks, the volumetric information gain is a central element for planning. Besides the information gain, some methods also consider the path costs to reach a view. The NBV problem is mainly addressed in industrial robotic applications, but is now more and more common in humanoid robots, where it is much

more complex. For humanoid robots it is important to include anthropomorphic aspects in the system design. In particular, this includes the reachability checks and the costs to reach a view. The majority of NBV planning systems for humanoid robots focused on the modeling of the utility function rather than on the execution on a real robot system.

2.4. Gaze Control and Stabilization

As highlighted in the animate vision concept (Ballard, 1991) gaze control is a crucial aspect in robotics. Gaze control is either to attend a new region of interest, to fixate a moving target, or to stabilize the gaze.
Methods on identifying regions of interest have been addressed in the previous sections. This section focuses on switching and stabilization of the gaze, which is required to enable perception during motion. Gaze stabilization is especially important during locomotion and while shifting the gaze. During the eye and head movement, self-motion stimuli are often simply ignored by discarding the data (Vijayakumar et al., 2001).

Gaze stabilization methods are often inspired by human head and eye stabilization strategies replicating human reflexes. In humans, gaze stabilization for the eyes is mainly governed by the Vestibulo-Ocular Reflex (VOR) and the Optokinetic Reflex (OKR) (Miles, 1998). The VOR generates compensatory eye-movements to counter head movements. Head movements are detected by the acceleration measured in the vestibular system, a sensory system located in the human inner ear. The OKR stabilizes the view with eye movements to cancel the retinal slip, i. e., minimize the optical flow in the image. In robotics systems, an Inertial Measurement Unit (IMU) can mimic the human vestibular system to implement the VOR (Corke et al., 2007). The retinal slip can be directly computed from the optical flow in the image (Farnebäck, 2003) to implement the OKR. Both reflexes can be complementary since the reflexes have different stabilization goals. The VOR compensates for fast movements while the OKR compensates for slower ones (Schweigart et al., 1997). Similar to the VOR, the Vestibulo-Collic Reflex (VCR) stabilizes the head in humans.
Shibata and Schaal (2001) present a learning system to combine both the VOR and the OKR for the humanoid robot DB. The work only considers eye movements. Since gaze includes both head and eye movements, the head can also

support the stabilization. In humans, this is mainly done by the Vestibulo-Collic Reflex, which is similar to the VOR. Kryczka et al. (2012) present a method for head stabilization based on inertial measurement data. The system is evaluated on the KOBIAN humanoid robotic platform. Gaze stabilization modalities are often biologically inspired. This includes, for example, an integrated eye and head stabilization framework for the iCub platform inspired by cerebellar theories (Vannucci et al., 2016). The work coordinates the Vestibulo-Collic Reflex (VCR), the Vestibulo-Ocular Reflex (VOR), and the Optokinetic Reflex (OKR).

Other methods, compensating for self-induced perturbations only, rely purely on kinematics information (Kryczka et al., 2012), (Roncone et al., 2014), and (Habra and Ronsse, 2016). These methods use an internal model of the robot and using Inverse Kinematics (IK) (Habra and Ronsse, 2016; Roncone et al., 2016) to predict the position of the cameras while also computing head motions to compensate for the self-induced motions. The idea behind these methods is to intercept the motor commands which are then applied to a simulated robot model in order to predict and correct the next head position. New motor commands are then generated for the head to keep the visual frame stable. Habra and Ronsse (2016) propose a feed-forward gaze stabilization controller based on copies of motor commands. The inverse Jacobian defined by the gaze stabilization controller is relaxed by minimizing the optical flow. Furthermore, a fast method to approximate the optical flow using the robot's kinematics is derived. The approach was recently evaluated in simulation using the active head of the humanoid robot ARMAR-4 (Asfour et al., 2013). Gaze stabilization methods based on the IK are quite efficient since they can effectively compensate perturbations with a feed-forward controller. However, they can only compensate for self-induced perturbations. To this end, Habra et al. (2017) present a bio-inspired system that combines VOR and OKR with an IK method. The system is based on the reafference-principle (von Holst, 1954; von Holst and Mittelstaedt, 1950) and removes self-induced motions from sensor values, i. e., the optical flow and the head rotational velocity measured by an IMU. Thereby, eye stabilization reflexes are only invoked if the perturbation is not induced by the robot and therefore cannot be compensated by an inverse kinematics method. The system is evaluated in the humanoid robot ARMAR-III and experiments have been later extended in Habra et al. (2017) to the humanoid robot ARMAR-4.

Indeed, gaze stabilization and gaze control are intertwined. Roncone et al. (2016) designed a gaze control architecture allowing head stabilization and object tracking by executing saccadic eye movements on the iCub robot. Nonetheless, the system only allows for a single object to be tracked and does not support attention shifts based on the task acuity, which are required for a more complex scenario like grasping.

Other robotic applications include object tracking. Ude et al. (2003) integrated foveal and peripheral vision to track moving objects. Once a new area of interest is selected, the robot directs its gaze towards it and the object is subjected to a more detailed analysis. The system detects events to trigger saccadic eye motions. After a saccade, the robot starts pursuing the area of interest within the high-resolution foveal region. Similar work was studied by Omrčen and Ude (2010). The authors realized an object tracking controller for a Karlsruhe Humanoid Head (Asfour et al., 2008) using a virtual joint. In Ude and Asfour (2008), the authors exploit the properties of an active humanoid vision system to construct an effective object recognition system, where wide angle views were used to search for objects, direct the gaze towards them and keep them in the center of narrow-angle views. Milighetti et al. (2011) present a system for the Karlsruhe Humanoid Head. The system uses a Kalman filter approach to predict the trajectory of a moving target.

2.4.1. Summary

Methods for gaze control and gaze stabilization were presented in this section. Both coordinate the gaze. Gaze stabilization and gaze control mechanisms are not only present in humans, but also in almost any animals with visual perception.

Gaze control is a crucial concept in humanoid robotics due to the anthropomorphic design. Gaze control includes stabilization, tracking, and attention switching. These are several aspects of a humanoid gaze control system that need to be considered. Gaze stabilization methods are required to improve visual perception and to reduce image blur. An overview of the most important gaze stabilization approaches is provided in Table 2.5. Typically, gaze stabilization methods either mimic human inspired stabilization reflexes, such as the Vestibulo-Ocular Reflex, or are based on an internal model. In Chapter 5 of this

Publication		Gaze Stablization Modalities				Active Vision	
Author	Year	Eyes		Head	Both	bottom-up	top-down
		VOR	OKR	VCR	IK		
Shibata and Schaal	2001	✓	✓	-	-	-	-
Kryczka et al.	2012	✓	✓	-	✓	-	-
Vannucci et al.	2016	✓	✓	✓	-	-	-
Roncone et al.	2014	-	-	-	✓	-	-
Habra and Ronsse	2016	-	-	-	✓	-	-
Roncone et al.	2016	✓	-	-	✓	✓	-
Habra et al.	2017	✓	✓	-	✓	-	-
This approach						✓	✓

Table 2.5.: Comparison of relevant approaches for gaze stabilization.

thesis, methods for gaze stabilization are linked to an active vision method. Gaze stabilization modalities enable visual perception during motion.

2.5. Discussion

This chapter discussed the most relevant work with respect to this thesis. Therefore, a definition of the most relevant concepts was given. Here, the active vision paradigm is essential since an active vision method considers the robot as an active observer. Active vision methods purposefully manipulate the camera pose and camera parameters in order to improve the current perception. Thus, active vision is linked to a task or is stimulus driven. The approach of this thesis was classified as active vision method.
After the concepts were introduced, the chapter reported on related work. In particular, this includes work on scene modeling and understanding, active

vision methods, especially NBV planning, as well as gaze control and stabilization modalities. Alternatives to scene modeling, as presented in Chapter 3, were also presented. Similar to that, active vision methods that deal with the NBV problem are classified and positioned with respect to the approach in Chapter 4. Visual attention methods, relevant for humanoids, are also listed. Gaze stabilization is necessary to enable visual perception while moving as shown in Chapter 5. Hence, this chapter also gave an overview of gaze control and gaze stabilization methods.

3. Semantic Scene Representation

Knowledge about the environment is of utmost importance for both humans and robots when interacting with the scene. This knowledge includes not only the geometric structure of the scene, but also the physical relationships among objects and possible actions that can be performed.
For example, a humanoid robot exploring a partially collapsed building must autonomously create a scene representation. The actual scene can largely differ from the expected situation. Therefore a robot's mission cannot be planned in advance and the scene representation including interaction possibilities has to be extracted autonomously. Some tasks even require an understanding of the effects of actions, especially in search and rescue operations. If a person is buried under a steel girder, the steel girder must be lifted first.

This chapter introduces a 3D scene model including a semantic representation of unknown environments. The scene model is extracted from the robot's current view, i. e., from visual sensor data streams and forms the basis for the subsequent Next-Best-View (NBV) planning method described in Chapter 4. The geometric primitive detection method presented here corresponds to the *Segmentation then fitting* according to the classification of Kaiser et al. (2018). Methods presented here include results from Kaiser et al. (2015a), Grotz et al. (2017b), Kartmann et al. (2018), and Grotz et al. (2019).
Figure 3.1 depicts the data flow and different processing steps. The data flow starts with visual sensor data streams followed by segmentation. For each segment, geometric primitives are extracted. The geometric primitives build a geometric representation of the scene. The outline of the chapter follows the processing steps for semantic scene perception. Hence, Section 3.1 begins with a method to build a geometric scene model from the current view. Section 3.2 completes the semantic scene representation by describing methods for scene understanding and the extraction of affordances. Section 3.3 presents a spatio-temporal approach to fuse results from multiple views. Section 3.4 describes evaluation of the methods and Section 3.5 summarizes the results.

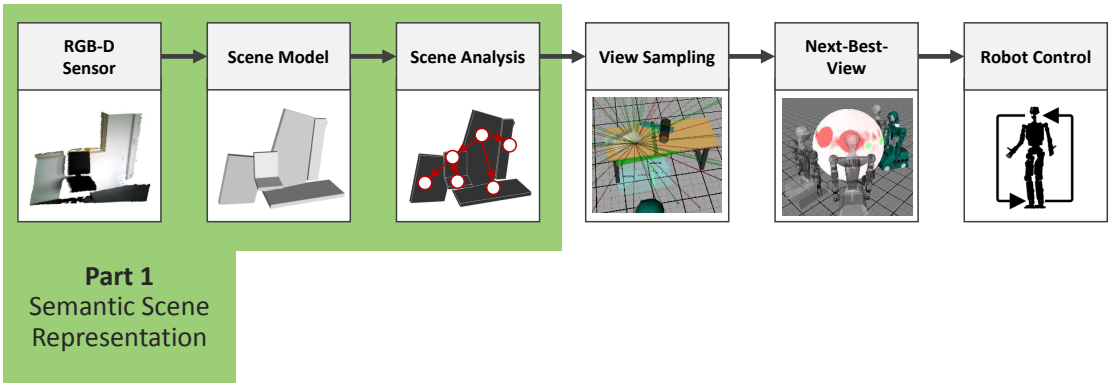


Figure 3.1.: Semantic scene representation of unknown environments from visual sensor data streams. A 3D geometric scene model is constructed. Spatial relations, including support and stability, are inferred among objects. Figure adapted from Grotz et al. (2017b).

3.1. Geometric Scene Modeling

A 3D scene representation is of utmost importance for a humanoid robot. Without it a robot cannot interact with the scene. In a first step, spatial information must be encoded into a 3D model of the scene. The scene model has to cover all necessary geometric details. At the same time, the scene model must allow a compact and memory efficient representation, as the resources of the robot, e. g., CPU and memory, are limited. In this approach, the objects are therefore abstracted using geometric primitives. The set of considered geometric primitives consists of cuboids, cylinders, planes and spheres. Using geometric primitives has two advantages. They preserve geometric properties of the scene while at the same time they allow for a simplification of the input data. Thus, a geometric primitive based representation requires less memory as well as an easier analysis in subsequent steps. The scene model interprets the acquired visual sensor data. Since the field of view is limited, multiple views must be considered. To aggregate data from multiple views, the robot must also register the current view with respect to previous views. The following assumes that the pose of each view is known in relation to the world coordinate system. The problem of registering views with respect to each other is briefly described in Appendix A.5. A temporal parameter is introduced to differentiate between two different point clouds that were recorded at different times. I In the following, a superscript notation denotes this temporal information. (i) The superscript $\cdot^t$ denotes data at time t, e. g., $\mathcal{P}^t$ denotes the point cloud captured at time t. (ii) The superscript $\cdot^{1,t}$ denotes aggregated data from time

1 to t, e. g., $\mathcal{P}^{1,t}$ denotes the registered point cloud from time 1 to time t. The temporal parameter t is omitted in the following if it can be derived from the context.

To create a geometric model, the point cloud $\mathcal{P}$ captured from the current view is decomposed by segmentation into plausible and disjoint regions $\mathcal{P}_i$, such that

$$\mathcal{P} = \bigcup_i \mathcal{P}_i \qquad \text{and} \tag{3.1}$$

$$\mathcal{P}_i \cap \mathcal{P}_j = \emptyset \;\; \forall i, j \;\; i \neq j \;\; . \tag{3.2}$$

The additional segmentation step divides the visual sensor data into plausible and disjoint parts. These can already indicate possible objects in the scene. The process of segmentation is described in Appendix A.6. Subsequent steps therefore process already partitioned data. For subsequent steps, the segmentation not only reduces the input size but also enables the parallel execution. For example, the extraction of the geometric primitives can be accelerated significantly by parallel execution. After the segmentation step, the geometric primitives ψ_i are iteratively fitted to each segment in the point cloud. For this purpose, a method based on Random Sample Consensus (RANSAC) (Fischler and Bolles, 1987) estimates the model parameters from a set of points. A customized approach uses the geometric primitive fitting methods provided by the widely used Point Cloud Library (PCL) (Rusu and Cousins, 2011). Formally, this step allows the decomposition of a segmented point cloud acquired at time t into a set of geometric primitives

$$\Psi^t = \{\psi_1^t, \ldots, \psi_{m_t}^t\} \;\; . \tag{3.3}$$

For further processing, each geometric primitive ψ_i is linked to an inlier point cloud $\mathcal{P}_{\psi_i} \subset \mathcal{P}_{s_i}$ of the corresponding segment s_i. Inliers are points that match the fitted model of the geometric primitive. As emphasized in Schnabel et al. (2007), it is important to distinguish between the segment and the inlier point cloud. For each $\mathcal{P}_{\psi_i}$, the RANSAC based approach randomly selects a certain minimum number of points to determine the model parameters as a new model hypothesis. The points associated with the segment s_i are then tested to see if they belong to the fitted model, i. e., if they count as inliers. Fitting the model is done for each geometric primitive shape and the model with the highest number of inliers is selected as best model. Finally, the model inliers $\mathcal{P}_{\psi_i}$ are re-

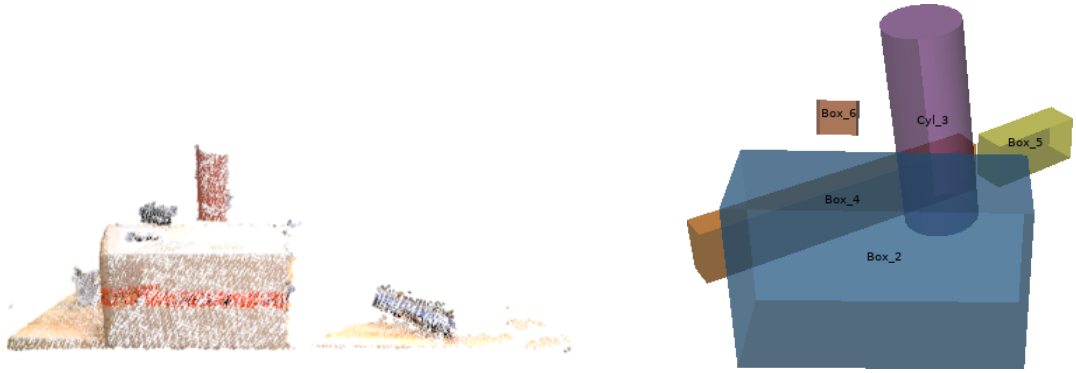


Figure 3.2.: An input point cloud and the fitted geometric primitives.

moved from the point cloud $\mathcal{P}_{s_i}$ and the step is repeated until no more geometric primitives are found for the segment s_i or the cardinality of the remaining points in $\mathcal{P}_{s_i}$ is smaller than a threshold. This means that each fitted geometric primitive must have at least τ_{min} inlier points. Similar to that, the number of maximum points is limited by τ_{max}. For more details on the approach see Kaiser et al. (2015a) and Grotz et al. (2017b). For each geometric primitives ψ_i geometric properties like the oriented bounding box OBB(ψ_i) are calculated from the inlier point cloud $\mathcal{P}_{\psi_i}$. Figure 3.2 shows an example scene model from a point cloud. Algorithm 1 outlines the fitting of geometric primitives as described in Kaiser et al. (2015b).

3.2. Semantic Scene Modeling

Scene understanding of unstructured environments plays an essential role in the autonomous planning and execution of grasping and manipulation tasks. Therefore, the geometric information has to be interpreted first to allow for further reasoning and better understanding. Section 3.2.1 presents a method that represents spatial relationships among the extracted geometric primitives. Spatial relations indicate how a geometric primitive is located in the scene in relation to another geometric primitive. Similar to that, Section 3.2.2 discusses a method that analyses stability and support relations between the extracted geometric primitives. These relations are used by the active vision method developed in the following chapter. Finally, Section 3.2.3 recaps the automatic extraction of interaction possibilities, so called affordances, from geometric primitives. Both, reasoning over support and stability as well as the autonomous extraction of affordances are not the major focus of this thesis. Therefore, this

Algorithm 1: Geometric Primitive Extraction

Data: Segmented Point Cloud $\mathcal{P}_1 \dots \mathcal{P}_n$, Minimum number of inliers τ_{min}, Maximum number of inliers τ_{max}

Result: Geometric Primitives Ψ

$\Psi \leftarrow \emptyset$;

foreach $\mathcal{P}_i \in \mathcal{P}_1 \dots \mathcal{P}_n$ **do**

  while $|\mathcal{P}_i| \in [\tau_{min}, \tau_{max}]$ **do**

    $\psi_{\text{plane}} \leftarrow \text{RANSAC}_{\text{plane}}(\mathcal{P}_{s_i})$;

    $\psi_{\text{cylinder}} \leftarrow \text{RANSAC}_{\text{cylinder}}(\mathcal{P}_{s_i})$;

    $\psi_{\text{sphere}} \leftarrow \text{RANSAC}_{\text{sphere}}(\mathcal{P}_{s_i})$;

    $\psi_{\text{best}} \leftarrow \arg\max_{\psi \in \{\psi_{\text{plane}}, \psi_{\text{cylinder}}, \psi_{\text{sphere}}\}} |\mathcal{P}_\psi|$;

    if $\psi_{best} = \emptyset$ **then**

      break;

    $\psi_{\text{new}} \leftarrow \text{EuclideanClustering}(\mathcal{P}_{\psi_{\text{best}}})$;

    ComputeGeometricProperties(ψ_{new});

    $\Psi \leftarrow \Psi \cup \psi_{\text{new}}$;

    $\mathcal{P}_i \leftarrow \mathcal{P}_i \setminus \mathcal{P}_{\psi_{\text{new}}}$;

return Ψ

section recaps the most relevant definitions for the NBV approach. Methods are described in detail in Kartmann et al. (2018) and in Kaiser and Asfour (2018).

3.2.1. Spatial Reasoning

Spatial relations among extracted geometric primitives are encoded using a graph structure. The approach presented here is similar to work of Schnabel et al. (2008). Formally the spatial relations are modeled with a graph $\mathcal{G} = (V, E)$. The vertices V of graph map the set of extracted geometric primitives $\psi_i \in \Psi$. The edges E model the spatial relations among the geometric primitives. First, a distinguished geometric primitive is selected as the root of the graph. Typically, this is defined as the ground floor plane or the table-top. Specifically, the root ρ of the graph is identified as the geometric primitive with the lowest

Figure 3.3.: The humanoid robot ARMAR-III in a kitchen environment. The robot does not have any knowledge about the scene. The goal of the experiment is to lift a box from the table. Since both sides of the box are required for executing the manipulation action more than one view is required. Hence, multiple-views are registered with respect to each other and the geometric primitives are fitted.

height in the scene with respect to the robot and thus

$$\rho = \arg\min_{\psi \in \Psi} \text{height}(\psi) \ . \tag{3.4}$$

Second, the remaining geometric primitives are iteratively checked for intersections and added to the graph of the scene representation. Formally, an edge $e = (\psi_i, \psi_j)$ is added to the graph if

$$\text{OBB}(\psi_i) \dot{\cap} \text{OBB}(\psi_j) \neq \emptyset \ , \tag{3.5}$$

where $\text{OBB}(\cdot)$ denotes the associated Oriented Bounding Box (OBB). When calculating the OBBs, both $\text{OBB}(\psi_i)$ and $\text{OBB}(\psi_j)$ are slightly extended by $\varepsilon > 0$ in order to account for perceptual inaccuracies. Formally, the intersection operator $X \dot{\cap} Y$ for two sets X and Y is defined as follows:

$$X \dot{\cap} Y = \{x \in X \mid \exists\, y \in Y : \|x - y\|_2 \leq \varepsilon\} \ . \tag{3.6}$$

In practice, the parameter ε depends on the sensor and setting $\varepsilon = 5\,\text{cm}$ yields good results. To speed up the process, geometric primitives outside the current

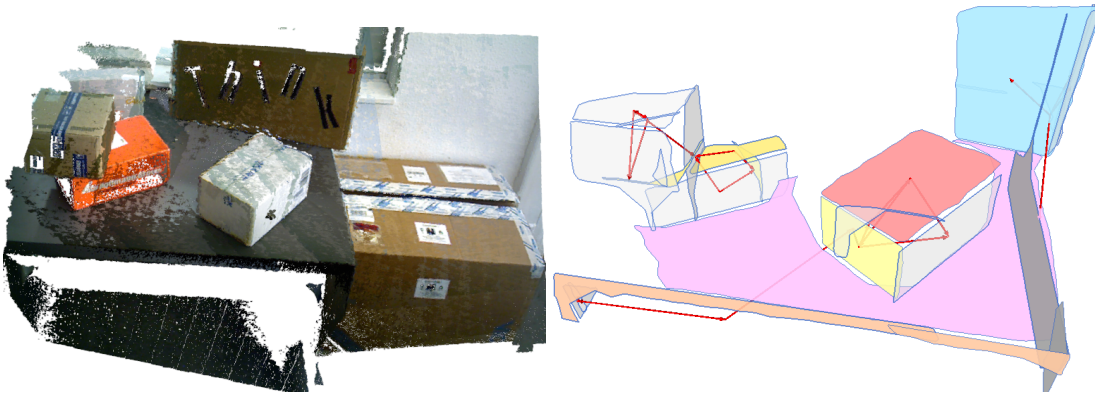


Figure 3.4.: *Left*: The input point cloud registered from multiple-views. The scene is shown in Figure 3.3. *Right*: Extracted geometric primitives including spatial relations. Figures taken from Grotz et al. (2017b) (© 2017b IEEE).

field of view are discarded. Thus the number of computationally exhaustive intersection tests is reduced. Figure 3.4 depicts an exemplary set of geometric primitives the obtained scene graph. The scene is shown in Figure 3.3.

3.2.2. Stability and Support Reasoning

Support relations among geometric primitives are based on the spatial extent of the geometric primitives. The notation and definition of the support relations follow the work of Mojtahedzadeh et al. (2015) and Kartmann et al. (2018). Similar to the spatial reasoning, the support graph $\mathcal{G}_s = (V, E)$ spans the geometric primitives. However, support relations are not symmetric. Therefore, the support graph is a directed graph. Again, the vertices V map the set of geometric primitives ψ. Furthermore, the edges E model possible support relations among the objects. Two geometric primitives $\psi_i, \psi_j \in \Psi$ are denoted as $\texttt{SUPP}(\psi_i, \psi_j) \iff \psi_i$ supports ψ_j. In other words, ψ_j loses its motionless state if ψ_i is removed, then ψ_i supports ψ_j.

To determine the support relations, all pairs of geometric primitives in contact are identified. For each pair (ψ_i, ψ_j) in contact, a separating plane is constructed at the contact points. A support relation edge is added to the graph. The edge starts from the geometric primitive below the constructed separating plane and ends at the geometric primitive above the supporting plane.
In the following, a geometric primitive is considered *unstable* if there is no support for the geometric primitive. Vice versa a geometric primitive ψ_j is consid-

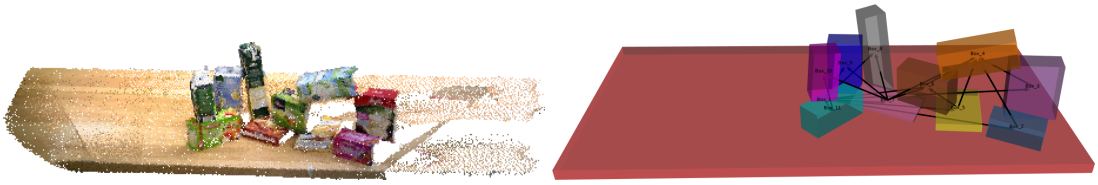


Figure 3.5.: *Left:* Input point cloud. *Right:* Fitted geometric primitives and extracted support relations based on the input point cloud.

ered *stable* if a path exists from ψ_j to ρ with $(\psi_j, \cdot) \dots (\cdot, \rho) \in E$. A major reason why a geometric primitive is considered unstable is because of an incomplete scene model. This means that a supporting geometric primitive is missing or the scene is not yet fully explored. Therefore, support relations are of particular interest as a hint for incomplete data of the scene model. Figure 3.5 shows an example for extracted geometric primitives and the support graph.

3.2.3. Affordance Extraction

The geometric scene model can be further enriched with affordances. Affordances are interaction possibilities that are associated with an object in relation to the abilities of an agent. The term *affordances* was coined by Gibson (1979). Hence, only relevant parts are given in the following. For more details about the method and implementation, the reader is referred to Kaiser et al. (2016) and Kaiser (2018). Affordances are detected by sampling the surface and geometric properties of the geometric primitives with respect to the end-effector of the robot. A detailed formulation is outside the scope of work. More formally, an affordance is defined as a function which maps an end-effector pose $x \in SE(3)$ to a belief expression $d \in \mathcal{D}$. These interaction possibilities include affordances such as (i) support, (ii) lean, (iii) grasp, (iv) hold, (v) push, and (vi) lift. Affordances are automatically extracted by sampling end-effector poses using the geometric primitives. The geometric properties, such as the normal, size, and orientation of the fitted geometric primitives are the basis to determine the affordances in the scene. Figure 3.6 visualizes an examples scene with an extracted *lift* affordance for the humanoid robot ARMAR-III.

Figure 3.6.: An executed bimanual lift affordance for the humanoid robot ARMAR-III. *Left:* An external view of the experiment. *Right:* Geometric primitives and extracted affordances are visualized and selected by an operator. Figure taken from the experiment of Grotz et al. (2017b) (© 2017b IEEE).

