Robust Machine Vision for Service Robotics

From the Faculty of Physics and Electrical Engineering
University of Bremen

for obtaining the academical degree of

Doktor-Ingenieur (Dr.-Ing)

approved dissertation

by

Dipl.-Ing. Sorin Mihai Grigorescu

Primary Supervisor:	Prof. Dr.-Ing. Axel Gräser
Secondary Supervisor:	Prof. Dr.-Ing. Walter Anheier
Submitted at:	01. October 2009
Date of Dissertation Colloquium:	20. January 2010

To my mother
(Pentru mama mea)
Marmandiu Maria

Acknowledgements

I would like to thank Prof. Dr. Axel Gräser for his support and encouragement during my stay at The Institute of Automation (IAT) in Bremen, as well as for many helpful discussions. I would also like to thank my second reviewer, Prof. Dr. Walter Anheier and also to Prof. Dr. Udo Frese and Prof. Dr. Karl-Ludwig Krieger for showing interest to be on my dissertation committee.

A very special thank you goes to my friend Dr. Danijela Ristić-Durrant for her guidance and help during the years spent on my work. Without her the results presented in this thesis would probably never taken shape. Also, I wish to send my regards to Prof. Dr. Florin Moldoveanu who managed to give me during my years of study a part of his passion for engineering. I would like to show my gratitude to my friend Ing. Maria Johansson for her patience in correcting this thesis. My kind regards go to my colleagues, especially to Ing. Saravana Natarajan for helping me with evaluating the 3D object reconstruction results. I am also deeply thankful to my former students, Ing. Dennis Mronga and Ing. Björn Steinsträter, who invested a lot of their time in the development of the algorithms presented here. I thank also my IAT colleagues with whom I have worked with and wish them a good luck in their research activities and doctoral dissertations: Ing. Christos Fragkopoulos, Inf. Torsten Heyer, Ing. Uwe Lange, Ing. Adrian Leu, Ing. Leila Fotoohi, Ing. Tyaniu Wang and Ing. Thorsten Lüth. Many thanks to my former colleagues and friends, Dr. Oliver Prenzel, Dr. Dorin Aiteanu, Ing. Alin Brindusescu, Inf. Sorin Ivascu, Ing. Sai Krishna Vuppala and Inf. Chao Wang.

My deepest gratitude goes to my mother Maria Marmandiu and to my girlfriend Psi. Roxana Comnac. Thank you for supporting me all those years spent away from home and from you. Also, a big thank you goes to my Bremen family composed of Dr. Ioana Daraban and Chem. Daniel Daraban. I thank all my friends for the good moments that we had in the past years: Dr. Sandip Dhomse, Dr. Martyn Durrant, Psi. Daniela and Daniel Cristea, Psi. Costi Ciocoiu and Veronica Urjan.

Last but not least, I thank my mountain hiking friend that shaped a part of my character, Econ. Marian Huiban, who, during my research work, gave his life to the mountain.

Bremen, April 2010 Sorin M. Grigorescu

Abstract

In this thesis the vision architecture ROVIS (*RObust machine VIsion for Service robotics*) is suggested. The purpose of the architecture is to improve the robustness and accuracy of visual perceptual capabilities of service robotic systems. In comparison to traditional industrial robot vision where the working environment is predefined, service robots have to cope with variable illumination conditions and cluttered scenes. The key concept for robustness in this thesis is the inclusion of feedback structures within the image processing operations and between the components of ROVIS. Using this approach a consistent processing of visual data is achieved.

Specific for the suggested vision system are the novel methods used in two important areas of ROVIS: definition of an image ROI, on which further image processing algorithms are to be applied, and robust object recognition for reliable 3D object reconstruction. The ROI definition process, build around the well known "bottom-up top-down" framework, uses either pixel level information to construct a ROI bounding the object to be manipulated, or contextual knowledge from the working scene for bounding certain areas in the imaged environment. The object recognition and 3D reconstruction chain is developed for two cases: region and boundary based segmented objects. Since vision in ROVIS relies on image segmentation on each processing stage, that is image ROI definition and object recognition, robust segmentation methods had to be developed. As said before, the robustness of the proposed algorithms, and consequently of ROVIS, is represented by the inclusion of feedback mechanisms at image processing levels. The validation of the ROVIS system is performed through its integration in the overall control architecture of the service robot FRIEND. The performance of the proposed closed-loop vision methods is evaluated against their open-loop counterparts.