3.3. Spatio-Temporal Fusion of Geometric Primitives

Two sets of geometric primitives Ψ^{t_1} and Ψ^{t_2} resulting from multiple individual extraction processes at times t_1 and t_2 cannot be considered as entirely independent of each other. For example, a consistent geometric primitive that is too large to be visible within the robot's field of view and thus results in an arrangement of multiple smaller extracted geometric primitives. Parts of the larger geometric primitive, however, can be detected from different views. To this end, this section describes a fusion step for geometric primitives extracted from multiple views.

Let $\Psi^{1,t}$ denote the set of existing primitives from previous views. Then, given the scene graph of the geometric primitives $\Psi^{1,t}$, other geometric primitives Ψ^{t+1} from the current view are fused with previous ones. The Jaccard index (Jaccard, 1912) is a measure for to express the similarity of two sets. The Jaccard index is defined as:

$$\mathrm{J}(\psi_i, \psi_j) = \frac{|\mathcal{P}_{\psi_i} \cap \mathcal{P}_{\psi_j}|}{|\mathcal{P}_{\psi_i} \cup \mathcal{P}_{\psi_j}|} = \frac{|\mathcal{P}_{\psi_i} \cap \mathcal{P}_{\psi_j}|}{|\mathcal{P}_{\psi_i}| + |\mathcal{P}_{\psi_j}| - |\mathcal{P}_{\psi_i} \cap \mathcal{P}_{\psi_j}|} \ , \tag{3.7}$$

Algorithm 2: Spatio-temporal Fusion of Geometric Primitives.

Data: New primitives Ψ^{t+1}, Previously fused primitives $\Psi^{1,t}$

Result: Set of fused primitives $\Psi^{1,t+1}$

$\hat{\Psi}^{1,t} \leftarrow$ FrustrumCulling($\Psi^{1,t}$)

foreach $\psi \in \hat{\Psi}^{1,t}$ **do**

  foreach $\varphi \in \Psi^{t+1}$ **do**

    if $OBB(\varphi) \cap OBB(\psi) = \emptyset$ **then**

      continue;

    if *not CompareModelParameters*(φ, ψ) **then**

      continue;

    if $\varphi \subset \Psi$ **then**

      $\Psi^{t+1} \leftarrow \Psi^{t+1} \setminus \varphi$;

    else if $Inlier(\varphi, \psi) > \lambda_o \wedge\ Inlier(\psi, \varphi) < \lambda_p$ **then**

      $\Psi^{t+1} \leftarrow \Psi^{t+1} \setminus \varphi$;

    else if $Inlier(\psi, \varphi) > \lambda_o \wedge\ Inlier(\varphi, \psi) < \lambda_p$;

    then

      $\hat{\Psi}^{1,t} \leftarrow \hat{\Psi}^{1,t} \setminus \psi$;

$\Psi^{1,t+1} \leftarrow \hat{\Psi}^{1,t} \cup \Psi^{t+1}$;

return $\Psi^{1,t+1}$;

where $|\cdot|$ denotes the cardinality, e. g., here the number of points. Since geometrical properties such as the OBB are already available the following equation is used instead

$$\text{Inlier}(\psi_i, \psi_j) = \frac{|\mathcal{P}_{\psi_i} \cap \text{OBB}(\psi_j)|}{|\mathcal{P}_{\psi_i}|} \quad . \tag{3.8}$$

Algorithm 2 outlines the spatio-temporal fusion of geometric primitives. As soon as a new set of geometric primitives Ψ^{t+1} is computed, the current geometric primitives $\Psi^{1,t}$ are filtered according to the robot's current field of view. Thereby, geometric primitives, which are not visible in the field of view, are discarded and thus the overall computing time is reduced. To begin with, each pair of geometric primitives $\psi \in \Psi^{1,t}$ and $\varphi \in \Psi^{t+1}$ is tested if both overlap at all. During this step, it is tested whether OBB(ψ) and OBB(φ) are intersecting. If two geometric primitives ψ and φ are intersecting, then Equation 3.8 is used to evaluate the geometric similarity. This ratio expresses the degree of coverage between two primitives. When the degree of coverage between ψ and φ exceeds

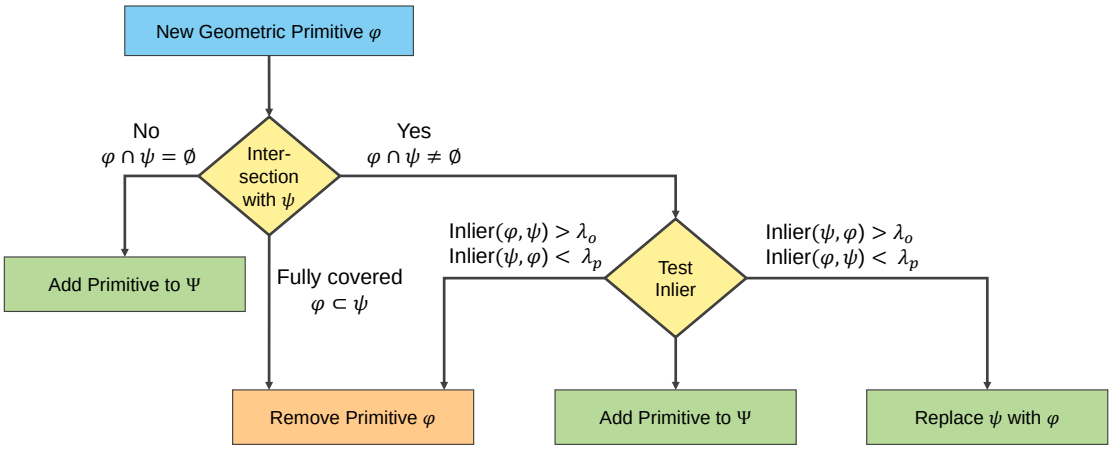


Figure 3.7.: Workflow of the spatio-temporal fusion of geometric primitives.

a threshold value λ_o, the covered geometric primitive is removed from the new set of geometric primitives and only the large geometric primitive is kept. To prevent partially overlapping geometric primitives from being removed, the number of inliers between φ and ψ is also checked. The threshold value is denoted by the lower bound λ_p. This is due to the fact that the OBB can be larger than the actual geometric primitive. For this work, the following thresholds are used $\lambda_o = 0.7$ and $\lambda_p = 0.3$. Figure 3.7 visualizes the workflow for fusing a new geometric primitive $\varphi \in \Psi^{t+1}$.

3.3.1. Support Graph Combination

For the scene model and semantic scene understanding, the information from multiple views must be combined. The following approach for merging support graphs is used in Chapter 4. A simple strategy to combine the support graph information is to extract the scene model and support relations after every single view from a global point cloud $\mathcal{P}^{1,t+1}$, containing all the registered previous views. In the following, this case will be denoted as *Point Cloud only (PC)*. To speed-up the process, the global point cloud is first downsampled because the geometric primitive fitting step is computationally expensive and the calculation time scales with the input size of the point cloud. Overall, there are two major arguments against computing the support graph from a global point cloud: (a) the total runtime is significantly increased, and (b) the registration of the views may not be optimal, so that the RANSAC based geometric primitive fitting might fail to find all inliers for a geometric shape. Therefore the geometric primitives and the support graph are iteratively extracted from

	Point Cloud		Support Graph Combination		
t	Input	Registered	PC	SG	PC + SG
1	$\mathcal{P}_1$	$\mathcal{P}^{1,1} = \mathcal{P}_1$	$\mathcal{G}_s^1$	$\mathcal{G}_s^1$	$\mathcal{G}_s^1$
2	$\mathcal{P}_2$	$\mathcal{P}^{1,2} = \mathcal{P}_1 \cup \mathcal{P}_2$	$\mathcal{G}_s^2$	$\mathcal{G}_s^{1,2}$	$\mathcal{G}_s^1 \cup \mathcal{G}_s^2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	$\mathcal{P}_n$	$\mathcal{P}^{1,n} = \bigcup_{i=1}^{n} \mathcal{P}_i$	$\mathcal{G}_s^n$	$\mathcal{G}_s^{1,n}$	$\mathcal{G}_s^{n-1} \cup \mathcal{G}_s^n$

Table 3.1.: Schema of support graph combination methods.

each view. The result then is fused with a global consistent support graph $\mathcal{G}_s^{1,t}$. Given the support graph $\mathcal{G}_s^{t+1} = (V^{t+1}, E^{t+1})$ extracted from the current view, the vertices, i. e., the geometric primitives, are first matched with vertices of the existing support graph $\mathcal{G}_s^t = (V^t, E^t)$. The approach is similar to the fusion of geometric primitives as presented in Section 3.4.2. For this purpose, shape, position, and orientation as well as the extent of the geometric primitives are compared. Similarly, it is tested if an edge $e = (\mathtt{A}, \mathtt{B}) \in E^{t+1}$ already exists in E^t. If $e \notin E^t$ then it is added as a new edge. For each edge e, the number of times the edge e has been extracted and matched is counted. The number of matches is denoted as $occ(e)$. This allows the NBV algorithm to later validate the existence of the support relations, which have been visible only a few times. This support graph fusion method is called *Support Graph (SG)*.
Finally, these two methods are combined into a method called *(PC + SG)*. For this method, the point cloud is registered as proposed in the first approach *(PC)*. The support graph $\mathcal{G}_s^{t+1} = (V^{t+1}, E^{t+1})$ is extracted from $\mathcal{P}^{1,t+1}$ and merged with $\mathcal{G}_s^t$ as described in the second approach *(SG)*. Table 3.1 outlines the different support graph combination methods.

3.4. Evaluation

This section presents real world experiments to assess the quality of the scene model. The following experiments were performed using the humanoid robots ARMAR-III (Asfour et al., 2006) and ARMAR-6 (Asfour et al., 2019b). In the

first case, the RGB-D sensor ASUS Xtion Pro is used for the experiments, while in the second case, the Carmine Primesense 1.09 is used. Both work in a similar way. The major difference is that the ASUS Xtion Pro has a larger sensing distance and the Carmine Primesense has a lower minimum sensing distance. Robot and sensor systems are described in Appendix B.

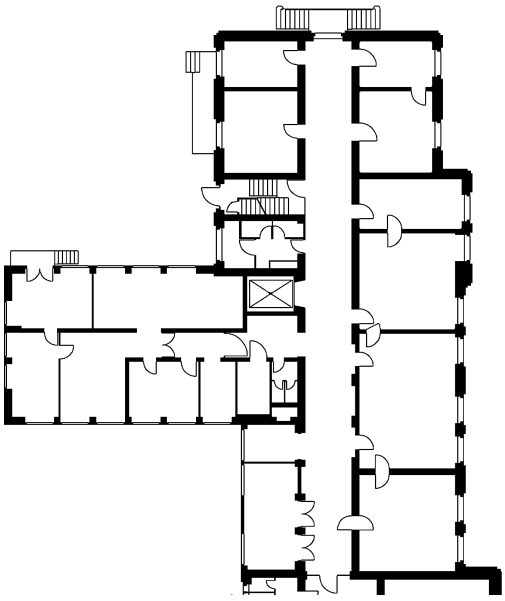
3.4.1. Qualitative Evaluation

For the qualitative evaluation of the geometric primitive fusion, the humanoid robot ARMAR-6 is located in a corridor. The scene was chosen because of the many branches and corners, and thus several views are necessary to cover the entire area. The robot moves within the corridor and constantly shifts the gaze. Here ElasticFusion (Whelan et al., 2017) is used to register multiple views. To increase the robustness of the registration, the robot's forward kinematics and odometry are included. The Locally Convex Connected Patches (LCCP) method (Stein et al., 2014) is used to segment the point cloud. Due to the building architecture, the scene mainly consists of planes. The entrance corridor has a total length of $33.5\,\mathrm{m}$ and a total width of $2.78\,\mathrm{m}$. Besides that, there is a second corridor branching off from the main corridor, which is not visible from the entrance and thus requires the robot to change the position. Additionally, many embellishments around the room entrances hide parts of the walls. Figure 3.8 shows parts of the scene, the floor plan, the registered views as point cloud, and the extracted geometric primitives.

3.4.2. Spatio-Temporal Primitive Fusion Experiment

To assess the presented method on semantic scene perception, the accuracy of the spatio-temporal fusion of geometric primitives is investigated by comparing primitives fused from real RGB-D camera data with manually labeled ground truth.
The ground truth is created by using a simulated version of the environment. The simulation uses CAD models of the kitchen elements. A simulated RGB-D sensor with the same camera parameters as the sensor mounted on ARMAR-III captures the scene. Since the pose of the camera is known exactly, the additional registration step is not necessary, which typically induces noise to the

(a) Scene



(b) Floor layout

(c) Fused Point cloud

(d) Extracted Geometric primitives

Figure 3.8.: Qualitative evaluation of the geometric primitives fitting with the humanoid robot ARMAR-6. ARMAR-6 is scanning a large corridor. The point cloud was segmented using the LCCP method. Geometric primitives are extracted and fused iteratively.

estimated camera pose. The captured ground truth point cloud G is then manually labeled and thus $G = \{G_1 \dots G_m\}$, with mutually disjoint segments G_i:

$$G_i \cap G_j = \emptyset \;\; \forall i \neq j \;\; . \tag{3.9}$$

For each ground truth segment G_i, the OBBs are computed. Given OBB(G_i), the extracted geometric primitive $\psi_j \in \Psi$ is selected that maximizes the number of inliers as specified by the inlier ratio defined in Equation 3.8 is denoted as $\psi(G_i)$. Formally,

$$\psi(G_i) = \arg\max_j \text{Inlier}(G_i, \Psi_j) \;\; . \tag{3.10}$$

Equation 3.10 is now used to quantify the quality of extracted geometric primitives Ψ compared to a ground truth segmentation G. The value is then averaged and denoted as InlierIndex(G, Ψ) with

$$\text{InlierIndex}(G, \Psi) = \frac{1}{m} \sum_{i=1}^{m} \text{Inlier}(G_i, \psi(G_i)) \;\; . \tag{3.11}$$

Since Equation 3.11 only captures over-segmentation, the Jaccard index is used to determine overlap. Therefore, the formulation in Equation 3.7 is adjusted, summed and averaged with

$$\text{OverlappingIndex}(G, \Psi) = \frac{1}{m} \sum_{i=1}^{m} \max_j \frac{|G_i \dot{\cap} \Psi_j|}{|G_i \cup \Psi_j|} \;\; . \tag{3.12}$$

An outline of the experimental setup is shown in Figure 3.9.

A total of 58 point clouds resembling a kitchen environment were recorded from different angles using the sensory equipment of the humanoid robot ARMAR-III. The robot is moved through the kitchen environment. Captured point clouds are sequentially processed by utilizing the approach presented in this chapter. Figure 3.10 plots the corresponding values of Equation 3.11 and Equation 3.12 for the geometric primitives aggregated from consecutive frames over time. Since the robot is constantly moving, and thus new parts of the scene become visible over time, the number of inliers increases monotonically until the scene is fully covered. Over time, both indices converge towards the maximum value of 1. This indicates that the result of the spatio-temporal fusion of iteratively detected geometric primitives eventually almost resembles the ground truth primitives. Figure 3.11 displays the number of removed and modified primitives over time in the same experimental setup. It can be seen that the number

of geometric primitive modifications drops after the scene is reasonably well covered by the aggregated scene model, indicating that the method for spatio-temporal primitive fusion eventually reaches a state of equilibrium, where additional views of the scene have only little influence on the aggregated geometric primitive model.

3.5. Summary

This chapter presented a method for a semantic scene perception of unknown environments using RGB-D sensor data. This is the first key component for the automatic perception of unknown environments.
The scene model is based on the detection of geometric primitives comprising planes, cylinders, spheres, and cuboids. These geometric primitives store the geometric information of the environment and thereby represent objects and elements in the scene. The geometric representation is then enriched with semantic information. Semantic information is inferred by the construction of a graph-based scene representation that takes neighborhood relations among geometric primitives into account. The spatial reasoning also makes it possible to identify physically plausible support relations among the geometric primitives, which are essential for the interaction with the scene.
Further, a method for spatio-temporal fusion of geometric primitives was shown. The method is needed when combining consecutive views, as it resolves inconsistency in the scene model and reduces computation time. Finally, the methods were evaluated on the humanoid robots ARMAR-III and ARMAR-6.

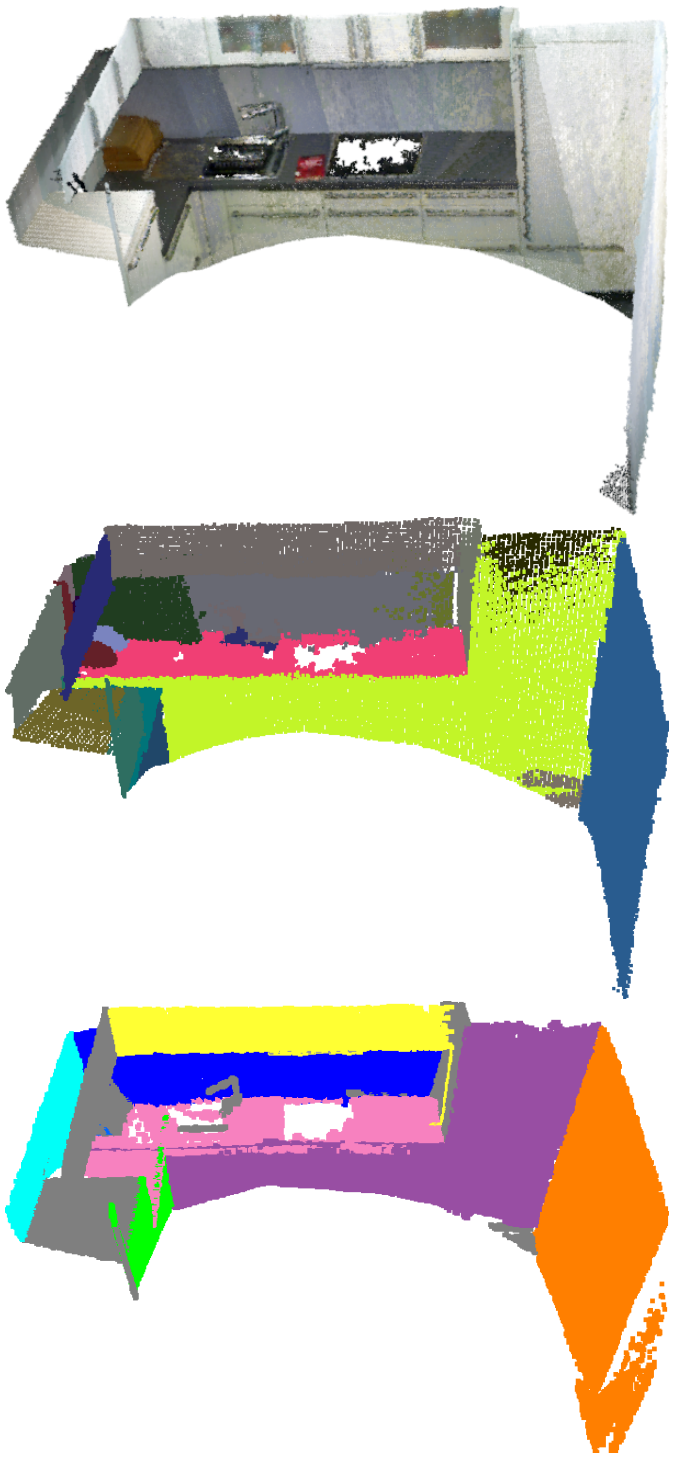


Figure 3.9.: Kitchen scenario as described in Section 3.4.2. *Top*: The image shows the registered result of the source point cloud. *Middle*: Multiple views are first registered and then geometric primitives are extracted. *Bottom*: Primitives are extracted iteratively from consecutive frames. Figures taken from Grotz et al. (2017b) (© 2017b IEEE).

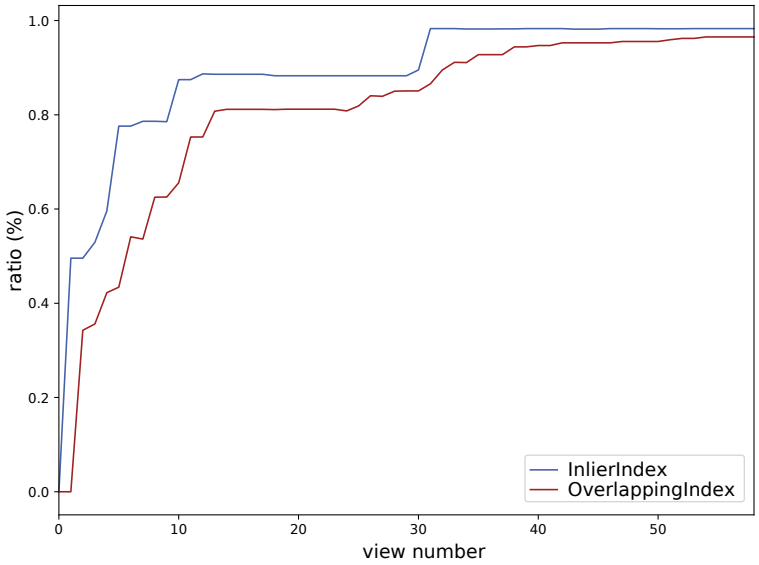


Figure 3.10.: The plot visualizes the inlier ratio (Equation 3.11) and the overlapping index (Equation 3.12) over time. After 30 views, most of the scene is mapped to geometric primitives. In the following views the scene model is refined as can be seen from the increasing *OverlappingIndex* value. Figure taken from Grotz et al. (2017b) (© 2017b IEEE).

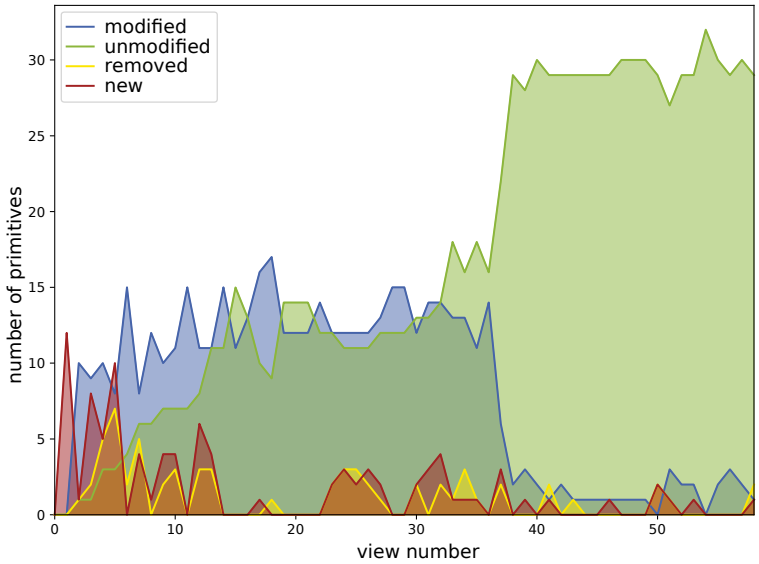


Figure 3.11.: Number of primitives modified, removed and added. After 40 views the scene is almost fully covered. Thus, only a few geometric primitives are added. Re-detected primitives are updated. Figure taken from Grotz et al. (2017b) (© 2017b IEEE).

4. Next-Best-View Planning

Extracting a complete scene model from a single view is not possible since relevant parts of the scene are hidden. Figure 4.1 illustrates an example, where more than one view is required. The essential part of the table-top is hidden, a common issue when dealing with a cluttered environment. This prevalent limitation is due to occlusions. Hence, it is crucial to actively control the camera, by changing the robot's position and gaze direction, in order to mitigate the effect of occlusion and to resolve inconsistencies in the scene model. What is visible and relevant in a view is one of the major questions an active vision system should answer. To overcome these limitations, this chapter describes an active vision system for automatic scene perception. The active vision system determines the Next-Best-View (NBV), which yields an improvement of the semantic scene representation. Therefore information from the scene model is considered. Figure 4.2 outlines the system architecture. The algorithm described here follows the idea of many NBV approaches, e. g., Connolly (1985), Banta et al. (2000), Vasquez-Gomez et al. (2017), Monica et al. (2016) or Oßwald et al. (2017), which divide view planning into two steps: (1) view sampling, and (2) view evaluation. The first step samples possible views, which are then subjected to further evaluation in the second step. The NBV is then defined as the view which scores best in the evaluation. The presented approach includes results of Sippel (2019) and Grotz et al. (2019).
The chapter is organized as follows. Section 4.2 describes the view sampling and presents a novel approach using support relations among objects as a hint. Section 4.3 deals with the view evaluation to determine the NBV with respect to the semantic scene model. The view evaluation balances between exploration and validation of the scene model using a utility function described in Section 4.3.3. The formulation of the utility function includes task-oriented measures derived from the semantic scene model. Section 4.5 presents the evaluation of the methods. Finally, Section 4.6 concludes the chapter.

Figure 4.1.: A table-top scenario. The robot cannot perceive the scene completely from the initial view due to occlusion. The robot has to change the position in order to extract a complete representation of the scene. *Left:* External view showing ARMAR-6 at the initial position before exploring the scene. *Right:* The semantic scene representation extracted from the initial view with occlusion. The semantic scene representation comprises of fitted geometric primitives and support relations in the scene. The contact points between objects, however, are not visible from the initial view.

4.1. View Representation

A view $v \in SE(3)$ specifies the coordinate system of the visual sensor, i. e., the center and the orientation with respect to the world coordinate system. The orientation of the camera coordinate system includes the robot's gaze direction as well. A common approach is to sample views on a sphere with fixed radius r around a center $\boldsymbol{p}$, e. g., as proposed in Monica et al. (2016). Without loss of generality, the sphere is centered at the origin of the world coordinate system, namely $\boldsymbol{p} = \mathbf{o} = (0, 0, 0)$. To move the center of the sphere to a Point of Interest (PoI), all coordinates are simply translated by the position of the PoI. The sphere $\mathbb{S}^2$ of radius r is given by

$$\mathbb{S}^2_r = \left\{ (x, y, z) = \boldsymbol{p} \in \mathbb{R}^3 \mid \|\boldsymbol{p}\| = r \right\}.$$

To reduce the sampling size, views are further subjected to the upper half space with $z > 0$, since it is silently assumed that the ground plane is $z = 0$. In this work, the radius r is set to $1.5\,\mathrm{m}$ to account for inaccuracies in the depth image. While the maximum sensing range for structured light sensors is much higher, the depth error is not linear with respect to the sensed distance. Once the sphere is constructed, views are sampled. Possible views are then defined as camera

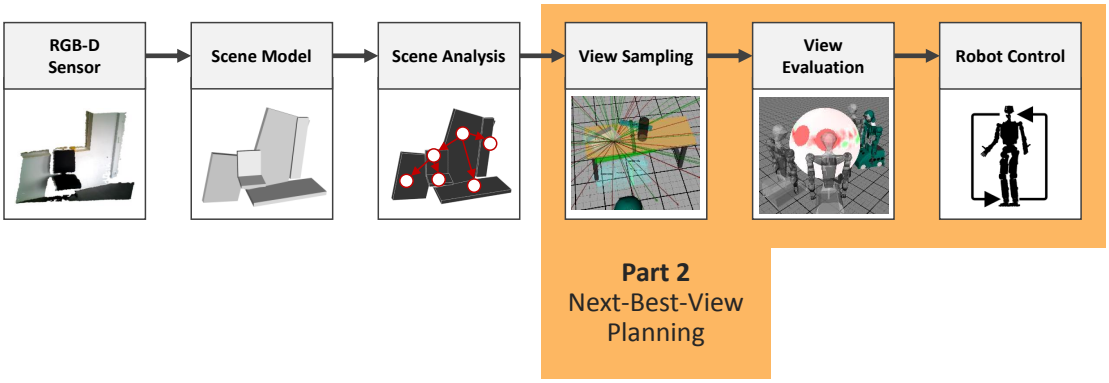


Figure 4.2.: The system architecture to determine the Next-Best-View. The semantic scene model is extracted from the current view as described in Chapter 3. Based on the physical plausible support relations views are sampled and evaluated using a utility function.

coordinate systems on the sphere with the view axis intersecting the center o of the sphere. Further, a rotation around the view direction is neglected as it does not contribute significantly to what is visible in the current view. Thus, for a view $v = (R, \boldsymbol{t})$ the rotation R is composed of $R = R_x \cdot R_z$, with R_x being a rotation around the x-axis and R_z a rotation around the z-axis respectively. Formally, the set of all possible views $\mathcal{V}$ given the previous constraints is defined as

$$\mathcal{V} = \left\{(R, \boldsymbol{t}) \mid R = R_x \cdot R_z \in SO(3), \boldsymbol{t} \in \mathbb{S}_r^2\right\} \subset SE(3) \ . \tag{4.1}$$

Evaluating all possible views in $\mathcal{V}$ is not possible and the space for sampling views has to be discretized. Discretizing the set affects the quality of the evaluated view slightly due to the opening angle of the visual sensor and the overlap between neighboring views, i. e., moving the sensor only slightly will only change slightly what is visible in the image. Formally, let $\mathcal{V} \subset SE(3)$ be the set of all sampled views. The Next-Best-View (NBV) is then defined as the view

$$\hat{v} = \arg\max_{v \in \mathcal{V}} u(v) \ , \tag{Next-Best-View}$$

which maximizes a utility function $u(\cdot)$, that models different aspects, such as information gain or path costs. The utility function is described in detail in Section 4.3.3. To reach the NBV $\hat{v}$, the view $\hat{v} = (R, \boldsymbol{t})$ is translated into a platform position and a gaze direction. The platform position (x, y) is given by the

projection

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \cdot \boldsymbol{t} \tag{4.2}$$

and the platform orientation α is given by

$$\alpha = -\frac{\pi}{2} + \arctan2\left(-t_2, t_1\right) \tag{4.3}$$

where t_i is the i-th element of vector $\boldsymbol{t}$. The range of $\arctan2\left(\cdot,\cdot\right)$ is $(-\pi, \pi]$. After the robot has reached the new platform position and orientation, the gaze direction is set to the camera orientation R using Inverse Kinematics (IK). Unreachable views are discarded.

4.2. View Sampling

A standard approach to sample views is to sample equidistantly on the view sphere (Monica et al., 2016; Banta et al., 2000). Depending on the utility function, the evaluation can be time consuming. One way to reduce the overall computation time, is to limit the number of sampled views.

4.2.1. Standard View Sampling

The easiest way to generate possible views is to sample equidistantly on each view sphere and to set the view direction to the center of the sphere. Given the radius r of the view sphere, possible views can be sampled using spherical coordinates (r, θ, φ), with θ being the polar angle and φ being the azimuthal angle, with $0 \leq \varphi < 2\pi$, $0 \leq \theta \leq \pi$, and $r > 0$. The spherical coordinates can then be translated to Cartesian coordinates with

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} r \cdot \sin(\theta) \cdot \cos(\varphi) \\ r \cdot \sin(\theta) \cdot \sin(\varphi) \\ r \cdot \cos(\theta) \end{pmatrix}. \tag{4.4}$$

Algorithm 3: Points of Interest Generation

Data: Support Graph $\mathcal{G}_s = (V, E)$, View sphere S, Previous view poses $\hat{v}_0, \ldots, \hat{v}_t$

$PoI \leftarrow \emptyset$;

foreach $\mathtt{A} \in V$ **do**

  $U \leftarrow \{\mathtt{B} \in V | (\mathtt{B}, \mathtt{A}) \in E\}$;

  if $U = \emptyset$ **then**

    $\theta \leftarrow$ GetLargestArc(S, $\hat{v}_0, \ldots, \hat{v}_t$);

    $x \leftarrow$ ComputeContactPoints(A, $\frac{\theta}{2}$);

    $PoI \leftarrow PoI \bigcup \{(x, s)\}$;

  foreach $\mathtt{B} \in U$ **do**

    $x \leftarrow \mathtt{A} + \frac{(\mathtt{B} - \mathtt{A})}{\frac{1}{2}||\mathtt{B} - \mathtt{A}||_2}$;

    $s \leftarrow$ ComputeSaliency(x);

    $PoI \leftarrow PoI \bigcup \{(x, s)\}$;

return PoI;

Thus, to sample k views linearly on the view sphere, the subset of possible views can be expressed as

$$x_{k,r} = \left\{ (r, \theta, \varphi) = (r, \frac{2\pi m}{k}, \frac{2\pi n}{k}) \mid 0 \leq \theta \leq \pi, 0 \leq \varphi < 2\pi, m, n \in \mathbb{Z}_0 \right\},$$

with k being the step size of the samples.

Instead of sampling views equidistantly, views can also be sampled randomly. To sample the view poses randomly, choose u_φ and u_θ to be standard uniformly distributed random variables, i. e., $u_\varphi, u_\theta \sim \mathcal{U}(0, 1)$. Given u_φ, u_θ, the azimuthal angle is then $\varphi = 2\pi u_\varphi$ and the polar angle is $\theta = \arccos(2u_\theta - 1)$. Using Equation 4.4, the spherical coordinates can be mapped to Cartesian coordinates.

4.2.2. Top-Down View Sampling

The following presents an approach that utilizes the semantic information to compute possible views instead of randomly sampling views on a sphere. This sampling strategy is a top-down approach since it uses higher level information. To this end, Points of Interest are identified in the scene and used to compute views. The semantic information and the support relations are used to generate PoIs in two different and independent steps. Algorithm 3 illustrates the approach. In a first step, for each unsupported object, PoIs are generated based on an object's extent and previous views. Therefore, the largest arc on the view sphere between previous positions of the robot is determined. The PoI is then the intersection of the object from the line of the middle of the largest arc to the object's center. The underlying concept is that each object must have at least one support edge due to gravity. A missing support for an object can be explained by the fact, that either the object itself is not fully visible or that other supporting objects are missing. A maximum saliency value, i. e. $s(v) = 1$, is used to account for further exploration of the object and its area. In a second step, points of interest are computed based on the edges between objects. Here, the idea is to consider a relation in the support graph as more stable depending on the number of times it has been observed. In this case, the saliency value $s(v)$ is computed as

$$s(v) = \lambda_s + (1 - \lambda_s) \cos\left(\frac{\pi}{2} \cdot \frac{occ(e_x)}{n}\right) \quad , \tag{4.5}$$

where e_x is the edge associated with the point of interest x, $occ(e_x)$ is the number of times the edge e_x was observed and n the number of total views and $\lambda_s \in [0, 1]$ is parameter to define the importance of the support edge validation.

The points of interest are then projected on to the view sphere to represent possible views of the robot. Similar to Grotz et al. (2017a), the saliency value is propagated to neighboring views on the sphere with a decreasing value. An example is shown in Figure 4.3. Occlusions are mitigated by checking if the line of sight between the PoI x and the projected PoI v is free. In case of occlusion the saliency value is inverted. Algorithm 4 lists the top-down view sampling approach.

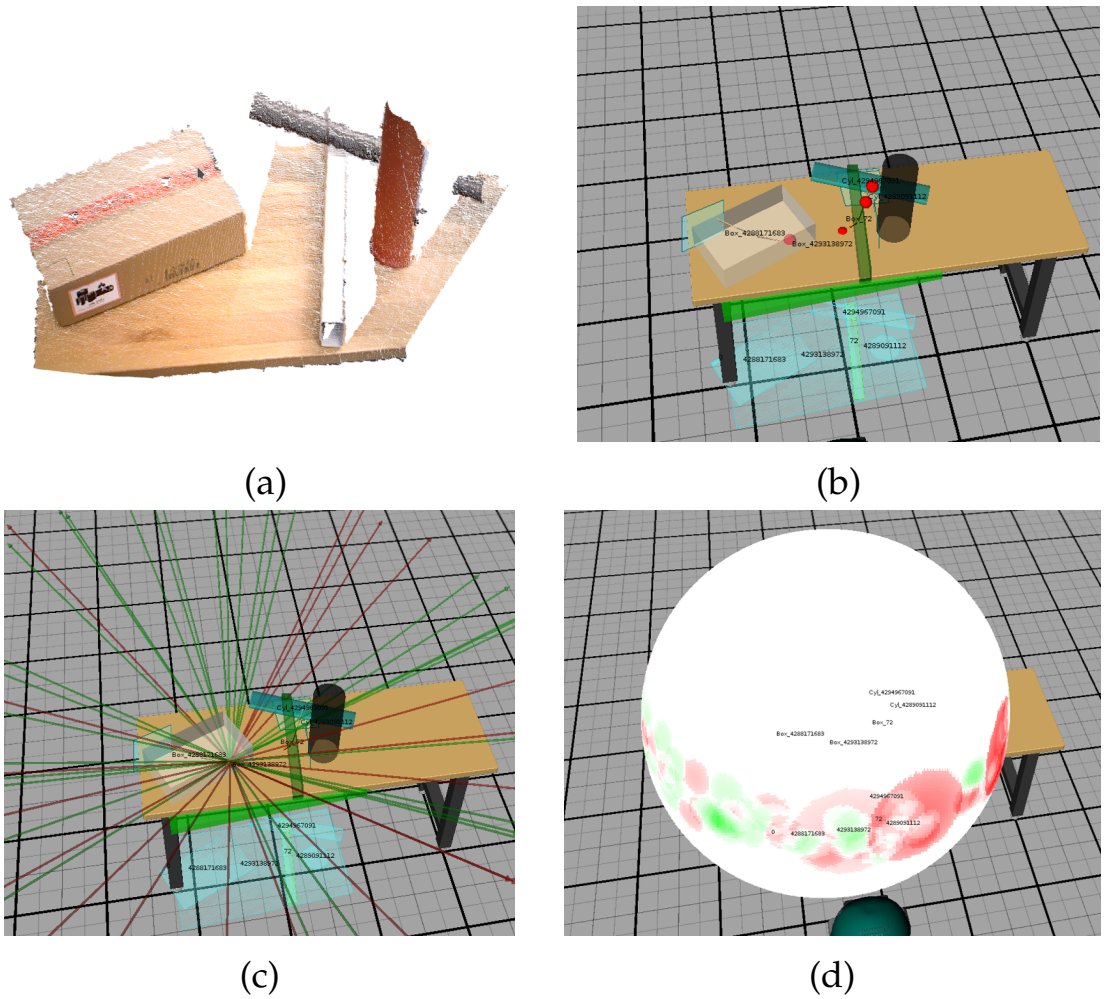


Figure 4.3.: Identified Points of Interest (PoIs) in the scene. These points are then projected to the view sphere. (a) shows the point cloud of the current view. (b) shows the generated PoIs. (c) visualizes the projection using ray casting. (d) shows the view sphere with the projected PoIs.

Algorithm 4: Top-down View Sampling

Data: Support Graph $\mathcal{G}_s$, Sphere Center c_x, Voxel Map V

$S \leftarrow$ CreateViewSphere(c_x);

$PoI \leftarrow$ GeneratePoI($\mathcal{G}_s$, S, ...);

foreach $(x, s) \in PoI$ **do**

    $v \leftarrow$ ProjectToSphere(x);

    $r \leftarrow$ SingleRayCast(x, v, V);

    if *IsIntersectionFree*(r, V) *and IsReachable*(v) **then**

        AddToViewSphere(v, S);

4.3. View Evaluation

Given the sample view poses as described in the previous section, the views have to be evaluated in order to determine the Next-Best-View. A typical approach is to predict the information gain, as described in the subsequent section. However, as noted by Vasquez-Gomez et al. (2014) relying only on the information gain does not consider the costs for the robot to reach the NBV. Hence, using a utility function the traveled distance and other costs need to be taken into account. In the following, two functions are described in order to evaluate a view: the predicted information gain and the path costs. Finally, both functions are combined and balanced in a utility function.

4.3.1. Predicted Information Gain

To quantify the quality of a sampled view v, the expected information gain is a popular approach. The expected information gain is the information an agent expects to gain when attending view v. In general, the expected information gain $\mathcal{IG}(v)$ is defined as the difference between the previous entropy $H(v)^{t-1}$ and the entropy $H(v)$ when attending the view, with

$$\mathcal{IG}(v) = H(v) - H(v)^{t-1}. \qquad \text{(Information Gain)}$$

Information gain formulations have been used for active SLAM systems (Thrun et al., 2005). Different formulations have been proposed to model the volumetric information gain. A comparison of different NBV formulations for object modeling tasks, is described for mobile robots in Isler et al. (2016) and for eye-in-hand robotic arms in Karaszewski et al. (2016). Here, the mathematical notation of Isler et al. (2016) is used. The volumetric information gain is predicted by casting rays from the view pose v and counting the number of unknown voxels. The predicted volumetric information gain for a view $v \in \mathcal{V}$ in general is expressed as

$$\mathcal{IG}(v) = \sum_{\forall r \in \mathcal{R}_v} \sum_{\forall x \in \mathcal{X}_r} \mathcal{I} \ . \tag{4.6}$$

Here, $\mathcal{X}_r$ denotes the set of all traversed voxels x by ray r. Typically, an occupancy map is used to allow for voxel counting. In this work, the *OctoMap* (Hornung et al., 2013) implementation is used. To speed-up ray casting a hierarchical approach as suggested in Vasquez-Gomez et al. (2014) is used. There-

fore, every 30th ray is considered. That reduces the number of rays to check from $640 \cdot 480 = 307200$ to 10240. Next, the views with the highest information gain are evaluated again, but this time without skipping a ray.

4.3.2. Path Costs

To reach a view $v \in SE(3)$, a collision-free path is computed using a Rapidly Exploring Random Tree (RRT) (Lavalle, 1998) based planner and the platform of the robot is moved along the path segments. The path planner is part of the robotics toolbox Simox (Vahrenkamp et al., 2013). Figure 4.4 shows an example of a planned path. The planned path consists of several tuples with platform position and orientation, i. e., $\text{path}_v = (p_0, \alpha_0), \dots (p_n, \alpha_n)$, where n is the path segment that corresponds to view v as specified in Equation 4.2 and Equation 4.3. The costs to reach the final position (p_n, α_n) are given by summing up the partial costs. Since the humanoid robots used for the experiment have a holonomic platform, rotational and translational movements can be executed at the same time. Therefore, the costs to reach an intermediate position of the path (p_i, α_i) considers rotation and translation independently. The former, i. e., translational costs, are given by Euclidean distance between the path segments, while the latter, i. e., rotational costs, are given by the angle difference. The angle difference of two angles α_{i-1} and α_i, is confined to $(-\pi, \pi]$ with

$$\Delta(\alpha_{i-1}, \alpha_i) = \arctan2\left(\sin(\alpha_{i-1} - \alpha_i), \cos(\alpha_{i-1} - \alpha_i)\right) \ . \tag{4.7}$$

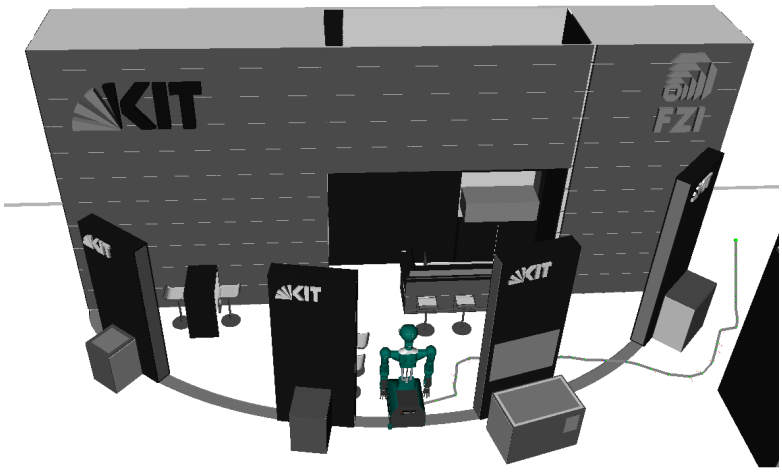


Figure 4.4.: Path planning example using a Rapidly Exploring Random Tree (RRT) based approach. The scene shows a CAD model of the exhibition booth at CeBIT 2018.

To compare both costs, the two are divided by the maximum velocity of the platform. For each path segment from (p_{i-1}, α_{i-1}) to (p_i, α_i) the maximum of the translational and rotational costs is used. Thus, the costs $d(p_{i-1}, \alpha_{i-1}, p_i, \alpha_i)$ are defined by

$$d(p_{i-1}, \alpha_{i-1}, p_i, \alpha_i) = \max\left(\frac{\|p_{i-1} - p_i\|_2}{\text{vel}_{lin}}, \frac{|\Delta(\alpha_{i-1}, \alpha_i)|}{\text{vel}_{rot}}\right) , \tag{4.8}$$

where vel_{lin} is the maximum linear velocity and vel_{rot} is the maximum rotational velocity respectively. Hence, $d(\cdot)$ provides an upper bound of the time to reach the next path segment (p_i, α_i). In order to reach a view v, in addition to the platform movements, the gaze has to be shifted as well. However, the time to set joint angles is significantly smaller compared to time to move the platform. Here, the time to shift the gaze is neglected as it does not contribute significantly to the costs. The total path costs $c(v)$ to reach a view v are thus defined as

$$c(v) = \begin{cases} \infty, & \text{if view } v \text{ is unreachable} \\ \sum\limits_{(p_i, \alpha_i) \in path_v} d(p_{i-1}, \alpha_{i-1}, p_i, \alpha_i), & \text{otherwise.} \end{cases} \tag{4.9}$$

4.3.3. Utility Function

Once a set of view poses is chosen, each view needs to be evaluated in order to select the Next-Best-View $\hat{v}$. A standard approach is to use the estimated volumetric information gain. However, the NBV does not depend solely on explored space and this approach is only an estimate. Different aspects, such as platform movement costs and task specific dependencies need to be considered as well. Vasquez-Gomez et al. (2014) mentions several constraints and aspects that a Next-Best-View system needs to consider. This includes, for example, the reachability of the view or the navigation distance. As already mentioned in Chapter 2, the NBV differs for humanoid robots compared to industrial eye-in-hand systems by uncertainty of the camera pose.

To evaluate the sampled view set, relevant aspects of the current task are modeled with a utility function (Vasquez-Gomez et al., 2017). Since the utility function covers different aspects, the parts of the utility function must be normal-

ized. Hence, the information gain is set to

$$\overline{\mathcal{IG}}(v) = \frac{\mathcal{IG}(v)}{\arg\max_v \mathcal{IG}(v) - \arg\min_v \mathcal{IG}(v)} \ , \tag{4.10}$$

and the path costs are set to

$$\overline{c}(v) = \frac{c(v)}{\arg\max_v c(v) - \arg\min_v c(v)} \ . \tag{4.11}$$

This normalizes both functions to $[0 \dots 1]$.

In this work, the NBV is determined using the following utility function u with

$$u(v) \colon \mathcal{V} \to \mathbb{R} \tag{4.12}$$

$$v \mapsto \overline{\mathcal{IG}}(v) - \lambda \cdot \overline{c}(v) \ . \tag{4.13}$$

Here, $\mathcal{IG}(v)$ models the predicted information gain and $c(v)$ are the costs of reaching view v. The weight λ balances between exploration and path costs. The utility function is similar to the function proposed in Oßwald and Bennewitz (2018). The only differences are that a different cost function has been used and both the costs and the information gain have been normalized. Normalizing is important since both measures have to be compared with each other.

4.4. System Architecture

The system architecture for the autonomous perception of unknown environments is shown in Figure 4.2. The system comprises several components. The components to extract a scene model from the current view are described in Chapter 3. For the determination of the Next-Best-View the two components named *View Sampling* and *View Evaluation* are of particular interest. The former component, is responsible for sampling views. Two approaches have been presented in this chapter and the interfaces allow for an easy extension. The latter component, is responsible for evaluating the views. Again, several methods are presented in this chapter. Figure 4.5 illustrates the workflow of computing a NBV. The system is designed in a modular way that allows reusing and exchanging of the components.

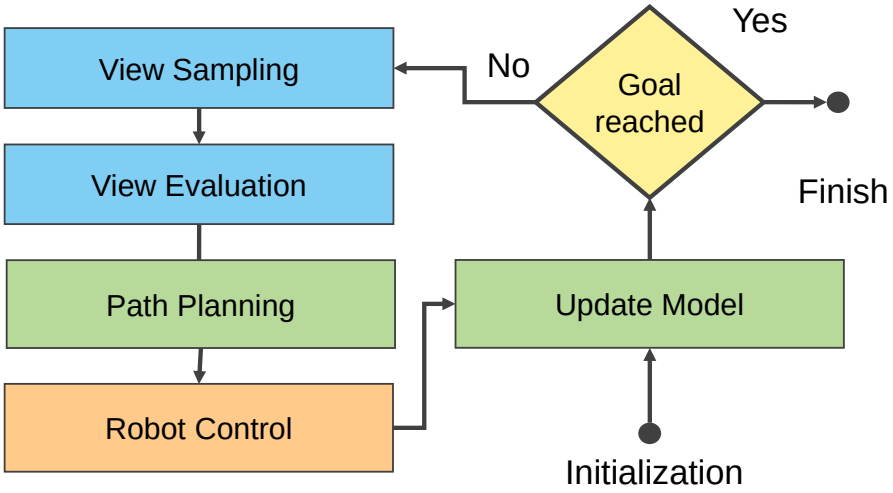


Figure 4.5.: Workflow of the NBV planning. Views are sampled, and evaluated. Once the robot reaches the NBV the scene representation is updated and termination criteria are checked.

Algorithm 5: Next-Best-View Planning

Data: Support Graph $\mathcal{G}_s$, Sphere Center c_x, Voxel Map V

$\mathcal{V} \leftarrow$ SampleViews();

foreach $(v, s) \in \mathcal{V}$ **do**

  $(R, \boldsymbol{t}) \leftarrow$ ComputePlatformPosition(v);

  $\mathcal{IG} \leftarrow$ PredictInformationGain(v);

  $p \leftarrow$ PathPlanning(v);

  $h \leftarrow$ ComputePathCosts(v);

$\hat{v} \leftarrow \arg\max_{v \in V} c(v)$;

return $\hat{v}$;

The active vision system terminates if the current view pose does not contribute significantly, i. e., $u(\hat{v}) < \lambda_{quality}$, or after n views are reached. Depending on the scene the total number of views is limited to n, e. g., $n = 10$. Algorithm 5 outlines the NBV planning.

4.5. Evaluation

Both the view sampling and as well as the view evaluation are qualitatively and quantitatively assessed in several experiments. Experiments are conducted using the humanoid robot ARMAR-6. Section B.2 gives an overview of the

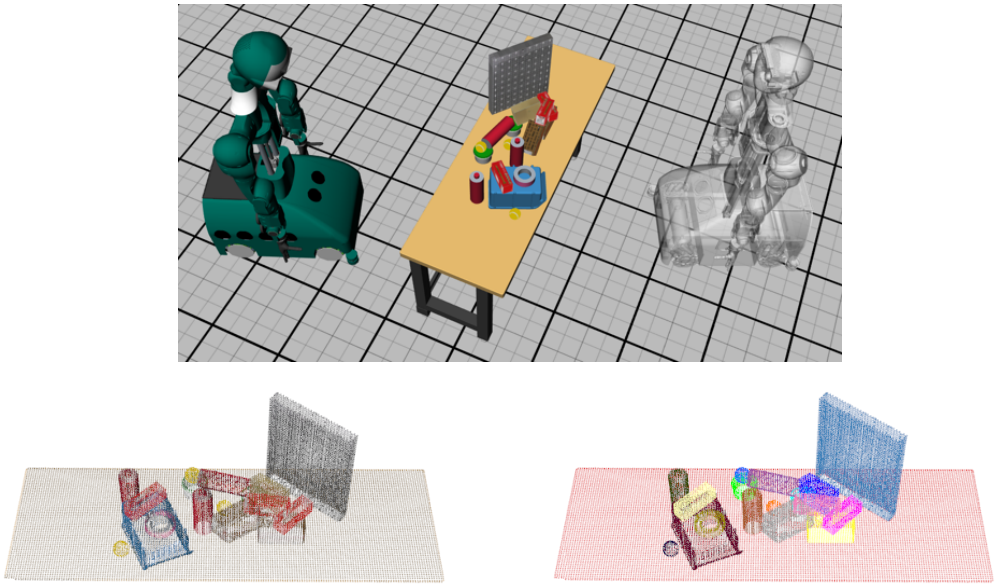


Figure 4.6.: ARMAR-6 simulated experiment. *Top:* A cluttered table-top scenario. *Bottom Left:* The registered point cloud of the cluttered table-top. *Bottom Right:* The ground truth segmentation of the scene.

sensor setup and the robotic platform. Among the several visual sensors, the *Primesense Carmine 1.09* is used to capture RGB-D data. To reduce sensor noise, the depth measurement is limited to $3\,\mathrm{m}$. To create the scene model, as described in Section 3.1, the segmentation was manually refined to avoid bias of the RANSAC based geometric primitive fitting and to make experiments reproducible. The leaf size of the voxel grid for occlusion checking was set to $1\,\mathrm{cm}$.