Kurzfassung

In der vorliegenden Dissertation wird das Bildverarbeitungsrahmenwerk ROVIS (*RObust machine VIsion for Service robotics*) vorgestellt. Dieses Rahmenwerk dient zur Verbesserung von Robustheit und Genauigkeit der visuell wahrnehmenden Fähigkeiten von Servicerobotiksystemen. Im Vergleich zu traditionellen Industrierobotern, bei denen die Arbeitsumgebung vordefiniert ist, müssen Serviceroboter variierende Beleuchtungsbedingungen und komplexe Umgebungen meistern. Das Schlüsselkonzept für die Robustheit in dieser Dissertation ist der Einsatz von Rückkopplungsstrukturen in den Bildverarbeitungsalgorithmen und zwischen den einzelnen ROVIS-Komponenten. Unter Verwendung dieses Ansatzes wird eine konsistente Verarbeitung der visuellen Daten erreicht.

Charakteristisch für das vorgeschlagene Bildverarbeitungssystem sind die neuartigen Methoden, die in zwei wichtigen Bereichen von ROVIS genutzt werden: die Definition von ROIs (Region Of Interest) im Bild, auf die dann weitere Bildverarbeitungsalgorithmen angewandt werden können, und die robuste Objekterkennung für zuverlässige 3D-Rekonstruktion. Das Verfahren zur Definition der ROI, das um das allgemein bekannte "bottom-up top-down" Rahmenwerk errichtet wurde, verwendet entweder Pixelinformationen zur Konstruktion einer ROI, die das interessierende Objekt enthält, oder kontextabhängige Erkenntnisse aus der Szene für die Begrenzung bestimmter Bereiche der visualisierten Umgebung. Die Objekterkennung und 3D-Rekonstruktion wurde für zwei Fälle entwickelt: bereichs- und kantenbasierte Erkennung von Objekten. Weil die Bildverarbeitung in ROVIS in jeder Verarbeitungsphase, d.h. bei der ROI-Definition und der Objekterkennung, auf Bildsegmentierung beruht, mussten robuste Segmentierungsalgorithmen entwickelt werden. Wie bereits erwähnt, wird die Robustheit der vorgestellten Verfahren und damit die Robustheit von ROVIS durch den Einsatz von Rückkopplungsstrukturen auf der Ebene der Bildverarbeitung erreicht. Eine Bestätigung der Güte von ROVIS ist durch die Integration im Steuerungsrahmenwerk des Serviceroboters FRIEND gegeben. Die Effizienz der vorgestellten visuellen Rückkopplungsmethoden wird durch einen Vergleich mit den zugehörigen Verfahren, die offene Regelkreise verwenden, bewertet.

Contents

1. Introduction

In humans, biological vision represents the transformation of visual sensation into visual perception [98]. The analogous computerized operation, also known as *computer vision*, deals with interpretation of digital images for the purpose of visual understanding of a scene by a machine, that is a computer. In comparison to computer vision, *machine vision* deals with the combination of different computer vision techniques and dedicated hardware for solving different tasks in fields like industrial manufacture, safety systems, control of Automated Guided Vehicles (AGVs), monitoring of agricultural production, medical image processing, artificial visual sensing for the blind or vision for robotic systems, also referred to as *robot vision*. The increased research in robot vision in the past years has spawned a large amount of systems and applications. From available literature, robot vision systems can be grouped according to the type of application they were designed for:

- *vision for manipulation* which represents the class of robot vision applications designed to detect objects that can be grasped by a dexterous manipulator;
- *vision in mobile robotics* characterized by the vision systems used for autonomous robot navigation and path following;
- *vision for mobile manipulation* representing a hybrid vision system designed for both robot navigation and dexterous manipulation.

Depending on the robot vision application, the visualized scene can be found either in an industrial environment, where position and orientation of objects is predefined and the illumination controlled, or, as the case of service robots, the imaged scene consists of typical human surroundings where objects are occluded and visualized in variable illumination conditions.