4.5.1. View Sampling in Simulation

The goal of this experiment is to quantitatively asses the top-down view sampling strategy described in Section 4.2. Therefore, a simulated environment is used. Figure 4.6 visualizes the scenario setup. For the saliency computation in Equation 4.5, $\lambda_s = 0.75$ is used, as it shows a good balance between exploration and validation of support relations. To show the effectiveness of the saliency value, the utility function in Equation 4.13 is substituted by the saliency value described in Equation 4.5. The position of the view sphere was fixed and the radius set to $1.5\,\mathrm{m}$. Figure 4.7 shows the view sphere after the initial view. To quantify the approach, a Support Graph $\mathcal{G}_s^{GT} = (V^{GT}, E^{GT})$ is created manually. This allows to compare the extracted Support Graph $\mathcal{G}_s^i = (V^i, E^i)$ of

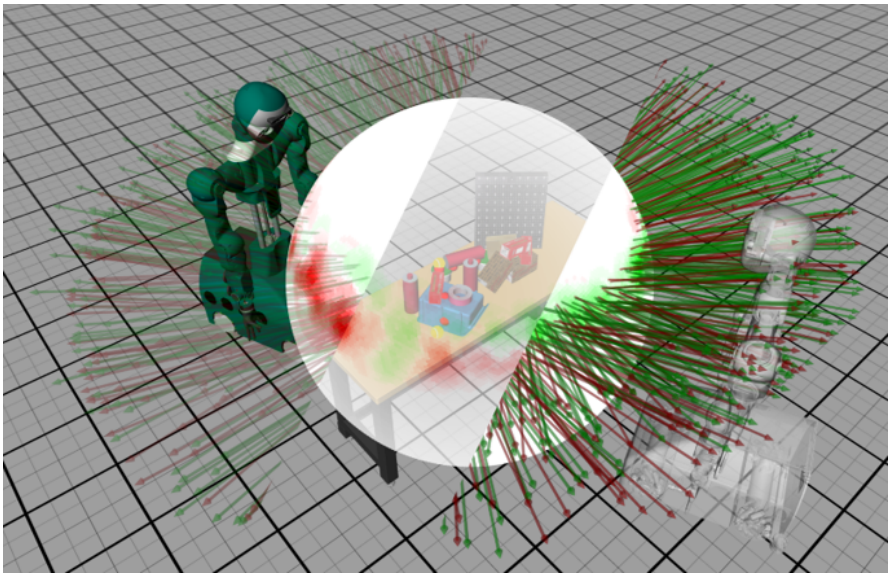


Figure 4.7.: The view sphere including the projected Points of Interest (PoIs) of the simulated experiment. The rays project the PoIs to the sphere and a saliency value models the interest of the view pose. Parts of the sphere have been made transparent for visualization purposes. The selected Next-Best-View is visualized by the robot pose shown in gray.

the i-th view using the F_1-score, which measures the accuracy of the extracted support graph and is defined as

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad , \tag{4.14}$$

where an $F_1 = 1$ means that both Support Graphs (SGs) match. In Equation 4.14 precision and recall are given as

$$\text{precision} = \frac{\left|V^{GT} \cap V^{i}\right| + \left|E^{GT} \cap E^{i}\right|}{|V^{i}| + |E^{i}|} \tag{4.15}$$

and

$$\text{recall} = \frac{\left|V^{GT} \cap V^{i}\right| + \left|E^{GT} \cap E^{i}\right|}{|V^{GT}| + |E^{GT}|} \quad . \tag{4.16}$$

True positives are vertices and edges that exist in $\mathcal{G}_s^{GT}$ as well as in $\mathcal{G}_s^i$. False positives equal the number of vertices and edges in $\mathcal{G}_s^i$, but not in $\mathcal{G}_s^{GT}$. A possible reason for false positives is due to an erroneous RANSAC model fitting. False negatives are the vertices and the edges missing in $\mathcal{G}_s^i$. Different methods include the combination of each view based on the spatial information (*PC*), the support graph (*SG*) or both (*PC* + *SG*). The active vision system was com-

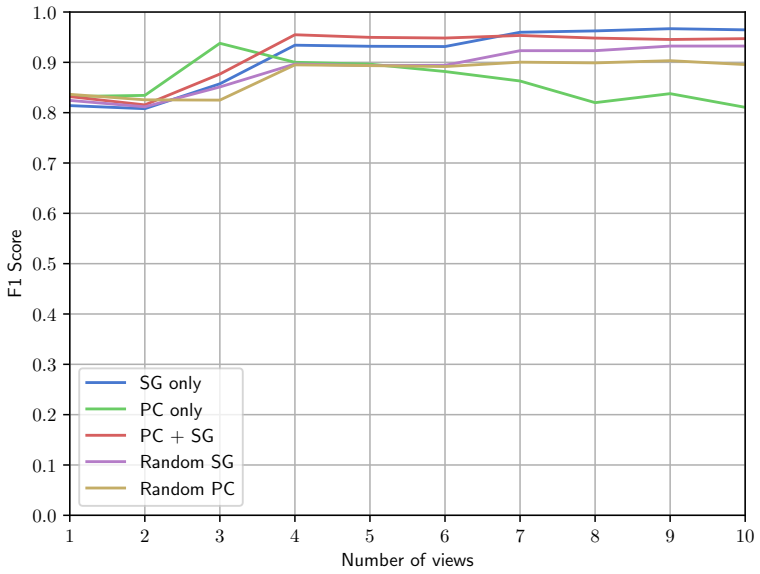


Figure 4.8.: The F_1 score for the first 10 next best views of the simulated experiment. The extracted support graph of the scene is compared to a ground truth support graph. Different methods include the combination of each view based on the spatial information (*PC*), the support graph (*SG*) or both (*PC + SG*) as described in Section 3.3.1. The active vision system was compared to random placement by fusing the information on the support graph (*Random SG*) and spatial information (*Random PC*).

pared to random placement of the robot while fusing the information on the support graph (*Random SG*) and spatial information (*Random PC*). Figure 4.8 shows a plot of the F_1- score for different matching approaches, as described in Section 3.3.1. Notably, all approaches yield an increase with respect to the F_1-score after the second view. However, extracting the support graph from a single registered point cloud only, results in a decline of the F_1-score after the fourth view. One reason for this is the fitting of the geometric primitives, which works with a fixed error threshold of the fitted geometric model. Furthermore, the F_1-score increases with the random placement of the robot as well. However, the random placement of the robot does not consider the distance to reach the next view. Therefore, the total distance traveled by the robot during exploration might be significantly larger than with the proposed methods. This is not taken into account by the F_1- score.

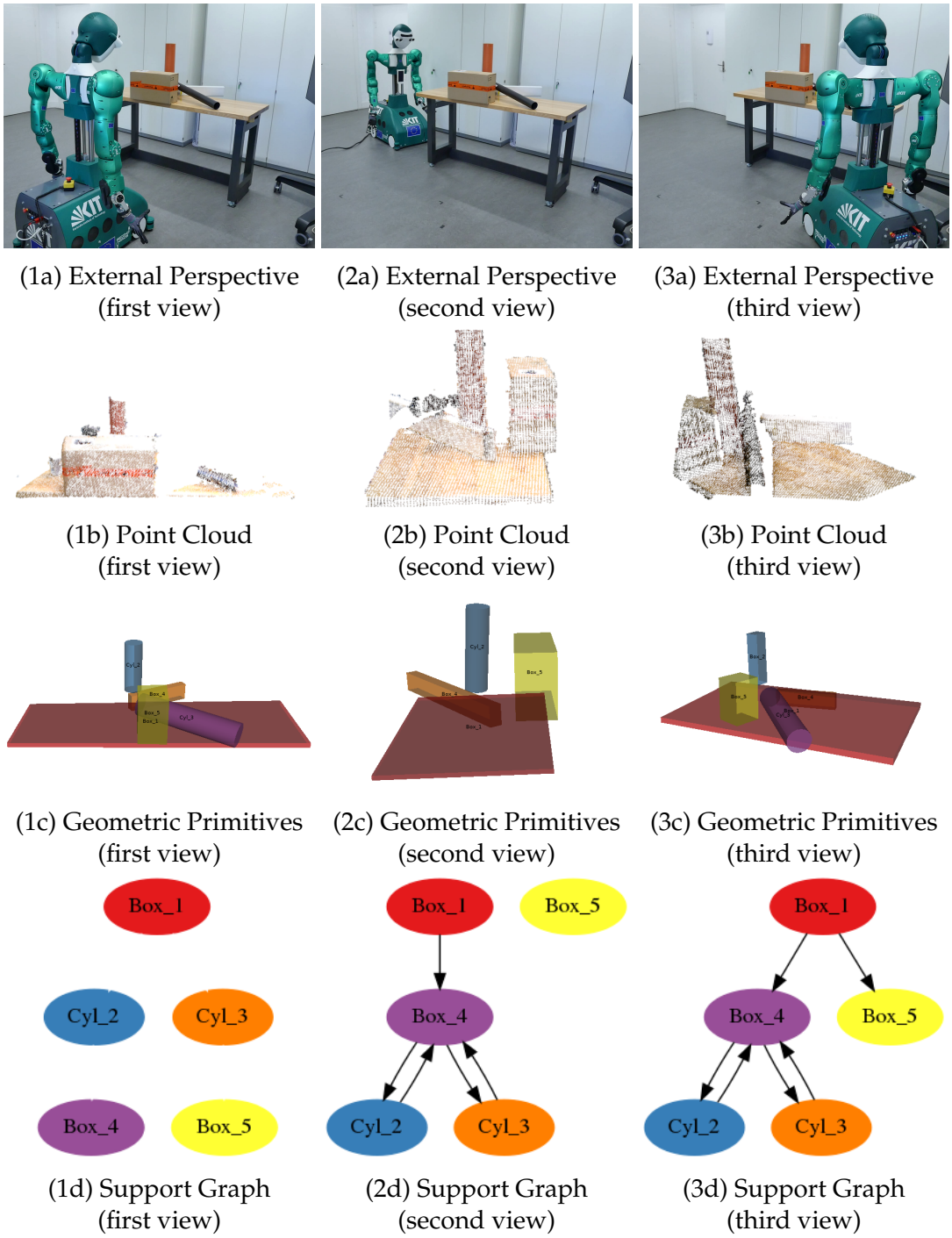
4.5.2. Real World Experiment

To qualitatively assess the top-down view sampling strategy, the following experiment is performed on the humanoid robot ARMAR-6. The goal is to show that (a) the system runs on real robot system, and (b) it yields an improvement with respect to the extracted SG. The real world evaluation is similar to the previous evaluation in simulation. This time, however, the noise of the sensor and the registration error injects noise into the system. Therefore, the support graph is fused with the approach *(SG)* as it performs well in simulation while reducing computation time. Figure 4.9 depicts three selected next best views at different timestamps of the experiment. The scene is relatively simple, but due to occlusion requires multiple views to extract a complete support graph of the scene. As one can observe from the first view (first column of Figure 4.9), no support relations are extracted due to an occluding object. The active vision system therefore generates PoIs between each object pair and the robot attends the NBV. In the second column, the support graph is still incomplete, but the most important support relations are discovered. Finally, in the third column the NBV discovers a missing support relation.
Overall, this experiment showed that the system is able to perform on a real robotic system using the top-down view sampling and its saliency measure only.

4.5.3. Evaluation of the Utility Function

The following experiment quantitatively evaluates the utility function, which determines the quality of a view. Therefore, a simulated environment is used, which shows the humanoid robot ARMAR-6 at an exhibition booth. The simulation uses a CAD model of a real exhibition booth at CeBIT 2018, where the robot was presented to the public for the first time. The layout of the booth included five meeting points, where visitors could get in touch and information was displayed. For this experiment, three of the contact points were identified as Points of Interest (PoIs). The goal of the experiment is that the robot visually investigates these regions. Figure 4.10 shows the simulated scene and the possible views to evaluate.

To compare the utility function to other methods, possible views are sampled first, as described in Section 4.2. For evaluation, the views are kept fixed. Further, to avoid bias of the RRT planner the Euclidean distance was used to cal-



(1a) External Perspective (first view) (2a) External Perspective (second view) (3a) External Perspective (third view)

(1b) Point Cloud (first view) (2b) Point Cloud (second view) (3b) Point Cloud (third view)

(1c) Geometric Primitives (first view) (2c) Geometric Primitives (second view) (3c) Geometric Primitives (third view)

(1d) Support Graph (first view) (2d) Support Graph (second view) (3d) Support Graph (third view)

Figure 4.9.: A real world experiment with ARMAR-6. The figure shows three next-best-views at different timestamps of the experiment. After the second view more support relations are discovered by the robot. *First row*: Scene and position of the robot. *Second row*: Current point clouds. *Third row*: Extracted geometric primitives. *Fourth row*: Extracted support graph. Figure taken from Grotz et al. (2019) (© 2019 IEEE).

culate the distance between the views. The experiment was repeated with different methods to determine the quality of a view. Therefore, the following methods are used:

(1) Occupancy: Only unknown voxels are considered to determine the NBV. Views are evaluated with Equation 4.6.
(2) Utility: A combination of unknown voxels and distance to the views is considered. Views are evaluated with Equation 4.13.
(3) Random: The robot selects a random view. Views are evaluated with $v \mapsto x$ with $x \sim \mathcal{U}(0, 1)$.

The random strategy is kept as a baseline. To compare the different methods with respect to each other, the following criteria are used.

(1) Total number of unknown voxels, and
(2) Traveled distance

The number of unknown voxels was determined by defining a 3D bounding box for the environment. Figure 4.11 shows the number of unknown voxels and the accumulated path costs. Independent of the evaluation, all methods yield a lower number of unknown voxels after each view. Since the occupancy evaluation only considers the information gain, it performs best with respect to the other methods. Figure 4.12 shows the accumulated estimated path costs. The utility function scores best, since it is the only method that considers the estimated path costs. In contrast, the occupancy evaluation shows the highest costs. This is due to the fact that, it is a greedy method that only considers the

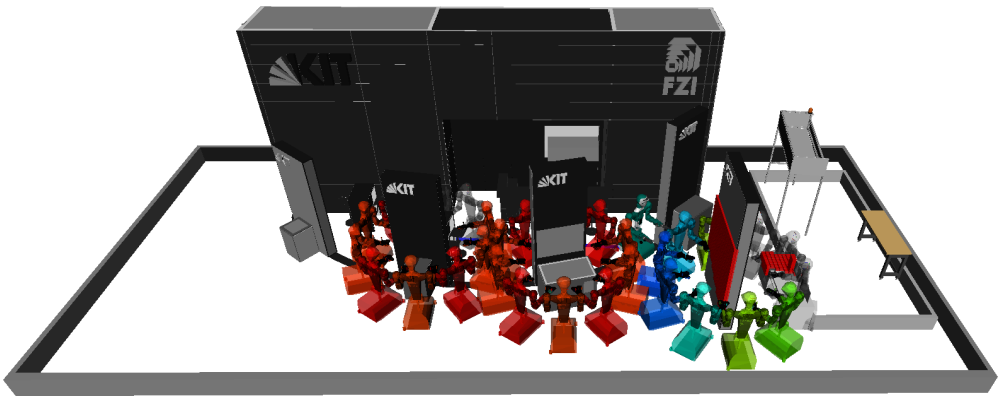


Figure 4.10.: The initial view. Possible next-best-views are indicated with robot silhouettes. The color scheme uses a heatmap to indicate the score. A low score is blue, whereas a high score is shown with a red color. Unreachable views are shown in gray.

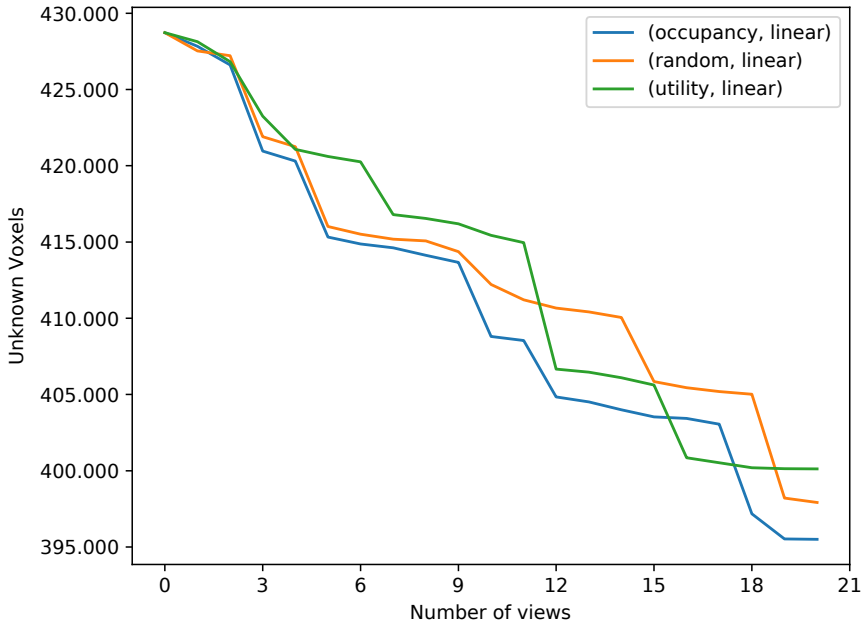


Figure 4.11.: The scene is shown in Figure 4.10. The utility function considers the estimated path costs and thus is able to reduce to accumulated distances. In contrast to that, the occupancy evaluation is a greedy method that only considers the expected information gain.

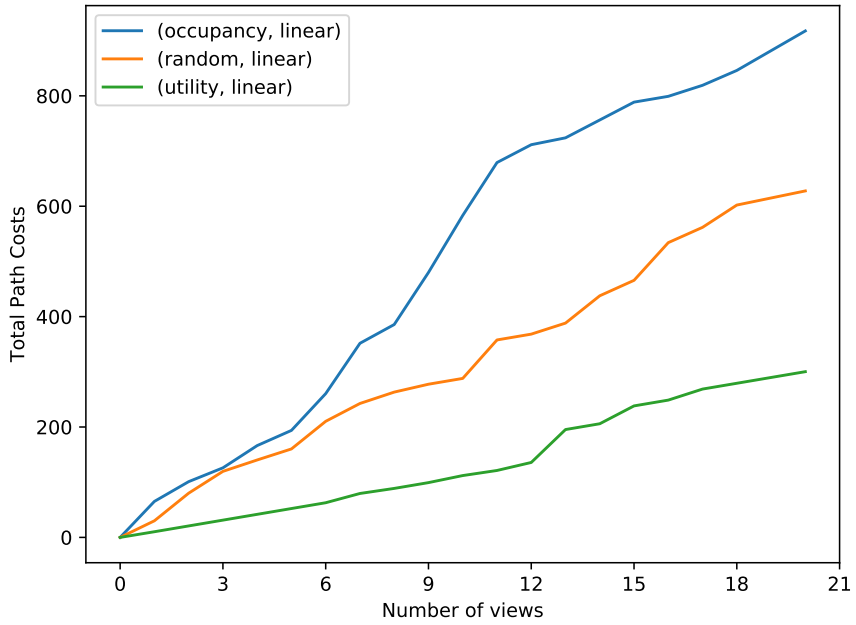


Figure 4.12.: The accumulated estimated path costs. The utility function considers the estimated path costs and thus is able to reduce to accumulated distances. In contrast, the occupancy evaluation is a greedy method that only considers the expected information gain.

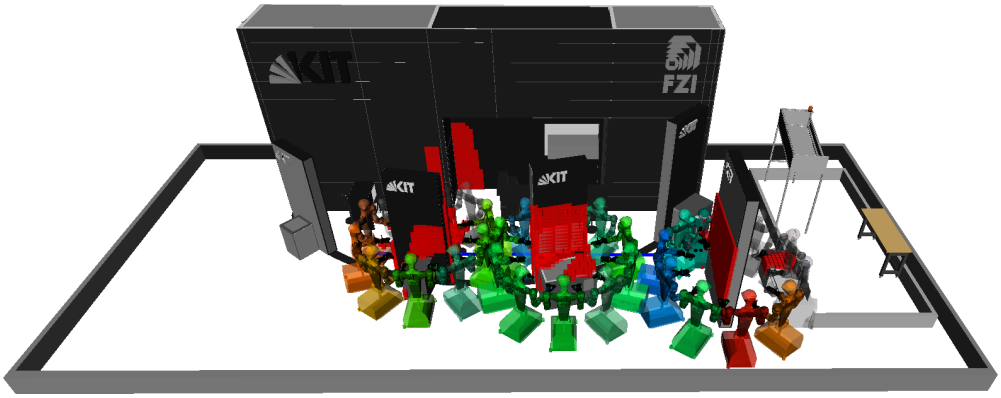


Figure 4.13.: The evaluation after 6 views. The color scheme uses a heatmap to indicate the score. A low score is blue, whereas a high score is shown with a red color. Unreachable views are shown in grey.

expected information gain. Figure 4.13 shows the simulated experiment with the occupancy evaluation after 6 views.

The experiment showed that the method increases the scene coverage with each view. While the view evaluation using the volumetric information gain scores best with respect to the scene coverage it also results in the highest path costs. Finally, the utility function shows a good balance between the scene coverage and path costs.

4.5.4. Real World Scene Coverage

This experiment evaluates the performance of the system on a real robotic system. ARMAR-6 is located in front of a cluttered table-top scenario. The robot does not have any knowledge about the scene or the objects on the table. The goal is to extract a scene model including physical relationships among the objects. The height during the view sampling was kept fixed during the experiment, i. e., the torso joint of the robot was disabled. Figure 4.14 depicts the scene.

Similar to the simulated experiment, this experiment was repeated with the occupancy and the utility evaluation. Figure 4.17 and Figure 4.18 show the first three views for the occupancy evaluation and the utility evaluation respectively. Using the occupancy evaluation, the robot visits the view with the highest number of unknown voxels. Again, the number of unknown voxels and the path costs are plotted for each view in Figure 4.15 and Figure 4.12.

(a) External view. (b) First view of the robot.

Figure 4.14.: Experiment Setup. The goal of this is experiment is to show that ARMAR-6 can autonomously explore the table-top scenario.

The experiment confirms the observations from the previous experiment on a real humanoid robot system. One can see a significant reduction of the unknown voxel count using the occupancy method. However, this method requires the most path costs.

4.6. Summary

This chapter presented an integrated active vision system to support the creation of a semantic scene model including the extraction of physically plausible support relations as described in Chapter 3 based on multiple views of the scene. This is a key contribution to the autonomous perception of unknown environments. The scene is iteratively explored by planning the Next-Best-View (NBV). Therefore, views are sampled based on semantic information of the scene model. Sample views are then subjected to further evaluation. The formulation of a utility function includes aspects of the semantic information as well. A comprehensive, quantitative evaluation of the proposed NBV system shows that multiple views are necessary to mitigate the effect of occluded objects. Both the evaluation in simulation as well as the real world experiment show a completion of the support graph after a few attended NBVs. The real world experiment demonstrates the necessity of the active vision method for cluttered table-top scenarios.

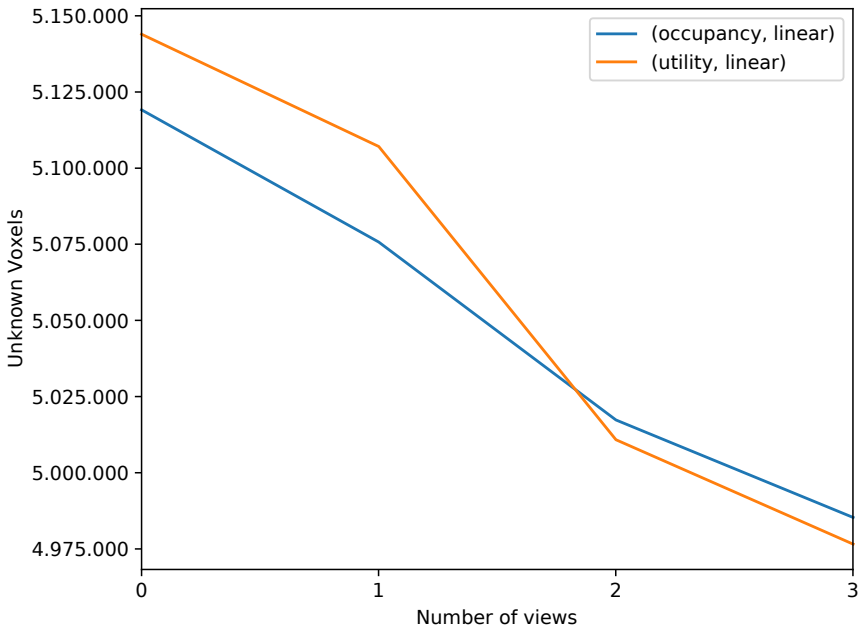


Figure 4.15.: The utility function considers the estimated path costs and thus is able to reduce to accumulated distances. In contrast, the occupancy evaluation is a greedy method that only considers the expected information gain.

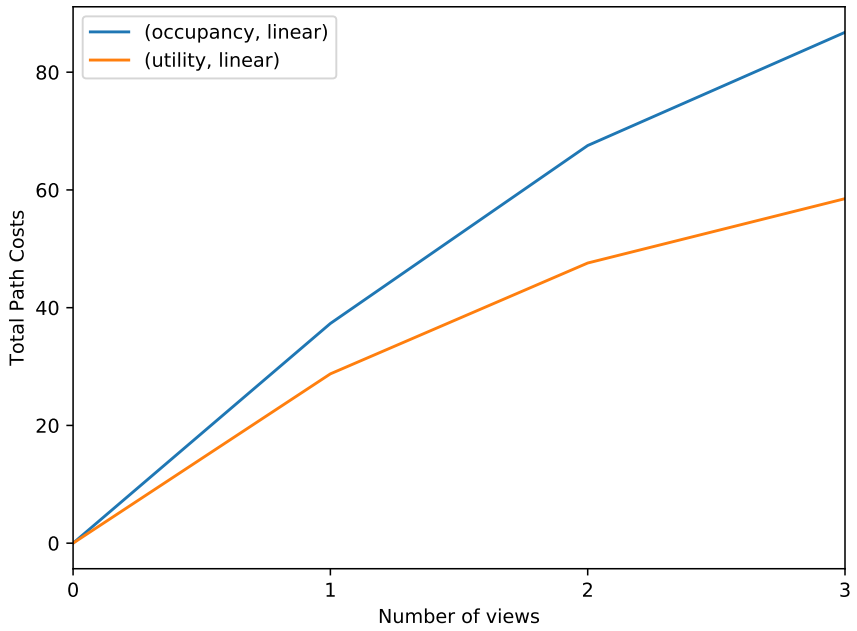
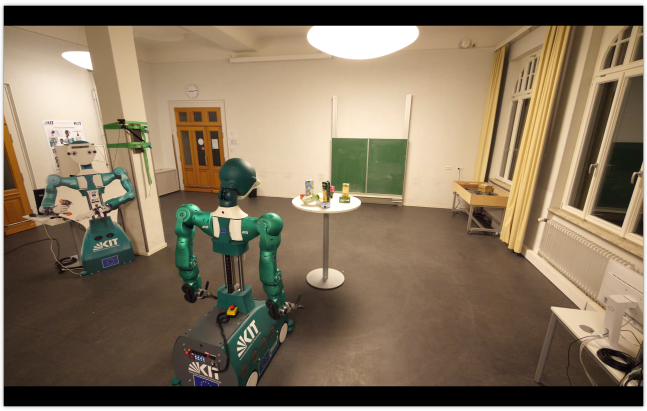


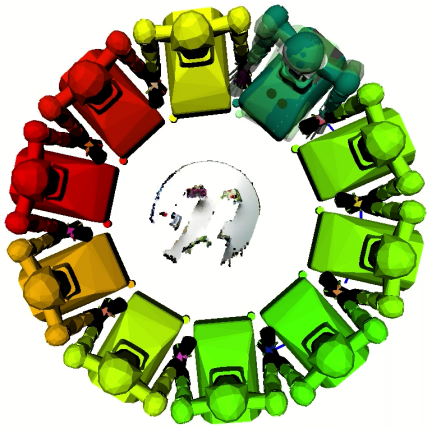
Figure 4.16.: The utility function considers the estimated path costs and thus is able to reduce to accumulated distances. In contrast, the occupancy evaluation is a greedy method that only considers the expected information gain.