In this thesis, the problem of designing, improving and implementing service robotic vision systems is approached. Service robots represent a class of robots designed to operate semi- or fully autonomously to perform tasks useful to the well-being of humans and equipment, excluding manufacturing operations [114]. The applications of service robots range from simple domestic systems (e.g. vacuum [110] or automatic pool cleanears [112]) to entertainment [111] and social robots [115]. A special case of service robots which received large attention in last years are assistive systems designed to help disabled and elderly people. Such a robotic platform is FRIEND (*Functional Robot with dexterous arm and user-frIENdly interface for Disabled people*), a semi-autonomous service robot in its 3rd generation designed to support disabled people with impairments of their upper limbs in Activities of Daily Living (ADL) and professional life. The system consists of a seven

Degrees-of-Freedom (7-DoF) manipulator mounted on an electrical wheelchair. FRIEND is equipped with various sensors that provide intelligent perception of the environment needed for task execution support. One of these sensors is a stereo camera system which provides visual information regarding the system's environment. In particular, this thesis concerns the improvement of visual perceptual capabilities of the robotic system FRIEND for visual guided object grasping, a field of robotics in which a computer controls a manipulator's motion under visual guidance, much like people do in everyday life when reaching for objects. A key requirement in this field is the reliable recognition of objects in the robot's camera image, extraction of object features from the images and, based on the extracted features, subsequent correct object localization in a complex 3D (three dimensional) environment.

The main problem with service robotic systems such as FRIEND is that they have to operate in dynamic surroundings where the state of the environment is unpredictable and changes stochastically. Hence, two main problems have been encountered when developing image processing systems for service robotics: *unstructured environment* and *variable illumination conditions*. Such a scene can be easily noticed in everyday life: when a human is searching for an object he/she looks for it through a multitude of different other objects. Although this process is relatively simple for humans, its implementation on a machine has a high degree of complexity since a large amount of visual information is involved. A second major problem in robot vision is the wide spectrum of illumination conditions that appear during the on-line operation of the machine vision system. In a large number of vision applications one important attribute used in object recognition is the color property of objects. In case of the human visual system color is a result of the processing done by the brain and the retina which are able to determine the color of an object irrespective to the illuminant. The ability of the human visual system to compute color descriptors that stay constant even in variable illumination conditions is referred to as *color constancy* [26]. Although the color constancy ability is taken for granted in the human visual system this is not the case of machine vision.

Reliable object recognition and 3D reconstruction in robot vision is approached in this thesis through image segmentation, which represents the partitioning of a digital image into subregions suitable for further analysis. In literature, a number of object recognition methods that calculate an object's *Position and Orientation* (POSE) without the use of image segmentation have been proposed. One important category of such methods are based on the *Scale Invariant Feature Transform* (SIFT) [57]. Since then, a large number of SIFT based methods has spawned applications in various fields of computer vision, ranging from robotics [91] to medical image processing [27]. SIFT is a transformation for image features generation which are invariant to image translation, scaling, rotation and partially invariant to illumination changes and affine projection. The method can be used to generate a set of key features of an object which can further be used to relocate the object in an image. The key features, also called keypoints, are defined as maxima/minima of *Difference of Gaussians* (DoG) that occur at multiple scales from an image convolved with Gaussian filters at different scales [57]. Although the method is relatively robust with

respect to occlusions and illumination conditions its major drawbacks are that firstly it needs an a priori model of the object to be recognized and secondly that the object has to be textured in order to calculate as many key features as possible. Also, the precision of the method is low if the object does not have a planar surface. These drawbacks motivate again the usage of image segmentation in robot vision where the precision of 3D object POSE estimation is crucial.

In Figure 1.1 the dependencies of object recognition and 3D reconstruction with respect to image segmentation are shown. The arrows in the figure represent the possible flow of information that may exist in a robot vision system where segmentation plays a central part. Image features, which provide object position in the 2D (two dimensional) image plane, extracted from binary segmented images, are used for recognizing object types and also to reconstruct their 3D shape in a virtual Cartesian space. Also, as it will be explained in Chapter 6.1.2, information regarding recognized objects can be used to improve segmentation quality, which directly influences precision of POSE estimation, that is the precision of detected 2D object feature points used for 3D reconstruction. Feature points are defined as key object points, obtained from segmented images, from which the object's 3D shape can be build. The type of object determined via 2D recognition influences the 3D reconstruction method in the sense that different feature points are extracted for different objects (e.g. for a bottle, its top neck and bottom are used as feature points, as for a book, feature points are represented by the four corners of the book).