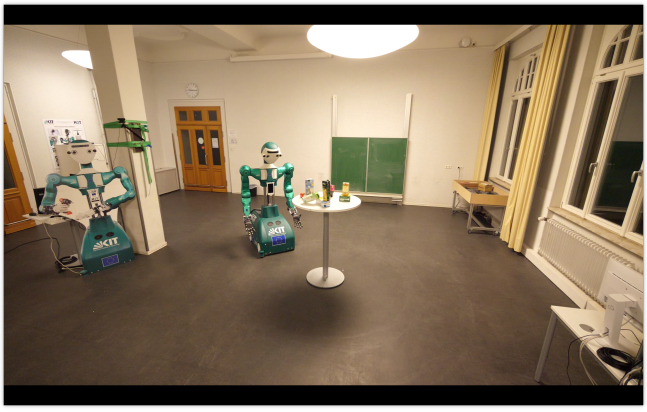
(1a) External View
(first view)

(1b) Evaluated Views
(first view)

(2a) External View
(second view)



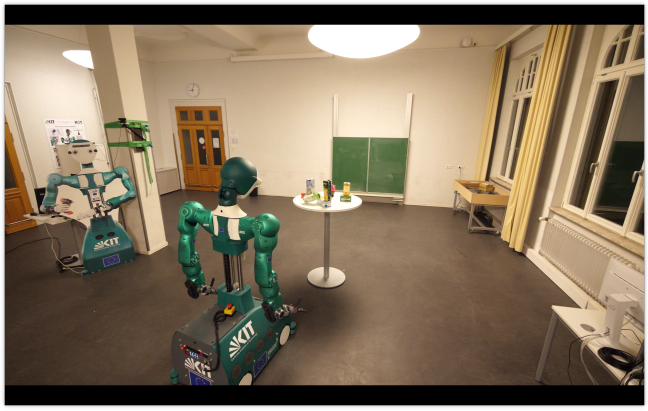
(2b) Evaluated Views
(second view)



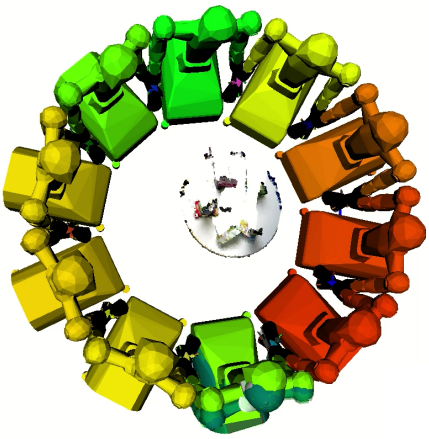
(3a) External View
(third view)

(3b) Evaluated Views
(third view)

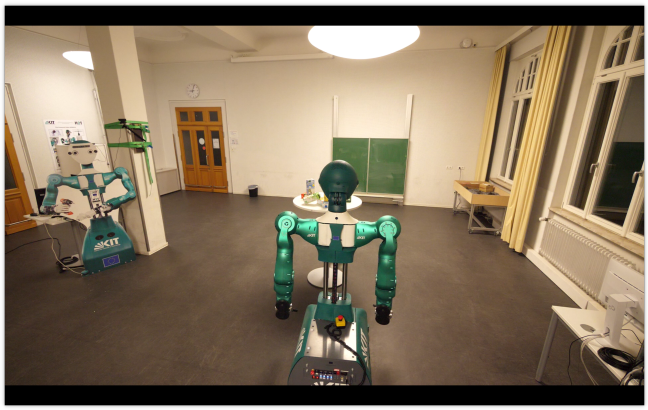
Figure 4.17.: A real world experiment with ARMAR-6. The figure shows the first three views. The NBV is determined by counting the number of unknown voxels.



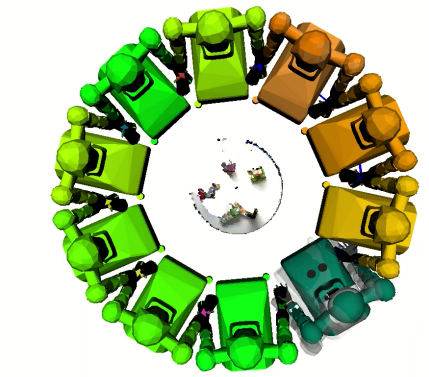
(1c) External View
(first view)



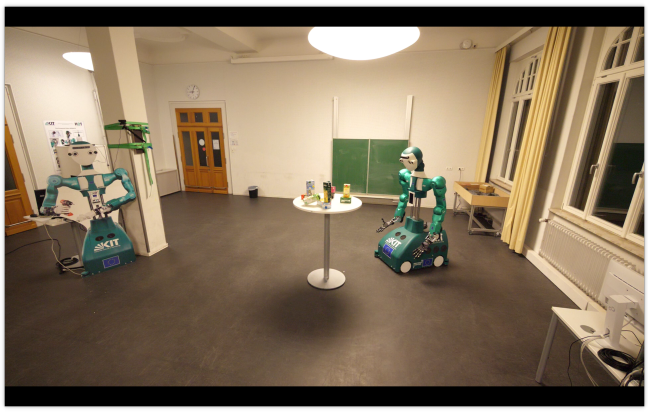
(1d) Evaluated Views
(first view)



(2c) External View
(second view)



(2d) Evaluated Views
(second view)



(3c) External View
(third view)

(3d) Evaluated Views
(third view)

Figure 4.18.: A real world experiment with ARMAR-6. The figure shows the first three views. The NBV is determined by using the utility function. The utility function considers both the information gain as well as the traveled distance by the robot.

5. View Selection and Gaze Stabilization

Changing the gaze or executing manipulation actions impairs the visual perception. Therefore, it is necessary to stabilize a humanoid robot's gaze. Figure 5.1 shows an example of ARMAR-6's camera images taken from a real world experiment without any camera stabilization. Camera images are blurry, the illumination changes, and the region of interest, e. g., an object being localized, is out of view due to motion of the robot. As consequence, vision based algorithms perform poorly due to noisy input data. Hence, the visual perception needs to be stabilized. Indeed, it is difficult to compensate for all disturbances. Methods for stabilization, however, are able to deal with some issues, such as keeping the region of interest centered, and thereby support vision-based tasks.

Figure 5.1.: Camera images of ARMAR-6's PrimeSense RGB-D sensor while the head is moving at different times. During the gaze shift towards the grasped spray bottle camera images are blurry.

This chapter investigates the link between gaze stabilization methods and active vision methods. Both, active vision and gaze stabilization, support independently a humanoid robot's visual perception and are crucial for a robust operation in real world scenarios. In fact, both components actively control a robot's gaze, i. e., head and eyes, and therefore these two components need to be orchestrated in order to avoid a resource conflict. For example, an active vision system itself shifts the gaze and thereby induces noise to the visual perception since it moves the camera. Such self-induced perturbation needs to be

compensated by gaze stabilization. This also includes movements of the robot, such as changing a robot's position or moving a joint. Hence, the system architecture presented in this chapter includes both gaze stabilization and active vision while also considering the interaction and the synchronization of both. Ultimately, the goal is a higher level gaze control architecture combining active vision and gaze stabilization.
This chapter first describes a view selection mechanism in Section 5.1 followed by gaze stabilization methods in Section 5.2. Section 5.3 outlines the system architecture, comprising of gaze stabilization and active vision. The evaluation consists of several real-world experiments described in Section 5.4.

Methods and results presented in this chapter are published at peer-reviewed conferences and mainly based on the work of Grotz et al. (2017a), Habra et al. (2017), and Sippel (2016).

5.1. View Selection

The active vision system, described in the following, focuses on supporting the execution of grasping and manipulation tasks on a humanoid robot. Therefore, the definition of a Next-Best-View is extended to include other task related and bottom-up information as well. However, this definition is in line with the ideas and the methods presented in previous chapters. Since some gaze stabilization methods require a fixation point in the scene, the term *view target* is used referring to a 3D point in the scene with respect to the world coordinate system, whereas the term *view* refers to a 6D camera pose. In conjunction with Chapter 4, given a *view target* in the scene the *view*, i. e., the pose of the camera, can be directly computed to shift the gaze and fixate the given view target. Since the focus of this chapter is on grasping and manipulation, the proposed methods in Welke et al. (2013) are used and extended to determine a view target. In particular, for grasping and manipulation actions, the pose of the object to interact with has to be known. In the context of this work, the object localization is provided by methods described and evaluated in Azad et al. (2009) and Azad et al. (2007).

The idea behind the active vision system is that possible views are represented on a sphere around the center of the robot's head. The representation is similar to the view sphere defined in Section 4.1. However, the representation here is egocentric since coordinates are stored with respect to the robot. Further, the

direction of the representation is inverted. Each point on the sphere represents a gaze direction with an associated saliency value. The saliency value is to measure the *interestingness* of an item, i. e., by which an item stands out from its neighborhood. The intuition behind this representation is that the saliency value corresponds to the importance of the information that can be gained if the robot shifts its gaze to the direction of the point. Such a sphere is known as *sensory ego-sphere* (Peters et al., 2009) and allows to consider multimodal sensor cues (Ruesch et al., 2008). In this implementation, the sensory ego-sphere is discretized to $40,000$ equally distributed points. Saliency values s_j^k from different sources for a point p_j on the sphere are accumulated using a weighted sum. Thus, the aggregated saliency value s_j is defined as

$$s_j = \frac{1}{\sum w_k} \sum_{k=1} w_k s_j^k \ , \tag{5.1}$$

where w_k is the associated weight of the computed saliency values s_j^k. In the following, the index k for the saliency value s_i^k is omitted and the object saliency value is referred to as s_i as it can be derived from the context. To track outdated values, a timestamp t is added to each saliency value s_j^k. Values with a timestamp older than $5\,\mathrm{s}$ are simply discarded if not specified otherwise. A post-processing step prunes unreachable points on the sphere, e. g., where no Inverse Kinematics (IK) solution exists and the robot is thus unable to direct its gaze to the point. Thus, these targets are not considered by the view selection mechanism. Similar to the definition of the Next-Best-View (NBV) as given in Chapter 4, the active vision system selects the point $p_{\hat{j}}$ on the sphere with maximum saliency value, i. e., $\hat{j} = \arg\max_j s_j$. Next, joint positions for the head and eye joints are computed using the IK solver of the Simox library (Vahrenkamp et al., 2013) and the view is shifted accordingly. Figure 5.2 illustrates the computation of a view target and the selection of a view. Multiple saliency values from different sources are combined and post processed. The computation of saliency values is described in the following section. The system architecture also allows setting manual view targets, which are then considered by the view selection mechanism.

5.1.1. Saliency Computation

The computation of the object saliency value is described in the following and follows the work of Welke et al. (2013). A point p_i on the sphere contains multi-

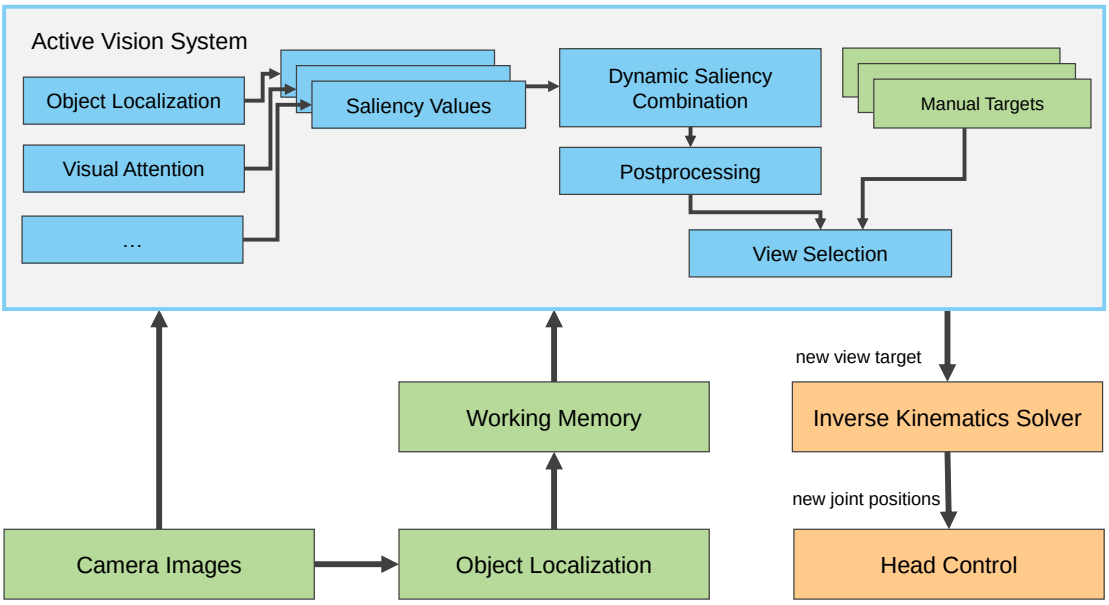


Figure 5.2.: System architecture to compute a *view target*. Multiple saliency values are considered in a dynamic saliency combination step and post processed. Saliency values are computed, for example, based on the uncertainty of object localization results. The view direction with the highest saliency is chosen as NBV. Figure adapted from Grotz et al. (2017a) (© 2017a IEEE).

ple saliency values. For grasping and manipulation, the most important visual component is the object pose. Besides that, a very small random noise is added to get a variance in gaze directions.

Formally, a saliency value is computed as follows. Let o_i be a target object that the robot is going to grasp or manipulate and thus needs to be localized in order to know the object pose $p_i = (R_i, \boldsymbol{t}_i) \in SE(3)$. Due to sensor noise, the object localization result is inaccurate. New object localization results are fused with a Kalman filter. The respective uncertainty about the object pose is modeled as a multivariate normal distribution $\mathcal{N}_i(\mu_i, \Sigma_i)$ with mean $\mu_i = \mathbf{0}$ and covariance matrix Σ_i. Overall, the localization result of object o_i is given by the quadruple

$$(R_i, \boldsymbol{t}, \mu_i, \Sigma_i) \ , \qquad \text{(Object Localization Result)}$$

where the first two elements are the 6D object pose and the last two elements model the object's pose uncertainty. An object pose with low uncertainty is desired and hence localization results for the same object o_i from multiple views are fused and the uncertainty is updated. Thus, when changing the view, the object localization uncertainty needs to be considered. The first step to compute a saliency value is to map the covariance matrix Σ_i to a scalar value. Hence, the

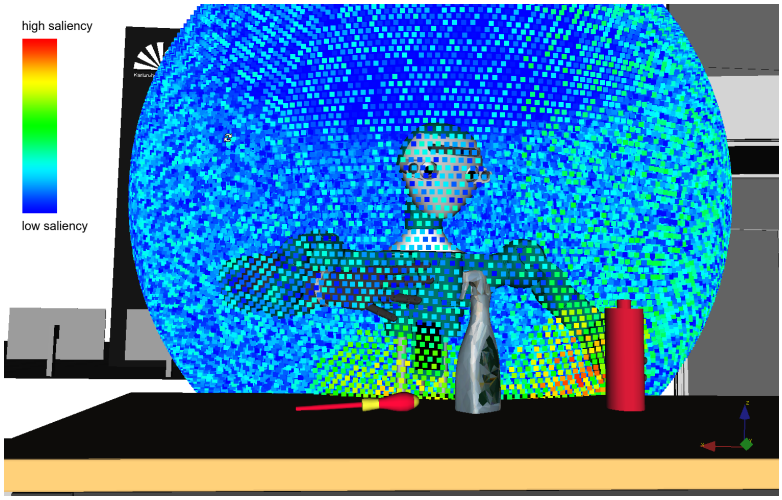


Figure 5.3.: Parts of the saliency ego-sphere for ARMAR-6. Different views are represented on a sphere around the robot's head. The sensory ego-sphere consists of 40,000 nodes each representing a possible view with a saliency value. Saliency values are visualized with a heatmap. The color blue indicates zero saliency and red a maximum saliency.

uncertainty is translated to the scalar value

$$\sigma_i = det(\Sigma_i)^{\frac{1}{6}} \quad . \tag{5.2}$$

The idea behind this is that the saliency value s_i of object o_i indicates the importance of the information given its pose uncertainty. Thus, it is set to equal to the differential entropy

$$u_i = \frac{1}{2} \ln \left(\left(2\pi e {\sigma_i}^2 \right)^3 \right) \quad . \tag{5.3}$$

For grasping and manipulation tasks, the importance of the object localization result can vary and therefore the saliency value should be task-specific. Hence, a task acuity is used in the calculation of object saliencies. The value α_i models the acuity of the object localization as required by the task. Similar to Equation 5.3, the differential entropy is utilized and thus the specific task acuity is modeled with

$$b_i = \frac{1}{2} \ln \left(\left(2\pi e {\alpha_i}^2 \right)^3 \right) \quad . \tag{5.4}$$

With Equation 5.3 and Equation 5.4, the final saliency for an object o_i is then set to $s_i = u_i - b_i$. To avoid the accumulation of negative saliency values, s_i is clamped to be ≥ 0. Figure 5.3 depicts the saliency ego-sphere for ARMAR-6.

5.2. Gaze Stabilization Methods

Keeping the area of interest centered in the field of view is a vital prerequisite for retrieving any useful information with cameras. This is even more important for foveated camera systems since the field of view is more narrow. Additionally, most vision algorithms rely on a stable camera image to reach optimal performance.
Gaze stabilization computes compensatory head and eye movements to compensate for disturbance in visual perception. In humans, the eye is mainly stabilized by the Vestibulo-Ocular Reflex (VOR) and Optokinetic Reflex (OKR) (Miles, 1998). For the ARMAR humanoid robots, several gaze stabilization methods have been developed. This includes most recently Sippel (2016) and the gaze stabilization controller described in Habra and Ronsse (2016), which was extended for the ARMAR humanoid robots in Habra et al. (2017). The controller is of particular interest as it allows fixation of the given view target. For details the reader is referred to Habra et al. (2017) and Habra (2017) since the focus of this chapter is not on gaze stabilization, but more on the link between gaze stabilization and active vision.
The basic idea of the aforementioned gaze stabilization controller rests upon intercepting the robot's motor command, apply them to an internal robot model. Thereby, the internal model predicts the new gaze direction of the robot and computes the deviation to the desired view target x_{des}. In the next step, compensatory head and eye motor commands are computed to correct the gaze and to fixate the view target. Figure 5.4 illustrates the functionality of the gaze stabilization controller, which is described in the following.

Given the current view target x_{FP}, the gaze stabilization controller formulates the task of determining the robot's joint parameters in order to direct the gaze at the view target x_{des} using a virtual end-effector. Thus, the gaze stabilization controller can be seen as classical control of a robot and hence solved with inverse kinematics method. The idea of using a virtual end-effector is described in Omrčen and Ude (2010). Due to more degrees of freedom available it is underdetermined and the solution is not unique. Indeed, the system of linear equations can be subject to additional constraints. For example, in Habra and Ronsse (2016) the optical flow is minimized. Compensatory eye and head velocities $\dot{q}_{head}$ are computed as sum of (1) feedback of the position error $\dot{x}_{FB}$, and (2) feed-forward velocity $\dot{x}_{pred}$. The gaze stabilization controller as presented in Habra and Ronsse (2016) was adapted and revised for the use for the ARMAR

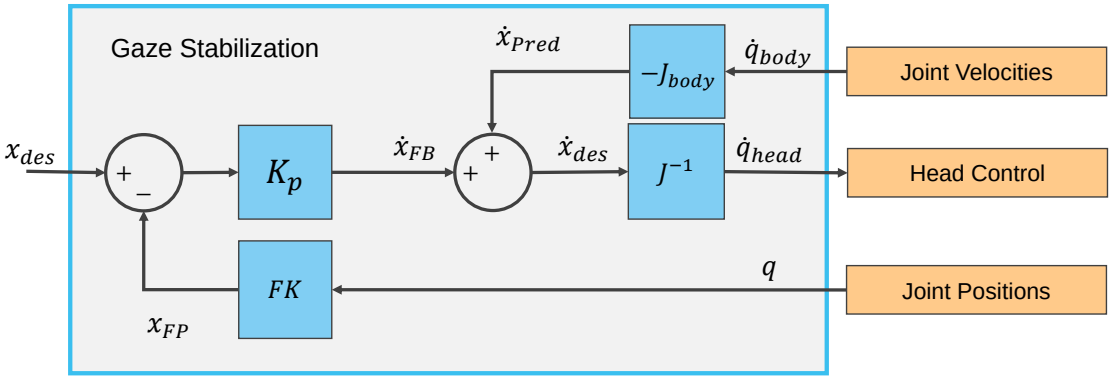


Figure 5.4.: Inverse kinematics methods for gaze stabilization. Based on the current joint velocities compensatory head and eye movements are computed to stabilize a given fixation point in the scene. Figure adapted from Habra and Ronsse (2016) (© 2016 IEEE).

humanoid robots as presented in Habra et al. (2017). In particular, the changes include that the term $\dot{x}_{pred}$ is computed as a compensation for the self-induced motion. Instead of estimating the self-induced motion based on intercepted velocity commands sent to the whole-body joints the velocity is directly measured from the encoder to compute the feed-forward term. Further, the feed-forward compensation was extended to include the result of the self-localization of the robot. Finally, the kinematic redundancy resolution was adjusted to take into account the difference between the eye and neck joints for active vision. Indeed, many vision based algorithms rely on an accurate stereo calibration. Thus, neck joints are preferred and eye joints are only moved when necessary to keep the stereo calibration as accurate as possible. Concretely, the weight factors for velocity minimization of the eye joints were set eight times higher than for the neck joints.

5.3. Gaze Control Architecture

The higher level gaze control architecture comprising a gaze stabilization system (Section 5.2) and a view selection system (Section 5.1), as shown in Figure 5.5, is described in the following. From a functionality point of view, both systems are intertwined since both systems update the gaze target by controlling the head motors and the gaze stabilization indirectly feedbacks the view selection by providing more stable input images. From a software engineering point of view, the systems are developed as standalone components. A well-defined interface allows for interaction between these components.

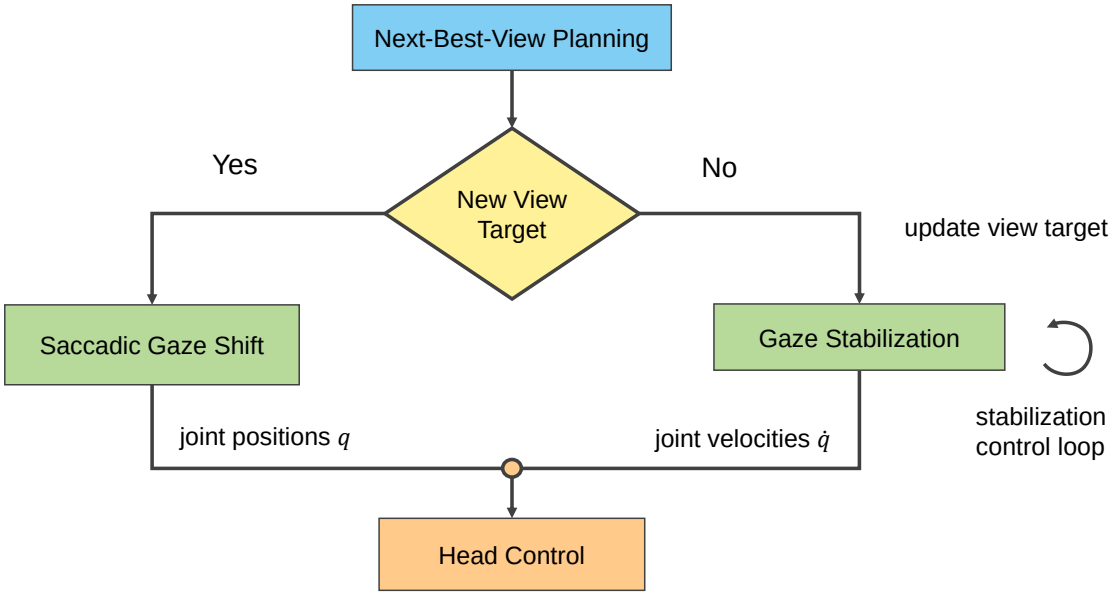


Figure 5.5.: The gaze control architecture with the active vision and the gaze stabilization components. The active vision method computes a Next-Best-View (NBV), which is constantly fixated by the gaze stabilization system. Both components support the visual perception by selecting the NBV and stabilizing the current view. Figure adapted from Grotz et al. (2017a) (© 2017a IEEE).

First, the view target is determined either based on the current task or based on salient regions in the scene. A new gaze directions is selected in regular intervals or the view selection is invoked to shift the gaze to a new target. Once a new view target x is computed, the gaze stabilization system is triggered to allow the execution of saccadic eye movements and to support visual perception by stabilizing the new view target. Saliency values for object localization are computed and passed to the view selection component. Sensor input values for the gaze stabilization consists of joint velocities, joint position, the computed sparse optical flow and acceleration values of an Inertial Measurement Unit (IMU). Output values of the gaze stabilization system are joint velocities for the head and eye. These velocities are passed to the hardware abstraction layer of the robot. Two human inspired reflexes for gaze stabilization can be invoked: the Vestibulo-Ocular Reflex (VOR) and the Optokinetic Reflex (OKR). Self-induced motions are predicted and removed from the sensor values. The system architecture is inspired by the reafference principle. For more details, the reader is referred to Habra et al. (2017). In the following only self-induced motions by the robot are considered since they are the most common perturbations. Figure 5.6 illustrates the gaze control system architecture.

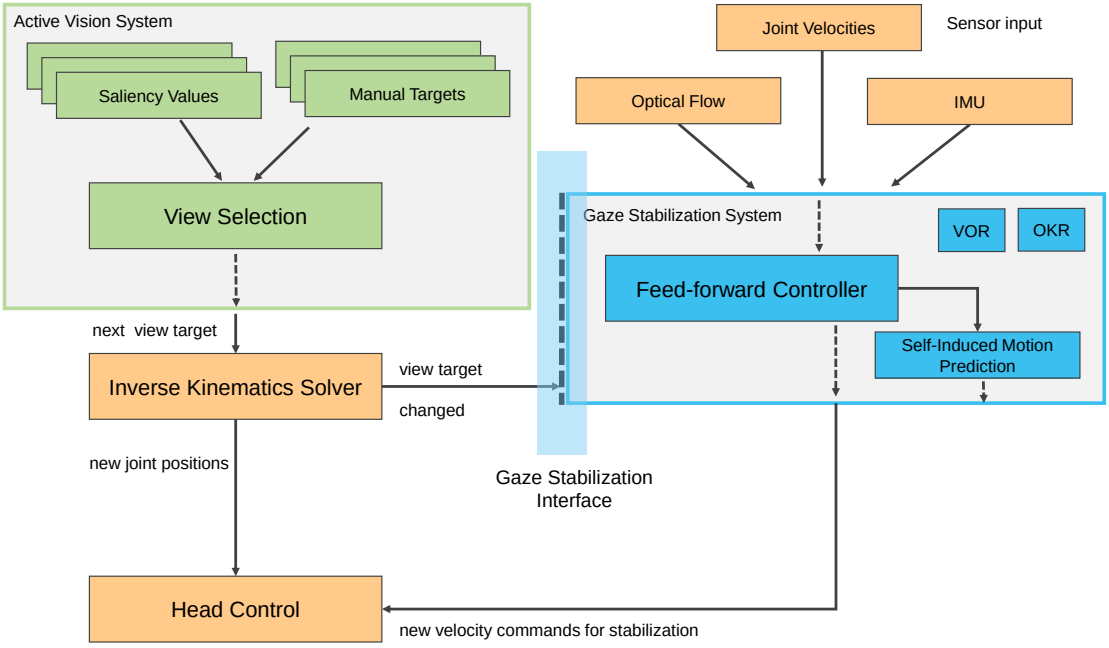


Figure 5.6.: The gaze control system combining gaze stabilization and active vision. The view target is determined by the active vision system while the gaze stabilization system computes motor commands to fixate and stabilize the current view target. The active vision system passes the view target to the gaze stabilization controller. The gaze stabilization controller supports the active vision system by providing stable camera images. Figure adapted from Grotz et al. (2017a) (© 2017a IEEE).

5.4. Evaluation

The presented gaze control system architecture, consisting of an integrated active vision system with gaze stabilization and a view selection strategy, has been validated in real world experiments to assess the interaction between gaze stabilization and active vision.

The following experiments were performed on ARMAR-III. First, a task-oriented experiment quantitatively evaluates the merits of using a gaze stabilization controller for visual perception tasks. In this experiments, objects are localized while the robot is subjected to a sinusoidal perturbation. Second, the link between active vision and gaze stabilization is qualitatively assessed with a grasping while moving action, a complex real world experiment. All experiments utilize foveal cameras, which are beneficial for the first and required by the second experiment. On the one hand, foveal cameras allow to increase the object localization accuracy and also enable to localize objects from a distance.

On the other hand, foveal cameras are more prone to movements of the robot since the field of view is smaller. The subsequent experiment shows that gaze stabilization enables to utilize foveal cameras while the robot is in motion.

5.4.1. Object Localization While Moving

Stabilization results of the gaze stabilization methods are reported in Habra (2017) with peripheral vision. Here, the gaze stabilization controller is evaluated in a more task-oriented setting using foveal vision. Object localization, i. e., 6-D pose estimation of unknown and known objects is a typical task-oriented setting relying on visual perception. Knowing an object's pose allows for interaction, such as grasping or manipulating the object. Accuracy of the object localization methods was previously evaluated in Azad et al. (2007, 2009). Further, the object localization methods have been used extensively in large scale experiments (Wächter et al., 2018). However, previous experiments ignored self-induced motions and external camera perturbations. Up to now, in order to localize objects the robot's platform was stopped and motion of the camera was minimized when localizing objects. Results during motion of the robot have been discarded since objects were either not found or the localization result was erroneous. Therefore, the effectiveness of gaze stabilization to support object localization with foveal cameras during self-induced motions is evaluated in the following. The setup of the experiment was carefully designed to reproduce similar real world applications. Several objects are placed in front of ARMAR-III, at a distance of $1.50\,\mathrm{m}$, as shown in Figure 5.7.

The gaze of the robot was initially set to a fixed view target x in the scene. This initial view target ensures that the localized object is visible in the center of the foveal cameras. Any other components controlling the head, such as the previously presented view selection, were deactivated during the experiment to avoid a conflict of control and for the sake of repeatability.
Since the utilized methods for object localization can be split into single colored objects and textured objects, the experiment was repeated with different kind of objects. A single colored object, i. e., the green cup shown in Figure 5.7, is localized while the robot periodically moves the torso joint. The torso velocity $\dot{q}_{torso}$ for the motion is generated using the sinusoidal function

$$\dot{q}_{torso} = A \cdot \cos(2\pi f t) \ , \tag{5.5}$$

Figure 5.7.: Setup of the object localization experiment. Several objects are placed in front of the humanoid robot ARMAR-III. The torso is subjected to a sinusoidal motion while the objects are constantly localized. Figure taken from Grotz et al. (2017a) (© 2017a IEEE).

with a frequency of $f = 0.25\,\mathrm{Hz}$ and with an amplitude of $A = 20°$. The frequency reflects the typical movements of the robot. In addition, the sinusoidal motion prevents a backlash at the joint limits which would trigger unpredictable camera perturbations. Overall, the self-induced motion is designed in such a way that camera perturbation and thus image noise are as reproducible as possible. Other body joints and platforms are kept fixed at the initial joint angle during the whole experiment. Thus, head and eyes follow the self-induced torso motion according to the sinusoidal motion in lateral direction without gaze stabilization. Figure 5.8 shows recorded images of the left foveal camera at selected key frames of the experiments. In this experiment, objects are localized every $50\,\mathrm{ms}$ for single colored objects and $70\,\mathrm{ms}$ for textured objects respectively. The difference is due to the fact that the approach for localizing textured objects computes keypoints and descriptors. There this approach is more computationally intensive than the approach for single colored objects, which uses a *HSV* color segmentation to detect possible objects in the image. Each individual experiment lasted $10\,\mathrm{s}$. The cameras settings, i. e., auto exposure and auto white balance were configured once and then kept fixed during the experiment to avoid any bias. The same experiment is repeated using a textured object, the multivitamin juice from the KIT object models database (Kasper et al., 2012).

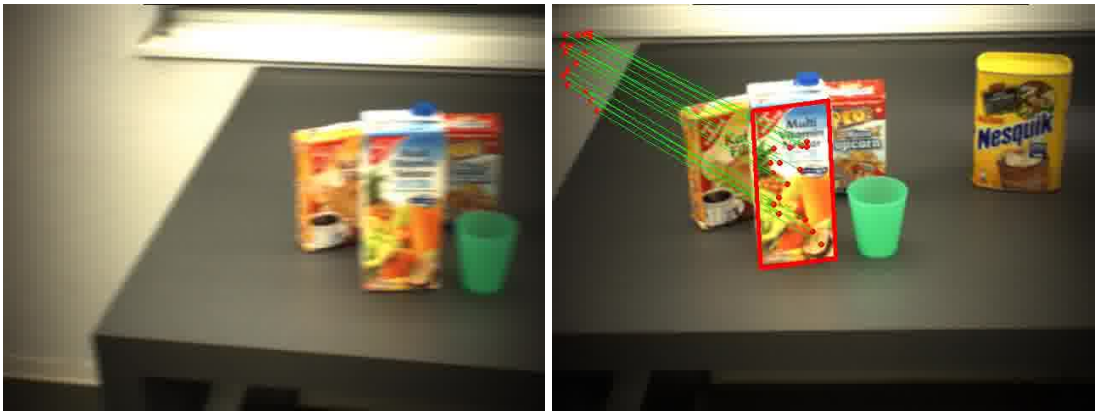
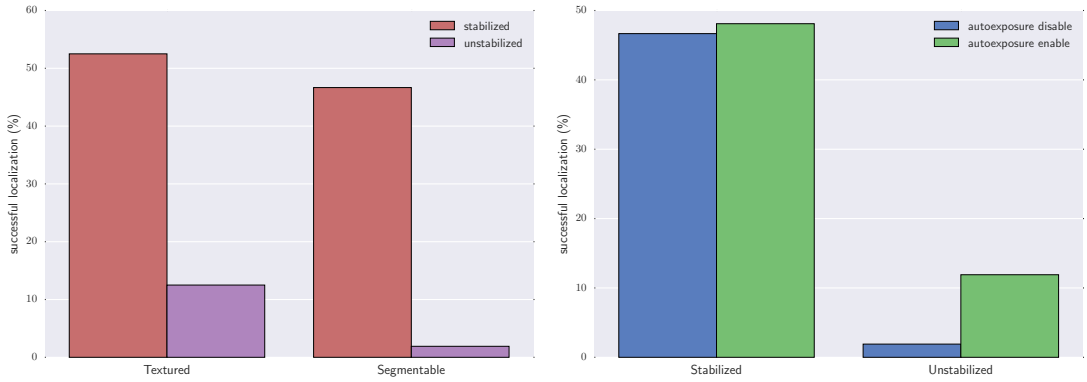


Figure 5.8.: Foveal camera images during self-induced motion of the robot. *Left:* textured object localization without gaze stabilization, and *right:* textured object localization with gaze stabilization. Images taken from Grotz et al. (2017a) (© 2017a IEEE).

Both experiments, with single colored and with textured objects, show that gaze stabilization significantly improves the object localization result during motion of the robot. Figure 5.9 reports the number of successful object localizations for the experiments. Gaze stabilization yields a significant increase in the number of successful localizations for both the textured and single colored (segmentable). Unstabilized, the object is successfully localized in only $13\,\%$ of all possible attempts for textured objects and in only $2\,\%$ for single colored objects respectively. The reason for the difference in the numbers can be explained by the fact that the detection of single colored objects is more prone to blurred images. The approach further depends on the current illumination which is changing when the robot is moving. Further, the recognition method for textured objects utilizes Scale Invariant Feature Transform (SIFT) and is therefore robust to local geometric distortion and also partially invariant to changes in illumination.

To support the robot's autonomy, it is preferred to adjust automatically the camera parameters for the current scene. The cameras offer an automatic mode for online control of autoexposure and other parameters. However, such adaption takes time if the illumination changes. Therefore, both object localization experiments were repeated with fixed settings and with the camera's automatic mode. Figure 5.9 depicts the number of successful object localization. Independent of using camera autoexposure, gaze stabilization increases the rate of successful localizations. The camera images shown in Figure 5.8 clearly depict a difference in the observed perceptual blur between the unstabilized and the stabilized case. Crete et al. (2007) propose a no-reference image quality met-



(a) Successful localization of textured and single colored (segmentable) objects with and without stabilization.

(b) Independent of using camera autoexposure, gaze stabilization improves the number of successful localization results.

Figure 5.9.: Number of successful object localization results. The experiment setup is illustrated in Figure 5.7. Left figure taken from Grotz et al. (2017a) (© 2017a IEEE).

ric to quantify the perceptual blur of images without requiring a baseline. The metric first blurs a given image artificially and then compares the variations between neighboring pixels between the images. The underlying principle is that neighboring pixels change with a large variation when the input image is only slightly blurred. A lower value of the metric indicates to a low perceptual blur in the image, while a high value corresponds to a high perceptual blur in the image. In addition, the work of Crete et al. (2007) also provides a correlation between subjective tests and the perceptual blur measured by the given metric. Figure 5.10 plots measured perceptual blur during the experiment. The plotted results indicate that the camera images for the stabilized case are significantly less blurry than for the unstabilized case. Unstabilized, the no-reference image quality metric shows that the image quality is degraded by the induced perturbation of the torso joint.

5.4.2. Grasping While Moving Experiment

The previous section shows a successful integration of gaze stabilization controller presented in Habra et al. (2017). Further, the object localization experiments using foveal vision demonstrate that gaze stabilization is required when moving. This experiment evaluates the gaze stabilization controller and the link to active vision in a complex real world application. Again, the humanoid

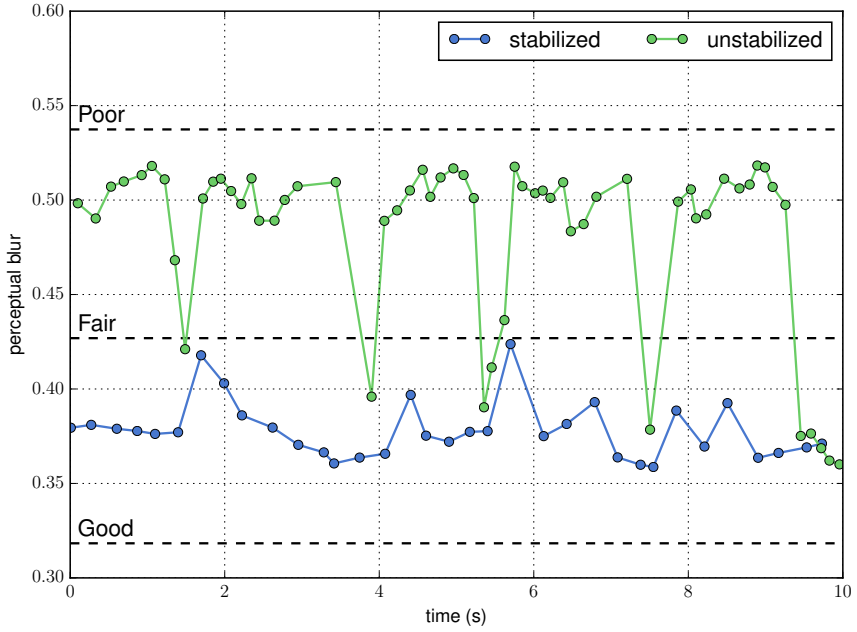


Figure 5.10.: Perceptual blur for the right foveal camera images during the object localization experiment. Higher values indicate a higher perceptual blur. Three subjective image quality levels are shown with a dashed line using the correlation provided by Crete et al. (2007). Plot taken from Grotz et al. (2017a) (© 2017a IEEE).

robot ARMAR-III is placed in a kitchen environment. The robot moves along a planned trajectory that passes close to a table with a green cup. The goal of the experiment is to grasp a cup while moving (see Figure 5.11). Mansard et al. (2007) conducted a similar experiment. In their work, the focus was more on control using a *stack of tasks* approach (Mansard and Chaumette, 2007), rather than combining active vision and gaze stabilization. In addition to the component that models object localization uncertainty by computing a saliency value, a second component draws the attention to single colored blobs in the scene. To this end, camera images are segmented and for the center of each segment a constant saliency value is added to the projected point on the sphere.

The position of the cup is unknown to the robot. In order to grasp the cup while moving an accurate object localization is therefore important. Hence, the foveated cameras have to be utilized and the object localization processes the foveal camera images. However, since the field of view for the foveal camera is very narrow, the gaze has to be shifted to relevant regions. The horizontal field of view of the foveal camera is $16\,\mathrm{deg}$ and the vertical field of view is $8\,\mathrm{deg}$ respectively. For the peripheral cameras the horizontal and a vertical field of

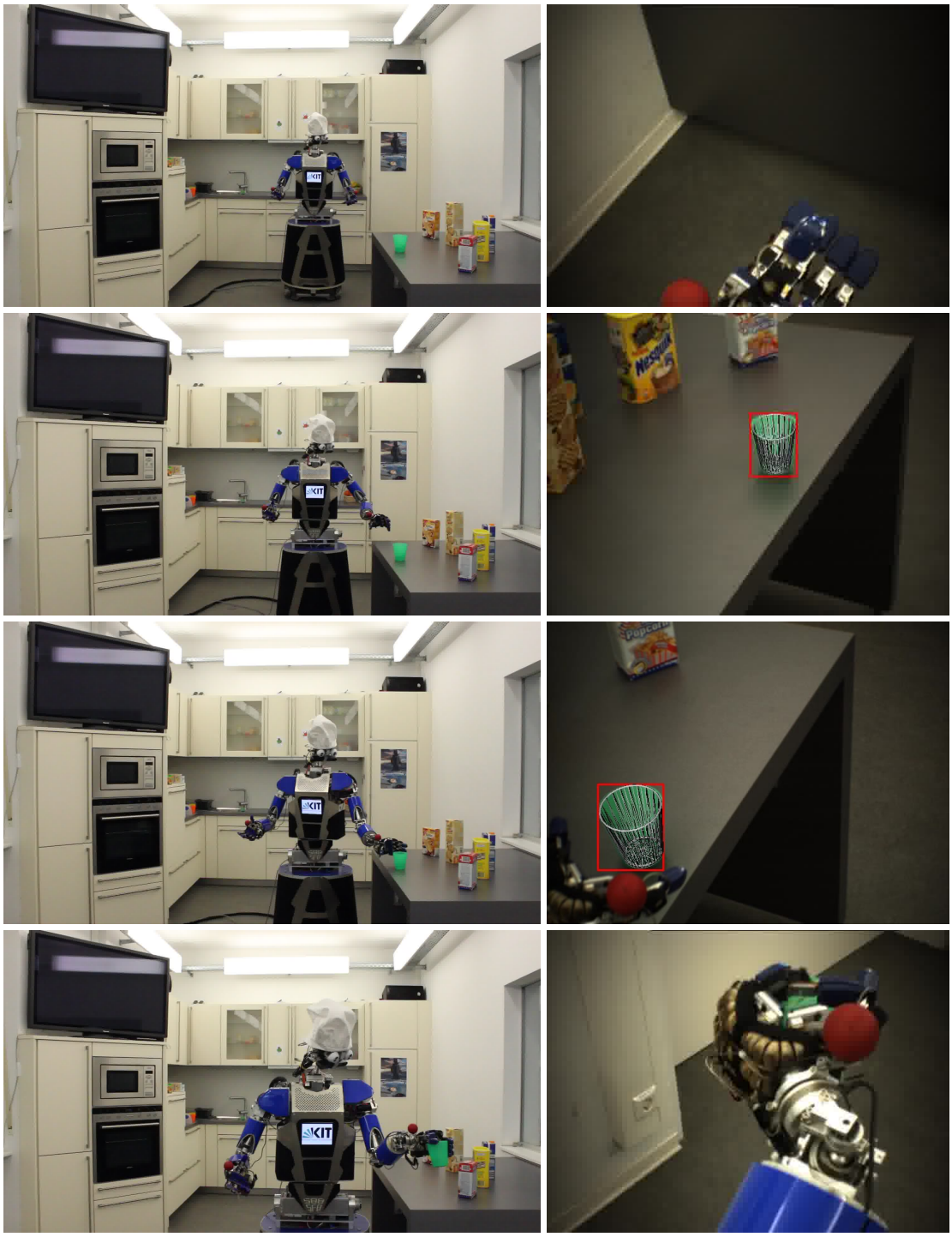


Figure 5.11.: Object grasping while moving. The left column shows an external view of the robot, while the right column shows the images of the right foveal camera during the experiment. Images taken from Grotz et al. (2017a) (© 2017a IEEE).

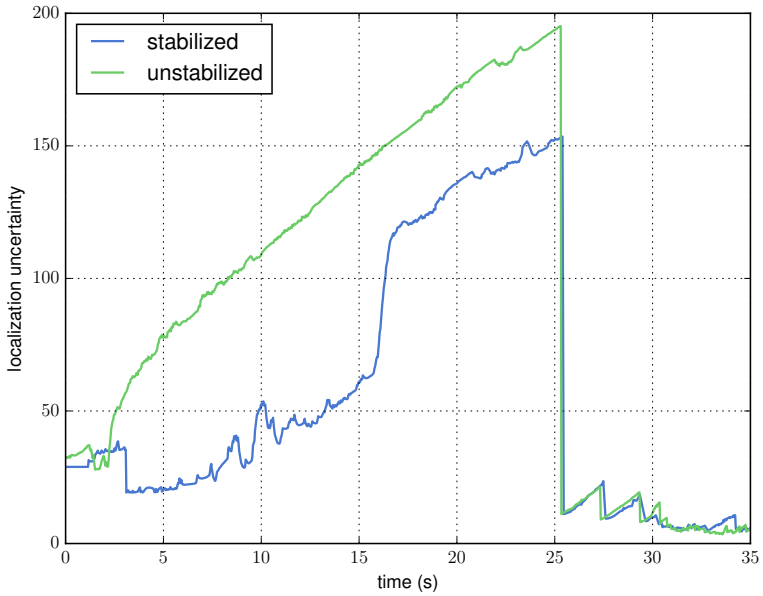


Figure 5.12.: Scalar value of the object localization uncertainty using the foveal cameras with and without stabilization as defined in Equation 5.2. After $25\,\mathrm{s}$, the object is close enough and the visual servoing strategy increases the task acuity. Thus, the active vision system shifts the gaze from the hand towards the known position of the cup. Once the cup is again in the field of view it can be localized and thus the uncertainty of the object localization drops. Figure taken from Grotz et al. (2017a) (© 2017a IEEE).

view is $60\,\mathrm{deg} \times 45\,\mathrm{deg}$. Given the larger field of view and the position, the cup is visible in the peripheral cameras. This allows for the active vision system to compute a saliency value based on the color value using the peripheral camera images. Since no other object is localized so far, the view selection shifts the attention of the robot and thereby the focus of the foveal camera to colored regions in the scene. From a cognitive perspective, the robot searches for color blobs in the peripheral view while trying to localize the green cup in the foveal cameras. As described in Section 5.4.1 the gaze stabilization controller stabilizes the view and thereby allows to leverage foveated cameras for object localization.

The grasping processes uses a position-based visual servoing to grasp the object of interest, i. e., the green cup. Visual serving (Chaumette and Hutchinson, 2006) uses visual feedback information to control the motion of a robot and it allows to resolve kinematic inaccuracies of a robot. Here, the implementation provided in Vahrenkamp et al. (2008) is utilized. The grasping pose is known a priori to the robot and stored in the robot's memory as part of the object rep-

Dense Optical Flow RMSE			
	Stabilized	Unstabilized	Decrease (%)
std	$0.87\,\mathrm{deg/s}$	$1.71\,\mathrm{deg/s}$	$-49.12\,\%$
mean	$1.01\,\mathrm{deg/s}$	$2.06\,\mathrm{deg/s}$	$-50.97\,\%$
max	$3.41\,\mathrm{deg/s}$	$10.49\,\mathrm{deg/s}$	$-67.49\,\%$

Table 5.1.: *Standard deviation (std)*, *mean* value and, *max* value of the optical flow root mean square error (RMSE). Table taken from Grotz et al. (2017a) (© 2017a IEEE). The dense optical flow is computed using the methods described in Farnebäck (2003).

resentation. Knowing the object pose, the grasping pose can be translated to a pose in world or robot coordinates. If the final grasping position is reached by the end-effector the robot executes the grasp on the object. The visual servoing controls the position of the robot's end-effector until the grasping pose is reached. Due to inaccuracies in the robot's kinematics and the object localization result, the visual servoing control loop requires to constantly localize the end-effector and the green cup. The visual servoing also prioritizes the end-effector and the object being grasped for the view selection. However, the platform movement following the predefined trajectory is too fast for the robot to reach the final grasping pose in time. Additionally, the object is out of reach and the visual servoing control loop cannot be started too early since it would position the end-effector at the wrong position. Hence, for this experiment the visual servoing is therefore slightly modified to cope with the movement of the robot. To start positioning the end-effector in advance, the object localization result is projected to a reachable pose within the robot's base coordinate system using the speed of the robot and the object localization result. The projected position intuitively corresponds to the expected position at the time, if the robot would be able to reach and grasp the object. Consequently, visual servoing runs with the projected pose in order to continuously align the end-effector. Once the robot is able to reach the object, the projection step is no longer necessary and the object's pose is used for visual servoing. This allows to position the hand and thus the grasping pose can later be reached more quickly. The view selection is required for this experiment to work since the foveal camera's field

of view is too narrow in order to localize the object or the end-effector during motion.

Figure 5.11 shows the camera view and an external view of the experiment at selected key frames. To show the support of gaze stabilization, the experiment was repeated without the gaze stabilizing controller. Figure 5.12 shows the object localization uncertainty. The view selection strategy combined with the gaze stabilization controller reduces the object localization uncertainty significantly. Table 5.1 reports the mean values of the optical flow Root-mean-square error (RMSE). Stabilized, the average optical flow in the camera images is reduced by more than $50\,\%$.

5.5. Summary

This chapter extends the system architecture of the active vision system by integrating a gaze stabilization controller. The experiments of this chapter show that gaze stabilization is required for visual perception during motion. The active vision system defines the next view target. The integrated gaze stabilization controller fixates the given view target and stabilizes the view target during motion.

Experiments show, that by keeping the visual target stable, the object localization is able to localize the object more often using the foveal cameras. Furthermore, the cameras are able to adapt to the environment more quickly using the preferred automatic mode for internal camera parameters. Further, the experiments show that gaze stabilization facilities the robust execution on vision-based algorithms by providing stable camera images. A complex real world scenario, where a humanoid robot was able to grasp an object while moving, demonstrates the successful integration and interaction of the different components.
Overall, the chapter shows that an active vision system requires both the determination of the NBV and also a gaze stabilization controller to fully support vision-based tasks during motion.

6. Conclusion and Perspective

This chapter briefly reviews the major contributions of this thesis and discusses the impact of the approach for semantic scene understanding. The contributions are put into context with the goals formulated in the thesis' introduction. Furthermore, the chapter gives a brief overview of future research directions, based on the research results obtained in this work.

6.1. Contributions

This thesis contributed a novel active vision method for the semantic perception of unknown environments. A semantic scene representation is created, which is extended by successively planning the Next-Best-View (NBV). By changing the robot's gaze to the Next-Best-View (NBV), the semantic scene representation is constantly updated. The methods were designed to explore unstructured and unknown scenes and to provide a semantic scene understanding of the robot to reason about interaction possibilities with the world. Therefore, the planning of the Next-Best-View (NBV) considers semantic information. Furthermore, a biologically motivated approach for gaze stabilization is presented and integrated into the system architecture to enable visual perception during motion. Comprehensive evaluations of the proposed methods showed the successful application and the benefit of using active vision methods. The following sections summarize the major achievements that have been evaluated and implemented.

Semantic Scene Representation

A robot can generate a semantic scene model using an RGB-D sensor aligned with the current gaze direction. Geometric primitives are fitted against the point cloud of the current view. Scene elements and objects are represented by

these geometric primitives, comprising cuboids, cylinders, and spheres. Having this geometric representation, spatial and physically plausible support relations between the geometric primitives are then inferred and other semantic information is extracted. Relations include spatial or support and stability. The semantic scene model was iteratively improved by fusing results from consecutive views into a global consistent scene model.

Next-Best-View Planning

The scene model was extended with an active vision method to determine the Next-Best-View. The approach allows humanoid robots to explore unknown scenes. The thesis presented a novel method that utilizes the semantic information of the extracted scene model for exploration. Planning the Next-Best-View allows to complete the scene model. Therefore, possible views are sampled based on the semantic information available in the scene representation. A view comprises of the robot's position and gaze direction. In a subsequent step, the sampled views are then evaluated with a utility function, that takes traveled distance and the volumetric information gain into account. By choosing the view maximizing the utility function, the number of views necessary to interpret the scene is minimized. A comprehensive, quantitative evaluation of the proposed methods in simulation, as well as on humanoid robots, show a significant improvement of the scene model compared to standard Next-Best-View (NBV) approaches.

View Selection and Gaze Stabilization

One of the goals of this thesis is to allow perception during motion of the robot. Gaze stabilization methods have been ported to the ARMAR humanoid robots and evaluated in task-oriented settings. Gaze stabilization yields a significant decrease in the perceived perpetual blur and thus improves the camera image quality. Experiments show further that it is necessary to link gaze stabilization with active vision methods in order to allow perception during motion of the robot. The active vision methods define the next visual target for the gaze stabilization controller. By keeping the visual target centered in the field of view vision-based components achieve much higher success. Finally, a comprehensive experiment shows that a humanoid robot is able to successfully grasp an object while moving.

6.2. Perspective

The thesis focused on a semantic scene representation supported by an active vision method. Concepts and methods can also be transferred to further fields of application. The following outlines a variety of possible future directions and extensions to this thesis.

Dynamic scenes

The presented semantic scene model has the major limitation of a static scene that needs to be considered. Regarding future research directions, the extension to dynamic scenes is an interesting research area. The changes in the scene could then be utilized as a hint for salient regions. Other research has already introduced attention based methods that take the motion of the human hand into account (Monica et al., 2016). Having a dynamic scene model would further allow the robot to observe and to validate the success of manipulation actions.

Human action recognition and joint visual attention

While the focus on this thesis was on semantic scene understanding, other task relevant information can be taken into account. A promising research direction would be to extend the active vision system to also include multimodal cues. This can also include action recognition (Dreher et al., 2019) and anticipation of the next action (Koppula and Saxena, 2015). Knowing the human's pose and gaze can also be used as a visual hint to shift the attention (joint visual attention). This is especially important for human-robot interaction. For example, joint visual attention would allow the system to include feedback from the human (Schauerte, 2016). The Next-Best-View system has been developed with the explicit intention of taking other information sources into account. A crucial step would be then to investigate how to balance the different task dependent goals in the utility function and how to priorities them.

Human inspired gaze control

The architecture presented in this thesis can be extended with a gaze controller that is biologically inspired. The gaze controller should be able to differentiate between saccade execution, and other types of human gaze movements. Furthermore, it should consider real-time constraints. Currently, the control loop of ARMAR-III is 100 Hz. In humans, the saccade execution is much faster and this would require new control strategies.

Industrial applications

Notably, the approach can be transferred to industrial applications. Up to now, robots in factories performing monotonous assembly tasks, often have a fixed camera position. Recent robotic systems, include cameras, but often use static views when executing pick and place tasks. Here, a smart approach with active vision could improve not only the model quality but also reduce the overall time to perceive the objects. Finally, in combination with gaze stabilization, robots could also detect objects and grasp while moving.

Appendices

A. Fundamentals

The chapter describes briefly fundamentals required for the methods presented in this thesis. Without loss of generality, the representation is restricted to the Euclidean space $\mathbb{R}^3$.

A.1. The Special Orthogonal Group

The group for all rotation matrices in the 3D space is denoted as

$$SO(3) \doteq \{R \in \mathbb{R}^{3\times3} | R^\mathsf{T} R = I, det(R) = 1\} \ . \qquad \text{(special orthogonal group)}$$

where I is the identity matrix. Thus multiplication of two rotation matrices leads to a new rotation matrix. Rotation matrices for a rotation around the X-, Y-, Z-axis are denoted with R_x, R_y and R_z.

A.2. The Special Euclidean Group

The special euclidean group $SO(3)$ describes poses, consisting of orientation and translation and movements of rigid object. The special euclidean group $SE(3)$ is defined with

$$SE(3) \doteq \left\{ \begin{pmatrix} R & \boldsymbol{t} \\ 0 & 1 \end{pmatrix} \subset \mathbb{R}^{4\times4} | R \in SO(3), \boldsymbol{t} \in \mathbb{R}^3 \right\} \ . \quad \text{(special euclidean group)}$$

A.3. Differential Entropy

The differential entropy is the entropy of a continuous random variable (Cover and Thomas, 2006). In general, the entropy is a measure of average *surprisal* of a random variable. Formally, the differential entropy for a N-dimensional multivariate normal distribution is defined as follows. Let $x \sim \mathcal{N}(\mu, \Sigma)$, then the differential entropy u is expressed as

$$u = \frac{1}{2} \ln \left(\left(2\pi e\right)^N det(\Sigma) \right) \ . \tag{A.1}$$

A.4. Point Cloud Representation

As already mentioned sensor data can be represented via points. Here, a point cloud is a discrete set of points with respect to a fixed coordinate system. Knowing the intrinsic and extrinsic camera parameters the RGB-D image can be translated into a point cloud. The 2.5D information can be translated to a 3D point in the camera coordinate system. The transformation is in line with the concept of animate vision (Ballard, 1991) as described in Chapter 2. Without loss of generality, for each 3D point $\boldsymbol{p}_i$ it is assumed that information about the color is available, i. e.,

$$\boldsymbol{p}_i = (x, y, z, r, g, b) \ , \tag{A.2}$$

where $(r, g, b) \in [0 \dots 255]^3 \subset \mathbb{N}_0^3$ is the color information encoded in the RGB color space and $\boldsymbol{t} = (x, y, z) \in \mathbb{R}^3$ the spatial information. Thus, a point cloud $\mathcal{P}$ can be formally written as set

$$\mathcal{P} = \{ \boldsymbol{p}_1, \dots \boldsymbol{p}_n \} \ , \tag{A.3}$$

where n is the total number of points. A point cloud also contains other information, most importantly the time t when the sensor data was captured. Here the time of a point cloud is indicated as $\mathcal{P}^t$. For brevity, the temporal parameter t is omitted if it can be derived from the context. In the following, the normal estimation $\hat{n}(\boldsymbol{p}_i, r)$ is defined as a function

$$\hat{n}(\boldsymbol{p}_i, r) : \mathbb{R}^3 \to \mathbb{R}^3 \tag{A.4}$$

$$(x, y, z) \mapsto (n_1, n_2, n_3) \tag{A.5}$$

Figure A.1.: *Left:* Using the robot's odometry only. *Right:* Registration with ElasticFusion.

that maps a point $\boldsymbol{p}_i \in \mathcal{P}$ to its computed normal vector (n_1, n_2, n_3). The parameter r specifies the radius used by the normal estimation method. Normals of a point cloud can hint at geometric constraint and are essential for the geometric primitive extraction as described in Chapter 3.

A.5. View Registration

For perception from multiple view it is important that each view is registered with respect to each other. That means that for consecutive views the transformation between each view needs to be estimated. The problem is known as Simultaneous Localization and Mapping (SLAM). Once the relative transformation between each view is determined the captured point cloud of each single view can be aggregated into a global consistent point cloud. Figure A.1 visualizes multiple views. Due to inaccuracies in the robot's kinematics the robot's forward kinematics give an estimate of the current camera pose, but are not sufficient for an accurate registration between two consecutive view poses. Hence, the registration between each view must be refined. A general algorithm to align two point clouds is the Iterative Closest Point (ICP) algorithm, which optimizes a transformation between two clouds by minimizes the error between the points. However, Iterative Closest Point (ICP) has some disadvantages. In this work, the state-of-the-art methods ElasticFusion (Whelan et al., 2017) or RTABMap (Labbe and Michaud, 2013) are utilized to register views if not stated otherwise. Both methods have real-time capabilities and work with RGB-D sensor data. Figure A.1 shows an example of a registration result.

A.6. Point Cloud Segmentation

Segmentation is to split a point cloud $\mathcal{P}$ into plausible disjunct regions $\mathcal{P}_i$, such that:

$$\mathcal{P} = \bigcup_i \mathcal{P}_i \ . \tag{A.6}$$

For each point in the point cloud a label is assigned. Points sharing similar characteristics are assigned the same label to represent the data in a more meaningful manner. More formally, a segmentation $\mathcal{S}(\mathcal{P})$ of a given point cloud $\mathcal{P}$ is defined as

$$\mathcal{P} = \{\, \mathcal{P}_1, \ldots \mathcal{P}_n \,\}, \mathcal{P}_i \subseteq \mathcal{P}, \forall i \neq j : \mathcal{P}_i \cap \mathcal{P}_j = \emptyset \ . \tag{A.7}$$

Segments $\mathcal{P}_i$ are regions of interest and hint at possible objects in the scene. Figure A.2 shows an example.

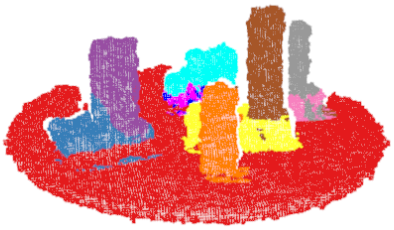


Figure A.2.: A segmented point cloud. The point cloud is shown in Figure A.1.

B. Robotic Platforms and Sensors

This chapter lists the robotic platforms and sensors mainly used in the experiments. Experiments in this thesis are conducted using humanoid robots. In particular, robots of the Karlsruhe ARMAR Humanoid Robot Family (Asfour et al., 2019a); namely, the humanoid robots ARMAR-III (Asfour et al., 2006) and ARMAR-6 (Asfour et al., 2018, 2019b). Both robots have a holonomic platform and feature an anthropomorphic design. While ARMAR-III has more degrees of freedom in the head, ARMAR-6 offers a wider variety of perception sensors. The following gives an overview of important aspects of both robotic platforms and sensors for perception.

B.1. The Humanoid Robot ARMAR-III

The humanoid robot ARMAR-III (Asfour et al., 2006) has an overall 43 Degrees of Freedom (DoF) and is designed for various daily kitchen activity and handover tasks. A holonomic platform enables this robot to navigate through indoor rooms and to rotate on the spot. The robot can localize itself using three laser scanners mounted near the platform. The robot's head features seven DoF in total and is also available as a stand-alone version, known as the Karlsruhe Humanoid Head (Asfour et al., 2008). More specifically, ARMAR-III's head has four DoF in the neck and three DoF in the eyes for common tilt and independent pan eye movements. Adding more degrees of freedom than the minimal requirement allows to control the head redundantly. To mimic human foveal and peripheral vision, each eye of ARMAR-III contains two *Point Grey Dragonfly 2* cameras placed next to each other. Camera lenses are detached from the camera module due to the limited space and thus allow for more compact design of the eye and thus flexible eye movements. The first camera has a wide angle lens for peripheral vision, while the second camera has a narrow angle lens for foveal vision. Each camera has a resolution of $640\,\mathrm{px} \times 480\,\mathrm{px}$. The cameras deliver a framerate up to $30\,\mathrm{Hz}$ sufficient for most vision-based tasks,

Figure B.1.: The humanoid robot ARMAR-III in a kitchen environment. Figure taken from (Asfour et al., 2019a) (© 2019a Springer).

such as object localization or object tracking. The ARMAR-III robotic platform has been widely used, also in large scale applications (Wächter et al., 2018). For more details about the hardware layout and cognitive abilities, the reader is referred to Asfour et al. (2006) and Asfour et al. (2008).

B.2. The Humanoid Robot ARMAR-6

The humanoid robot ARMAR-6 is designed as a collaborative agent for maintenance tasks in industrial environments (Asfour et al., 2018). Asfour et al. (2019b) gives a more comprehensive overview of ARMAR-6's capabilities. Similar to ARMAR-III the humanoid robot uses a holonomic platform as well. A linear torso joint allows to adjust the height of the robot and thus ARMAR-6 can spawn up to a maximum height of $1.9\,\mathrm{m}$. One crucial aspect when working in industrial environments is the visual perception. Therefore, the robot is endowed with several perception sensors. Figure B.2 outlines the robot's

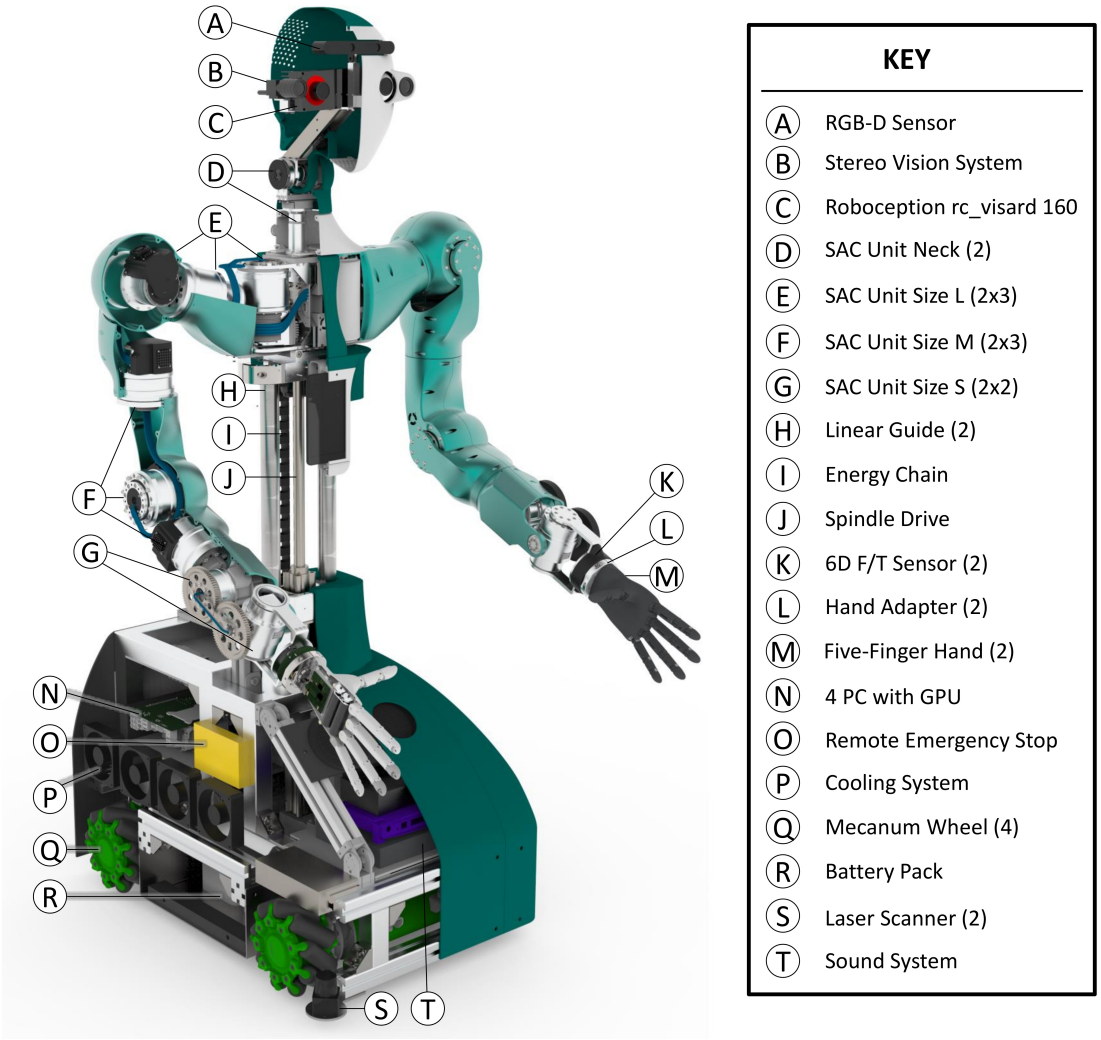


Figure B.2.: Sensor overview of the humanoid robot ARMAR-6. Figure taken from Asfour et al. (2019b) (© 2019b IEEE).

structure and its sensor system. Inspired by previous generations of the Karlsruhe ARMAR Humanoid Robot Family (Asfour et al., 2019b), ARMAR-6's features two *Point Grey Flea3* cameras for foveated vision and a Roboception rc_visard 160 color sensor for stereo vision and on-board depth image computation. The cameras used for foveal vision feature an image resolution up to $1600\,\text{px} \times 1200\,\text{px}$ resolution and framerate up to $60\,\text{Hz}$. Additionally, ARMAR-6's head features a *Primesense Carmine 1.09* RGB-D sensor with a $640\,\text{px} \times 480\,\text{px}$ resolution and framerate up to $30\,\text{Hz}$.

Glossary

Active Vision

Purposefully control the camera parameters and the position of the camera to improve the current visual perception. *See Section 2.1.1*.

Affordance

Interaction possibilities with scene elements or objects for an agent. *See Section 3.2.3*.

Next-Best-View

Iteratively determine the next-best-view with respect to a function. Mainly used for autonomous object modelling tasks and scene exploration.

Robot Development Environment

Software environment, which allows the software development and integration for robots.

SLAM

Build a map of the current environment while at the same time track a robot's location within it.

Support Graph

A graph that models the physical support among objects.

List of Acronyms

DoF	Degrees of Freedom
FOV	Field of View
FPS	Frame per Second
GPU	Graphics Processing Unit
ICP	Iterative Closest Point
IK	Inverse Kinematics
IMU	Inertial Measurement Unit
IOR	Inhibition of Return
LCCP	Locally Convex Connected Patches
NBV	Next-Best-View
OBB	Oriented Bounding Box
OKR	Optokinetic Reflex
PCL	Point Cloud Library

PoI	Point of Interest
RANSAC	Random Sample Consensus
RDE	Robot Development Environment
RGB-D	Red, Green, Blue and Depth
RMSE	Root-mean-square error
RRT	Rapidly Exploring Random Tree
SG	Support Graph
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SVM	Support Vector Machine
TSDF	Truncated Signed Distance Function
VCR	Vestibulo-Collic Reflex
VOR	Vestibulo-Ocular Reflex
WTA	Winner-Takes-All

List of Symbols

$\lvert\cdot\rvert$	The cardinality, i. e., the number of elements, of a set.
det	The determinant of a matrix.
ψ	A geometric primitive.
$\mathcal{N}$	Multivariate normal distribution $\mathcal{N}(\mu, \sigma^2)$ of random vector, where μ denotes the mean and σ^2 the variance.
$\lVert\cdot\rVert_p$	p-norm $\lVert x\rVert_p := \left(\sum_{i=1}^{n} \lvert x_i\rvert^p\right)^{1/p}$
$\mathcal{P}$	A point cloud, i. e., a set of points
$SE(3)$	The special euclidean group.
$SO(3)$	The special orthogonal group.
t	A timestamp, identifying when a certain event occurred. Usually in conjunction with sensor data, e. g., $\mathcal{P}^t$ denotes the point cloud captured at time t.
$\mathcal{U}$	Uniform distribution $\mathcal{U}(a, b)$ of random vector in the interval $[0, 1]$.

$v \in SE(3)$	A view defined by the coordinate system of the visual sensor.
$v_{target} \in \mathbb{R}^3$	A view target or fixation point defined in the world or the robot's coordinate system.

List of Figures

List of Tables

List of Algorithms

Bibliography

Aloimonos, J. (1990). Purposive and qualitative active vision. In *[1990] Proceedings. 10th International Conference on Pattern Recognition*, pages 346–360. IEEE Comput. Soc. Press. *Cited on pages 14, 30, and 32.*

Aloimonos, J., Weiss, I., and Bandyopadhyay, A. (1988). Active vision. *International Journal of Computer Vision*, 1(4):333–356. *Cited on pages 1, 11, 12, 13, and 17.*

Ammirato, P., Poirson, P., Park, E., Kosecka, J., and Berg, A. C. (2017). A dataset for developing and benchmarking active vision. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1378–1385. IEEE. *Cited on page 31.*

Archibald, S. (2008). Ways of seeing - alfred yarbus's science of visual attention. *Cited on page 12.*

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., and Savarese, S. (2016). 3d semantic parsing of large-scale indoor spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1534–1543. IEEE. *Cited on page 25.*

Arruda, E., Wyatt, J., and Kopicki, M. (2016). Active vision for dexterous grasping of novel objects. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2881–2888. *Cited on page 31.*

Asfour, T. (2019). Lecture notes in robotics iii: Sensors and perception in robotics. *Cited on page 18.*

Asfour, T., Dillmann, R., Vahrenkamp, N., Do, M., Wächter, M., Mandery, C., Kaiser, P., Kröhnert, M., and Grotz, M. (2019a). The karlsruhe ARMAR humanoid robot family. In *Humanoid Robotics: A Reference*, pages 337–368. Springer Netherlands. *Cited on pages 119 and 120.*

Asfour, T., Kaul, L., Wachter, M., Ottenhaus, S., Weiner, P., Rader, S., Grimm, R., Zhou, Y., Grotz, M., Paus, F., Shingarey, D., and Haubert, H. (2018). ARMAR-6: A collaborative humanoid robot for industrial environments. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 447–454. IEEE. *Cited on pages 1, 119, and 120.*

Asfour, T., Regenstein, K., Azad, P., Schroder, J., Bierbaum, A., Vahrenkamp, N., and Dillmann, R. (2006). ARMAR-III: An integrated humanoid platform for sensory-motor control. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 169–175. *Cited on pages 1, 30, 60, 119, and 120.*

Asfour, T., Schill, J., Peters, H., Klas, C., Bucker, J., Sander, C., Schulz, S., Kargov, A., Werner, T., and Bartenbach, V. (2013). ARMAR-4: A 63 dof torque controlled humanoid robot. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 390–396. *Cited on page 44.*

Asfour, T., Wächter, M., Kaul, L., Rader, S., Weiner, P., Ottenhaus, S., Grimm, R., Zhou, Y., Grotz, M., and Paus, F. (2019b). ARMAR-6: A high-performance humanoid for human-robot collaboration in real world scenarios. *IEEE Robotics & Automation Magazine*, 26(4):108–121. *Cited on pages 1, 2, 60, 119, 120, and 121.*

Asfour, T., Welke, K., Azad, P., Ude, A., and Dillmann, R. (2008). The Karlsruhe Humanoid Head. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 447–453. *Cited on pages 12, 30, 45, 119, and 120.*

Azad, P., Asfour, T., and Dillmann, R. (2009). Accurate shape-based 6-dof pose estimation of single-colored objects. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2690–2695. *Cited on pages 92 and 100.*

Azad, P., Gockel, T., and Dillmann, R. (2007). *Computer Vision - das Praxisbuch*. Elektor-Verlag. *Cited on pages 92 and 100.*

Bajcsy, R. (1988). Active perception. *Proceedings of the IEEE*, 76(8):996–1005. *Cited on pages 1, 11, 13, and 17.*

Bajcsy, R., Aloimonos, Y., and Tsotsos, J. K. (2018). Revisiting active perception. *Autonomous Robots*, pages 177–196. *Cited on pages 11, 15, 16, and 17.*

Bajcsy, R. and Campos, M. (1992). Active and exploratory perception. *CVGIP: Image Understanding*, 56(1):31–40. *Cited on page 39.*

Ballard, D. H. (1991). Animate vision. *Artificial Intelligence*, 48(1):57–86. *Cited on pages 11, 14, 17, 43, and 116.*

Banta, J. E., Wong, L. R., Dumont, C., and Abidi, M. A. (2000). A next-best-view system for autonomous 3-d object reconstruction. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 30(5):589–598. *Cited on pages 32, 38, 67, and 70.*

Beetz, M., Tenorth, M., and Winkler, J. (2015). Open-ease. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1983–1990. IEEE. *Cited on page 14.*

Berner, A., Li, J., Holz, D., Stuckler, J., Behnke, S., and Klein, R. (2013). Combining contour and shape primitives for object detection and pose estimation of prefabricated parts. In *2013 20th IEEE International Conference on Image Processing (ICIP)*, pages 3326–3330. *Cited on page 22.*

Biswas, J. and Veloso, M. (2012). Depth camera based indoor mobile robot localization and navigation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1697–1702. *Cited on page 24.*

Blodow, N., Goron, L. C., Marton, Z.-C., Pangercic, D., Ruhr, T., Tenorth, M., and Beetz, M. (2011). Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4263–4270. IEEE. *Cited on page 23.*

Bloomenthal, J. and Bajaj, C. (1997). *Introduction to implicit surfaces*. The Morgan Kaufmann series in computer graphics and geometric modeling. Morgan Kaufmann Publishers, San Francisco, Calif. *Cited on page 20.*

Bohg, J., Hausman, K., Sankaran, B., Brock, O., Kragic, D., Schaal, S., and Sukhatme, G. S. (2017). Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, pages 1–19. *Cited on pages 11, 15, and 17.*

Bohg, J., Welke, K., León, B., Do, M., Song, D., Wohlkinger, W., Madry, M., Aldóma, A., Przybylski, M., Asfour, T., Martí, H., Kragic, D., Morales, A., and Vincze, M. (2012). Task-based grasp adaptation on a humanoid robot. *IFAC Proceedings Volumes*, 45(22):779–786. *Cited on page 30.*

Borji, A. and Itti, L. (2013). State-of-the-art in visual attention modeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):185–207. *Cited on page 40.*

Chaumette, F. and Hutchinson, S. (2006). Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 13(4):82–90. *Cited on page 106.*

Chen, S., Li, Y., and Kwok, N. M. (2011). Active vision in robotic systems: A survey of recent developments. *The International Journal of Robotics Research*, 30(11):1343–1377. *Cited on pages 28 and 40.*

Cheng, R., Agarwal, A., and Fragkiadaki, K. (2018). Reinforcement learning of active vision for manipulating objects under occlusions. In Billard, A., Dragan, A., Peters, J., and Morimoto, J., editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 422–431. PMLR. *Cited on page 31.*

Connolly, C. (1985). The determination of next best views. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 432–435. *Cited on pages 32, 38, and 67.*

Corke, P., Lobo, J., and Dias, J. (2007). An introduction to inertial and visual sensing. *The International Journal of Robotics Research*, 26(6):519–535. *Cited on page 43.*

Corke, P. I. (2011). *Robotics, vision and control: Fundamental algorithms in MATLAB*, volume 73 of *Springer tracts in advanced robotics*. Springer, Berlin. *Cited on page 19.*

Cover, T. M. and Thomas, J. A. (2006). *Elements of information theory*. Wiley, 2nd ed. edition. *Cited on page 116.*

Crete, F., Dolmiere, T., Ladret, P., and Nicolas, M. (2007). The blur effect: Perception and estimation with a new no-reference perceptual blur metric. In *Human vision and electronic imaging XII*, volume 6492, pages 1–11. *Cited on pages 102, 103, and 104.*

Daudelin, J. and Campbell, M. (2017). An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-d objects. *IEEE Robotics and Automation Letters*, 2(3):1540–1547. *Cited on pages 35, 36, 37, 38, 39, and 129.*

Dreher, C. R. G., Waechter, M., and Asfour, T. (2019). Learning object-action relations from bimanual human demonstration using graph networks. *IEEE Robotics and Automation Letters*, page 1. *Cited on page 111.*

Eidenberger, R. and Scharinger, J. (2010). Active perception and scene modeling by planning with probabilistic 6d object poses. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1036–1043. IEEE. *Cited on page 30.*

Fang, H., Lafarge, F., and Desbrun, M. (2018). Planar shape detection at structural scales. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Cited on page 24.*

Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In Goos, G., Hartmanis, J., van Leeuwen, J., Bigun, J., and Gustavsson, T., editors, *Image Analysis*, volume 2749 of *Lecture Notes in Computer Science*, pages 363–370. Springer Berlin Heidelberg, Berlin, Heidelberg. *Cited on pages 43 and 107.*

Fiala, J. C., Lumia, R., Roberts, K. J., and Wavering, A. J. (1994). Triclops: A tool for studying active vision. *International Journal of Computer Vision*, 12(2-3):231–250. *Cited on page 12.*

Findlay, J. M. and Gilchrist, I. D. (2003). *Active vision: The psychology of looking and seeing*, volume 37 of *Oxford psychology series*. Oxford University Press, Oxford. *Cited on page 2.*

Fischler, M. A. and Bolles, R. C. (1987). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in Computer Vision*, pages 726–740. Elsevier. *Cited on page 51.*

Foissotte, T., Stasse, O., Escande, A., Wieber, P.-B., and Kheddar, A. (2009). A two-steps next-best-view algorithm for autonomous 3d object modeling by a humanoid robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1159–1164. IEEE. *Cited on pages 33 and 35.*

Frintrop, S. (2006). *VOCUS: A visual attention system for object detection and goal-directed search*. Thesis (ph.d.), University of Bonn, Berlin and London. *Cited on page 40.*

Frintrop, S. and Jensfelt, P. (2008). Attentional landmarks and active gaze control for visual slam. *IEEE Transactions on Robotics*, 24(5):1054–1065. *Cited on pages 14, 41, and 129.*

Frintrop, S., Rome, E., and Christensen, H. I. (2010). Computational visual attention systems and their cognitive foundations: A survey. *ACM Trans. Appl. Percept.*, 7(1):6–39. *Cited on page 40.*

Garcia, G. M., Potapova, E., Werner, T., Zillich, M., Vincze, M., and Frintrop, S. (2015). Saliency-based object discovery on rgb-d data with a late-fusion approach. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1866–1873. IEEE. *Cited on page 41.*

Gibson, J. J. (1979). *The ecological approach to visual perception*. Houghton Mifflin, Boston. *Cited on page 56.*

Gonzalez-Banos, H. H. and Latombe, J.-C. (2002). Navigation strategies for exploring indoor environments. *The International Journal of Robotics Research*, 21(10-11):829–848. *Cited on page 33.*

Grotz, M., Habra, T., Ronsse, R., and Asfour, T. (2017a). Autonomous view selection and gaze stabilization for humanoid robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1427–1434. IEEE. *Cited on pages 72, 92, 94, 98, 99, 101, 102, 103, 104, 105, 106, and 107.*

Grotz, M., Kaiser, P., Aksoy, E. E., Paus, F., and Asfour, T. (2017b). Graph-based visual semantic perception for humanoid robots. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 869–875. IEEE. *Cited on pages 49, 50, 52, 55, 57, 65, and 66.*

Grotz, M., Sippel, D., and Asfour, T. (2019). Active vision for extraction of physically plausible support relations. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 463–469. *Cited on pages 49, 67, and 83.*

Gualtieri, M. and Platt, R. (2017). Viewpoint selection for grasp detection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 258–264. IEEE. *Cited on page 31.*

Gupta, A., Efros, A. A., and Hebert, M. (2010). Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *Computer Vision – ECCV 2010*, volume 6314 of *Lecture Notes in Computer Science*, pages 482–496. Springer Berlin Heidelberg, Berlin, Heidelberg. *Cited on page 25.*

Habra, T. (2017). *Gaze stabilization of humanoid robots based on internal model*. PhD thesis, UCL - SST/IMMC/MEED - Mechatronic, Electrical Energy, and Dynamics Systems UCL - Ecole Polytechnique de Louvain. *Cited on pages 96 and 100.*

Habra, T., Grotz, M., Sippel, D., Asfour, T., and Ronsse, R. (2017). Multimodal gaze stabilization of a humanoid robot based on reafferences. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 47–54. IEEE. *Cited on pages 44, 46, 92, 96, 97, 98, and 103.*

Habra, T. and Ronsse, R. (2016). Gaze stabilization of a humanoid robot based on virtual linkage. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 163–169. IEEE. *Cited on pages 44, 46, 96, and 97.*

Hager, G. D. and Wegbreit, B. (2011). Scene parsing using a prior world model. *The International Journal of Robotics Research*, 30(12):1477–1507. *Cited on pages 23, 24, 29, and 129.*

Henry, P., Krainin, M., Herbst, E., Ren, X., and Fox, D. (2012). Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663. *Cited on page 20.*

Hermann, A., Drews, F., Bauer, J., Klemm, S., Roennau, A., and Dillmann, R. (2014). Unified gpu voxel collision detection for mobile manipulation planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4154–4160. IEEE. *Cited on page 21.*

Hoffman, E. M., Caron, S., Ferro, F., Sentis, L., and Tsagarakis, N. G. (2019). Developing humanoid robots for applications in real-world scenarios [from the guest editors]. *IEEE Robotics & Automation Magazine*, 26(4):17–19. *Cited on page 1.*

Horbert, E., Garcia, G. M., Frintrop, S., and Leibe, B. (2015). Sequence-level object candidates based on saliency for generic object recognition on mobile systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 127–134. *Cited on page 41.*

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206. *Cited on pages 21 and 74.*

Isler, S., Sabzevari, R., Delmerico, J., and Scaramuzza, D. (2016). An information gain formulation for active volumetric 3d reconstruction. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3477–3484. *Cited on pages 34, 35, 36, 37, 38, 74, and 129.*

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE transactions on pattern analysis and machine intelligence*, 20(11):1254–1259. *Cited on page 39.*

Izadi, S., Davison, A., Fitzgibbon, A., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., and Freeman, D. (2011). Kinectfusion. In Pierce, J., Agrawala, M., and Klemmer, S., editors, *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, page 559, New York, New York, USA. ACM Press. *Cited on page 20.*

Jaccard, P. (1912). The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50. *Cited on page 57.*

Jia, Z., Gallagher, A., Saxena, A., and Chen, T. (2013). 3d-based reasoning with blocks, support, and stability. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE. *Cited on pages 26 and 29.*

Kahn, G., Sujan, P., Patil, S., Bopardikar, S., Ryde, J., Goldberg, K., and Abbeel, P. (2015). Active exploration using trajectory optimization for robotic grasping in the presence of occlusions. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4783–4790. IEEE. *Cited on page 31.*

Kaiser, A., Ybanez Zepeda, J. A., and Boubekeur, T. (2018). A survey of simple geometric primitives detection methods for captured 3d data. *Computer Graphics Forum*, 38(1):167–196. *Cited on pages 22 and 49.*

Kaiser, P. (2018). *Whole-Body Affordances for Humanoid Robots: A Computational Approach*. KIT Scientific Publishing. *Cited on page 56.*

Kaiser, P., Aksoy, E. E., Grotz, M., and Asfour, T. (2016). Towards a hierarchy of loco-manipulation affordances. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2839–2846. IEEE. *Cited on page 56.*

Kaiser, P. and Asfour, T. (2018). Autonomous detection and experimental validation of affordances. *IEEE Robotics and Automation Letters*, 3(3):1949–1956. *Cited on page 53.*

Kaiser, P., Grotz, M., Aksoy, E. E., Do, M., Vahrenkamp, N., and Asfour, T. (2015a). Validation of whole-body loco-manipulation affordances for pushability and liftability. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 920–927. *Cited on pages 49 and 52.*

Kaiser, P., Vahrenkamp, N., Schültje, F., Borràs, J., and Asfour, T. (2015b). Extraction of whole-body affordances for loco-manipulation tasks. *International Journal of Humanoid Robotics*, page 1550031. *Cited on page 52.*

Kaneko, K., Kaminaga, H., Sakaguchi, T., Kajita, S., Morisawa, M., Kumagai, I., and Kanehiro, F. (2019). Humanoid robot hrp-5p: An electrically actuated humanoid robot with high-power and wide-range joints. *IEEE Robotics and Automation Letters*, 4(2):1431–1438. *Cited on page 1.*

Karaszewski, M., Adamczyk, M., and Sitnik, R. (2016). Assessment of next-best-view algorithms performance with various 3d scanners and manipulator. *ISPRS Journal of Photogrammetry and Remote Sensing*, 119:320–333. *Cited on pages 38 and 74.*

Kartmann, R., Paus, F., Grotz, M., and Asfour, T. (2018). Extraction of physically plausible support relations to predict and validate manipulation action effects. *IEEE Robotics and Automation Letters*, 3(4):3991–3998. *Cited on pages 27, 29, 49, 53, and 55.*

Kasper, A., Xue, Z., and Dillmann, R. (2012). The kit object models database: An object model database for object recognition, localization and manipulation in service robotics. *The International Journal of Robotics Research*, 31(8):927–934. *Cited on page 101.*

Kemp, C. C., Fitzpatrick, P., Hirukawa, H., Yokoi, K., Harada, K., and Matsumoto, Y. (2008). Humanoids. In Siciliano, B. and Khatib, O., editors, *Springer Handbook of Robotics*, pages 1307–1333. Springer Berlin Heidelberg, Berlin, Heidelberg. *Cited on page 1.*

Koch, C. and Ullman, S. (1987). Shifts in selective visual attention: Towards the underlying neural circuitry. In Vaina, L., editor, *Matters of Intelligence*, volume 188 of *Synthese Library*, pages 115–141. Springer Netherlands. *Cited on page 39.*

Koppula, H. and Saxena, A. (2015). Anticipating human activities using object affordances for reactive robotic response. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):14–29. *Cited on page 111.*

Kriegel, S., Brucker, M., Marton, Z.-C., Bodenmuller, T., and Suppa, M. (2013). Combining object modeling and recognition for active scene exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2384–2391. IEEE. *Cited on pages 34 and 35.*

Kriegel, S., Rink, C., Bodenmüller, T., and Suppa, M. (2015). Efficient next-best-scan planning for autonomous 3d surface reconstruction of unknown objects. *Journal of Real-Time Image Processing*, 10(4):611–631. *Cited on page 36.*

Kryczka, P., Falotico, E., Hashimoto, K., Lim, H.-o., Takanishi, A., Laschi, C., Dario, P., and Berthoz, A. (2012). A robotic implementation of a bio-inspired head motion stabilization model on a humanoid platform. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2076–2081. *Cited on pages 44 and 46.*

Kuniyoshi, Y., Kita, N., Suehiro, T., and Rougeaux, S. (1996). Active stereo vision system with foveated wide angle lenses. In Li, S. Z., Mital, D. P., Teoh, E. K., and Wang, H., editors, *Recent Developments in Computer Vision*, pages 191–200, Berlin, Heidelberg. Springer Berlin Heidelberg. *Cited on page 12.*

Labbe, M. and Michaud, F. (2013). Appearance-based loop closure detection for online large-scale and long-term operation. *IEEE Transactions on Robotics*, 29(3):734–745. *Cited on page 117.*

Lavalle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning. Technical report. *Cited on page 75.*

Li, Y., Wu, X., Chrysathou, Y., Sharf, A., Cohen-Or, D., and Mitra, N. J. (2011). Globfit. In Hoppe, H., editor, *ACM SIGGRAPH 2011 papers*, page 1. *Cited on page 22.*

Liu, H., Yuan, Y., Deng, Y., Guo, X., Wei, Y., Lu, K., Fang, B., Di Guo, and Sun, F. (2019). Active affordance exploration for robot grasping. In Yu, H., Liu, J., Liu, L., Ju, Z., Liu, Y., and Zhou, D., editors, *Intelligent Robotics and Applications*, pages 426–438, Cham. Springer International Publishing. *Cited on page 35.*

Mansard, N. and Chaumette, F. (2007). Task sequencing for high-level sensor-based control. *IEEE Transactions on Robotics*, 23(1):60–72. *Cited on page 104.*

Mansard, N., Stasse, O., Chaumette, F., and Yokoi, K. (2007). Visually-guided grasping while walking on a humanoid robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3041–3047. *Cited on page 104.*

Marchand, E. and Chaumette, F. (1999). Active vision for complete scene reconstruction and exploration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(1):65–72. *Cited on page 30.*

Marr, D. (1982). *Vision: A computational investigation into the human representation and processing of visual information / David Marr*. Freeman, New York and Oxford. *Cited on pages 9, 13, and 19.*

McCormac, J., Handa, A., Davison, A., and Leutenegger, S. (2017). Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4628–4635. IEEE. *Cited on page 25.*

Meißner, P. (2020). *Indoor Scene Recognition by 3-D Object Search*, volume 135. Springer International Publishing, Cham. *Cited on pages 25 and 39.*

Miles, F. A. (1998). The neural processing of 3-d visual information: Evidence from eye movements. *The European journal of neuroscience*, 10(3):811–822. *Cited on pages 43 and 96.*

Milighetti, G., Vallone, L., and de Luca, A. (2011). Adaptive predictive gaze control of a redundant humanoid robot head. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3192–3198. IEEE. *Cited on page 45.*

Mishra, A. K., Aloimonos, Y., Cheong, L.-F., and Kassim, A. A. (2012). Active visual segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):639–653. *Cited on page 14.*

Mojtahedzadeh, R., Bouguerra, A., Schaffernicht, E., and Lilienthal, A. J. (2015). Support relation analysis and decision making for safe robotic manipulation tasks. *Robotics and Autonomous Systems*, 71:99–117. *Cited on pages 26, 27, 29, 55, and 129.*

Monica, R. and Aleotti, J. (2018a). Contour-based next-best view planning from point cloud segmentation of unknown objects. *Autonomous Robots*, 42(2):443–458. *Cited on pages 35, 39, and 41.*

Monica, R. and Aleotti, J. (2018b). Surfel-based next best view planning. *IEEE Robotics and Automation Letters*, 3(4):3324–3331. *Cited on page 38.*

Monica, R., Aleotti, J., and Caselli, S. (2016). A kinfu based approach for robot spatial attention and view planning. *Robotics and Autonomous Systems*, 75:627–640. *Cited on pages 35, 38, 41, 67, 68, 70, and 111.*

Monica, R., Aleotti, J., and Piccinini, D. (2019). Humanoid robot next best view planning under occlusions using body movement primitives. In *IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pages 2493–2500. IEEE. *Cited on pages 35, 37, and 42.*

Newcombe, R. A., Davison, A. J., Izadi, S., Kohli, P., Hilliges, O., Shotton, J., Molyneaux, D., Hodges, S., Kim, D., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136. IEEE. *Cited on page 20.*

Nguyen, T. V., Zhao, Q., and Yan, S. (2018). Attentive systems: A survey. *International Journal of Computer Vision*, 126(1):86–110. *Cited on page 40.*

Omrčen, D. and Ude, A. (2010). Redundant control of a humanoid robot head with foveated vision for object tracking. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4151–4156. *Cited on pages 45 and 96.*

Oßwald, S. and Bennewitz, M. (2018). Gpu-accelerated next-best-view coverage of articulated scenes. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 603–610. IEEE. *Cited on pages 35, 37, 38, 42, and 77.*

Oßwald, S., Karkowski, P., and Bennewitz, M. (2017). Efficient coverage of 3d environments with humanoid robots using inverse reachability maps. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 151–157. IEEE. *Cited on pages 36, 38, 42, and 67.*

Panda, S., Hafez, A. H. A., and Jawahar, C. V. (2013). Learning support order for manipulation in clutter. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 809–815. *Cited on page 26.*

Peters, R. A., Hambuchen, K. A., and Bodenheimer, R. E. (2009). The sensory ego-sphere: A mediating interface between sensors and cognition. *Autonomous Robots*, 26(1):1–19. *Cited on page 93.*

Pfister, H., Zwicker, M., van Baar, J., and Gross, M. (2000). Surfels. In Brown, J. R. and Akeley, K., editors, *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, pages 335–342, New York, New York, USA. ACM Press. *Cited on page 20.*

Pham, T. T., Eich, M., Reid, I., and Wyeth, G. (2016). Geometrically consistent plane extraction for dense indoor 3d maps segmentation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4199–4204. IEEE. *Cited on page 24.*

Pito, R. (1999). A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030. *Cited on page 32.*

Potapova, E., Zillich, M., and Vincze, M. (2017). Survey of recent advances in 3d visual attention for robotics. *The International Journal of Robotics Research*, 36(11):1159–1176. *Cited on page 40.*

Potthast, C. and Sukhatme, G. S. (2014). A probabilistic framework for next best view estimation in a cluttered environment. *Journal of Visual Communication and Image Representation*, 25(1):148–164. *Cited on pages 35, 36, and 42.*

Radford, N. A., Strawser, P., Hambuchen, K., Mehling, J. S., Verdeyen, W. K., Donnan, A. S., Holley, J., Sanchez, J., Nguyen, V., Bridgwater, L., Berka, R., Ambrose, R., Myles Markee, M., Fraser-Chanpong, N. J., McQuin, C., Yamokoski, J. D., Hart, S., Guo, R., Parsons, A., Wightman, B., Dinh, P., Ames, B., Blakely, C., Edmondson, C., Sommers, B., Rea, R., Tobler, C., Bibby, H., Howard, B., Niu, L., Lee, A., Conover, M., Truong, L., Reed, R., Chesney, D., Platt, R., Johnson, G., Fok, C.-L., Paine, N., Sentis, L., Cousineau, E., Sinnet, R., Lack, J., Powell, M., Morris, B., Ames, A., and Akinyode, J. (2015). Valkyrie: Nasa's first bipedal humanoid robot. *Journal of Field Robotics*, 32(3):397–419. *Cited on page 1.*

Rasolzadeh, B., Bjorkman, M., Huebner, K., and Kragic, D. (2010). An active vision system for detecting, fixating and manipulating objects in the real world. *The International Journal of Robotics Research*, 29(2-3):133–154. *Cited on page 30.*

Richtsfeld, A., Morwald, T., Prankl, J., Zillich, M., and Vincze, M. (2012). Segmentation of unknown objects in indoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4791–4796. IEEE. *Cited on page 24.*

Roberts, L. G. (1963). *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology. *Cited on page 22.*

Roncone, A., Pattacini, U., Metta, G., and Natale, L. (2014). Gaze stabilization for humanoid robots: A comprehensive framework. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 259–264. *Cited on pages 44 and 46.*

Roncone, A., Pattacini, U., Metta, G., and Natale, L. (2016). A cartesian 6-dof gaze controller for humanoid robots. In *Proceedings of Robotics: Science and Systems*. *Cited on pages 44, 45, and 46.*

Rosman, B. and Ramamoorthy, S. (2011). Learning spatial relationships between objects. *The International Journal of Robotics Research*, 30(11):1328–1342. *Cited on page 25.*

Rothfuss, J., Ferreira, F., Aksoy, E. E., Zhou, Y., and Asfour, T. (2018). Deep episodic memory: Encoding, recalling, and predicting episodic experiences for robot action execution. *IEEE Robotics and Automation Letters*, 3(4):4007–4014. *Cited on page 14.*

Ruesch, J., Lopes, M., Bernardino, A., Hornstein, J., Santos-Victor, J., and Pfeifer, R. (2008). Multimodal saliency-based bottom-up attention a framework for the humanoid robot icub. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 962–967. *Cited on page 93.*

Rusu, R. B. and Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4. *Cited on page 51.*

Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., and Beetz, M. (2008). Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941. *Cited on pages 23 and 29.*

Schauerte, B. (2016). *Multimodal computational attention for scene understanding and robotics*, volume 30 of *Cognitive systems monographs*. Springer, Switzerland. *Cited on pages 40 and 111.*

Schiebener, D. (2017). *Integrating Vision and Physical Interaction for Discovery, Segmentation and Grasping of Unknown Objects*. PhD thesis, Karlsruher Institut für Technologie (KIT). *Cited on page 16.*

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226. *Cited on pages 22 and 51.*

Schnabel, R., Wessel, R., Wahl, R., and Klein, R. (2008). Shape recognition in 3d point-clouds. In Skala, V., editor, *The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2008*. UNION Agency-Science Press. *Cited on pages 22, 25, 29, and 53.*

Schweigart, G., Mergner, T., Evdokimidis, I., Morand, S., and Becker, W. (1997). Gaze stabilization by optokinetic reflex (okr) and vestibulo-ocular reflex (vor) during active head rotation in man. *Vision Research*, 37(12):1643–1652. *Cited on page 43.*

Scott, W. R., Roth, G., and Rivest, J.-F. (2003). View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1):64–96. *Cited on page 32.*

Shibata, T. and Schaal, S. (2001). Biomimetic gaze stabilization based on feedback-error-learning with nonparametric regression networks. *Neural Networks*, 14(2):201–216. *Cited on pages 43 and 46.*

Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *Computer Vision – ECCV 2012*, volume 7576 of *Lecture Notes in Computer Science*, pages 746–760. Springer Berlin Heidelberg, Berlin, Heidelberg. *Cited on pages 26, 29, and 129.*

Sippel, D. (2016). Verfahren zur bildstabilisierung für humanoide roboter. Bachelor's thesis, Karlsruhe Institute of Technology (KIT). *Cited on pages 92 and 96.*

Sippel, D. (2019). Aktive visuelle wahrnehmung für die extraktion von support relationen. Master's thesis, Karlsruhe Institute of Technology (KIT). *Cited on page 67.*

Spenko, M., Buerger, S., and Iagnemma, K. (2018). *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*, volume 121. Springer International Publishing, Cham. *Cited on page 1.*

Stasse, O., Foissotte, T., Larlus, D., Kheddar, A., and Yokoi, K. (2008). Treasure hunting for humanoids robot. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, W-"Workshop on Cognitive Humanoid Vision", page 9, Daejeon, South Korea. *Cited on page 33.*

Stein, S. C., Worgotter, F., Schoeler, M., Papon, J., and Kulvicius, T. (2014). Convexity based object partitioning for robot applications. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3213–3220. *Cited on page 61.*

Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J., Upcroft, B., Abbeel, P., Burgard, W., Milford, M., and Corke, P. (2017). The limits and potentials of deep learning for robotics. *The International Journal of Robotics Research*, 37(4-5):405–420. *Cited on page 31.*

Suppa, M., Wang, P., Gupta, K., and Hirzinger, G. (2004). C-space exploration using noisy sensor models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4777–4782. IEEE. *Cited on page 34.*

Tarabanis, K. A., Tsai, R. Y., and Allen, P. K. (1995). The mvp sensor planning system for robotic vision tasks. *IEEE Transactions on Robotics and Automation*, 11(1):72–85. *Cited on page 33.*

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic robotics*. Intelligent robotics and autonomous agents. MIT Press, Cambridge, Mass. *Cited on pages 21 and 74.*

Torabi, L. and Gupta, K. (2011). An autonomous six-dof eye-in-hand system for in situ 3d object modeling. *The International Journal of Robotics Research*, 31(1):82–100. *Cited on pages 32, 33, 34, and 37.*

Treisman, A. M. and Gelade, G. (1980). A feature-integration theory of attention. *Cognitive Psychology*, 12(1):97–136. *Cited on page 39.*

Tsagarakis, N. G., Caldwell, D. G., Negrello, F., Choi, W., Baccelliere, L., Loc, V. G., Noorden, J., Muratore, L., Margan, A., Cardellino, A., Natale, L., Mingo Hoffman, E., Dallali, H., Kashiri, N., Malzahn, J., Lee, J., Kryczka, P., Kanoulas, D., Garabini, M., Catalano, M., Ferrati, M., Varricchio, V., Pallottino, L., Pavan, C., Bicchi, A., Settimi, A., Rocchi, A., and Ajoudani, A. (2017). WALK-MAN: A high-performance humanoid platform for realistic environments. *Journal of Field Robotics*, 34(7):1225–1259. *Cited on page 1.*

Tsikos, C. J. and Bajcsy, R. K. (1991). Segmentation via manipulation. *IEEE Transactions on Robotics and Automation*, 7(3):306–319. *Cited on page 16.*

Ude, A. and Asfour, T. (2008). Control and recognition on a humanoid head with cameras having different field of view. In *2008 19th International Conference on Pattern Recognition (ICPR)*, pages 1–4. *Cited on page 45.*

Ude, A., Atkeson, C. G., and Cheng, G. (2003). Combining peripheral and foveal humanoid vision to detect, pursue, recognize and act. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2173–2178. *Cited on page 45.*

Vahrenkamp, N., Kröhnert, M., Ulbrich, S., Asfour, T., Metta, G., Dillmann, R., and Sandini, G. (2013). Simox: A robotics toolbox for simulation, motion and grasp planning. In Lee, S., Cho, H., Yoon, K.-J., and Lee, J., editors, *Intelligent Autonomous Systems 12*, volume 193 of *Advances in Intelligent Systems and Computing*, pages 585–594. Springer Berlin Heidelberg, Berlin, Heidelberg. *Cited on pages 75 and 93.*

Vahrenkamp, N., Wächter, M., Kröhnert, M., Welke, K., and Asfour, T. (2015). The robot software framework armarx. *it - Information Technology*, 57(2). *Cited on page 10.*

Vahrenkamp, N., Wieland, S., Azad, P., Gonzalez, D., Asfour, T., and Dillmann, R. (2008). Visual servoing for humanoid grasping and manipulation tasks. In *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, pages 406–412. *Cited on page 106.*

Vannucci, L., Tolu, S., Falotico, E., Dario, P., Lund, H. H., and Laschi, C. (2016). Adaptive gaze stabilization through cerebellar internal models in a humanoid robot. In *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 25–30. IEEE. *Cited on pages 44 and 46.*

Vasquez-Gomez, J. I., Sucar, L. E., and Murrieta-Cid, R. (2017). View/state planning for three-dimensional object reconstruction under uncertainty. *Autonomous Robots*, 41(1):89–109. *Cited on pages 33, 35, 67, 76, and 129.*

Vasquez-Gomez, J. I., Sucar, L. E., Murrieta-Cid, R., and Lopez-Damian, E. (2014). Volumetric next-best-view planning for 3d object reconstruction with positioning error. *International Journal of Advanced Robotic Systems*, 11(10):159. *Cited on pages 3, 36, 38, 74, and 76.*

Vijayakumar, S., Conradt, J., Shibata, T., and Schaal, S. (2001). Overt visual attention for a humanoid robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2332–2337. *Cited on pages 40 and 43.*

von Holst, E. (1954). Relations between the central nervous system and the peripheral organs. *The British Journal of Animal Behaviour*, 2(3):89–94. *Cited on page 44.*

von Holst, E. and Mittelstaedt, H. (1950). Das reafferenzprinzip. *Naturwissenschaften*, 37(20):464–476. *Cited on page 44.*

Wächter, M., Ovchinnikova, E., Wittenbeck, V., Kaiser, P., Szedmak, S., Mustafa, W., Kraft, D., Krüger, N., Piater, J., and Asfour, T. (2018). Integrating multi-purpose natural language understanding, robot's memory, and symbolic planning for task execution in humanoid robots. *Robotics and Autonomous Systems*, 99:148–165. *Cited on pages 100 and 120.*

Wagner, R., Frese, U., and Bauml, B. (2013). Real-time dense multi-scale workspace modeling on a humanoid robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5164–5171. IEEE. *Cited on page 20.*

Wahrmann, D., Hildebrandt, A.-C., Bates, T., Wittmann, R., Sygulla, F., Seiwald, P., and Rixen, D. (2019). Vision-based 3d modeling of unknown dynamic environments for real-time humanoid navigation. *International Journal of Humanoid Robotics*, 16(01):1950002. *Cited on page 24.*

Walther, D., Rutishauser, U., Koch, C., and Perona, P. (2005). Selective visual attention enables learning and recognition of multiple objects in cluttered scenes. *Computer Vision and Image Understanding*, 100(1-2):41–63. *Cited on page 40.*

Welke, K. (2011). *Memory-Based Active Visual Search for Humanoid Robots*. PhD thesis. *Cited on pages 14 and 40.*

Welke, K., Schiebener, D., Asfour, T., and Dillmann, R. (2013). Gaze selection during manipulation tasks. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 652–659. *Cited on pages 30, 92, and 93.*

Westfechtel, T., Ohno, K., Mertsching, B., Nickchen, D., Kojima, S., and Tadokoro, S. (2016). 3d graph based stairway detection and localization for mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 473–479. IEEE. *Cited on page 24.*

Whelan, T., Salas-Moreno, R. F., Glocker, B., Davison, A. J., and Leutenegger, S. (2017). Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716. *Cited on pages 21, 61, and 117.*

Xu, K., Shi, Y., Zheng, L., Zhang, J., Liu, M., Huang, H., Su, H., Cohen-Or, D., and Chen, B. (2016). 3d attention-driven depth acquisition for object identification. *ACM Transactions on Graphics*, 35(6):1–14. *Cited on pages 35, 41, and 42.*

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, pages 146–151. *Cited on page 33.*

Yarbus, A. L. (1967). *Eye Movements and Vision*. Springer US, Boston, MA. *Cited on page 12.*

Yu, Y. and Gupta, K. (2004). C-space entropy: A measure for view planning and exploration for general robot-sensor systems in unknown environments. *The International Journal of Robotics Research*, 23(12):1197–1223. *Cited on page 33.*

Zampogiannis, K., Yang, Y., Fermuller, C., and Aloimonos, Y. (2015). Learning the spatial semantics of manipulation actions through preposition grounding. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1389–1396. IEEE. *Cited on page 25.*

Zheng, L., Zhu, C., Zhang, J., Zhao, H., Huang, H., Niessner, M., and Xu, K. (2019). Active scene understanding via online semantic reconstruction. *Computer Graphics Forum*, 38(7):103–114. *Cited on page 25.*

Zollhöfer, M., Stotko, P., Görlitz, A., Theobalt, C., Nießner, M., Klein, R., and Kolb, A. (2018). State of the art on 3d reconstruction with rgb-d cameras. *Computer Graphics Forum*, 37(2):625–652. *Cited on page 20.*