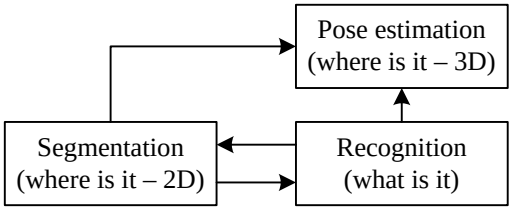


Figure 1.1.: Object recognition and 3D reconstruction in robot vision.

In this thesis, the novel robot vision system ROVIS (*RObust machine VIsion for Service robotics*) is suggested as a contribution to reliable and robust object recognition and 3D reconstruction for the purpose of manipulation motion planning and object grasping in unstructured environments with variable illumination conditions. In ROVIS, robustness must be understood as the capability of the system to adapt to varying operational conditions and is realized through the inclusion of feedback mechanisms at image processing level and also between different hardware and software components of the vision framework. A core part of the proposed system is the closed-loop calculation of an image Region of Interest (ROI) on which vision methods are to be applied. By using a ROI, the performance of object recognition and reconstruction can be improved since the scene complexity is reduced.

1.1. Related work and contribution of the thesis

According to the classification of robot vision applications made in introduction, a number of representative state-of-the art image processing architectures, related to the ROVIS system proposed in this thesis, are summarized below.

Vision for manipulation

Localization of objects for manipulative purposes has been treated in a relative large number of publications. A framework for object manipulation in domestic environments, based on visual robot control and a cognitive vision architecture, is proposed in [46, 47, 48]. The experimental platform for the framework is a Nomadic Technologies XR4000, equipped with a Puma 560 arm for object manipulation and a stereo camera set for scene recognition. The approach used in detecting objects of interest for grasping and manipulation is divided into localization of known and unknown objects in cluttered environments, using the SIFT method. The complex problem of unstructured environments is also discussed in [20] with respect to visual guided object grasping. In comparison to the mentioned work, in this thesis the problem of object recognition is treated from the point of view of improving the classical open-loop image processing chain and not from applying complex vision algorithms for object detection.

Investigations regarding the architectural aspects of robot vision for manipulative tasks can be found in [24], where a control framework for "hand-eye" manipulation, which manages complexity of tasks through the composition of a few simple primitives, is proposed. In [97], an extensive work regarding visual perception and control of robotic manipulation is given. Here, data processed by the vision system, based on 3D model shapes, is enhanced with shape recovery information acquired from robust light stripe scanning. In ROVIS, instead of using 3D primitive shape models for object reconstruction, 2D shape descriptions of objects are used for recognition and 2D feature points calculation.

In recent years, the application of vision based object manipulation has been extensively applied in the field of assistive and rehabilitation robots, as the case of the vision system proposed in this thesis. AVISO (*Assistance by VIsion for Seizure of Objects*), detailed in [52, 25], is composed of a MANUS arm and a stereo "eye-in-hand" camera system mounted on the end-effector of the arm. The AVISO vision system relies on the end-user (patient) to cope with the separation of the object to be grasped from the background and also from the other objects. This process is achieved through the user by manually moving the MANUS manipulator at a distance of approximatively ten centimeters to the object. The arm movement is controlled using a human-machine interface which displays in real-time images captured from the stereoscopic "eye-in-hand" camera. Once the object is centered on the displayed image, the user has to manually draw an image ROI bounding the object to be grasped. Finally, the object is automatically separated from the background by calculating interest points with the Harris and Stephens [33] feature detector. Using the epipolar geometry constraints, a 3D point representing the object

grasping point is calculated. Further, the manipulator arm approaches and grasps the object in a visual control manner. Although the AVISO method is reliable in unstructured environments, it implies a relatively large amount of user interaction, which can be tiring for the patient. An alternative method for AVISO, which uses less user interaction, is proposed in [25], where the user must only select a point from the object to be grasped on an image captured from the "eye-in-hand" camera. Using a second fixed camera present in the scene and epipolar geometry the system can approach and grasp the object. This method, although requiring less user interaction, it relies on the second camera present in the scene. In this thesis, user interaction is used for defining an interest point only once, hence limiting tiring interaction tasks. The defined interest point acts as a starting point for automatic adjustment of the image ROI, on which the object is automatically localized.

Similar to AVISO [52, 25], the AMOS (*Assistive MObile robot System*) vision system [96] also uses a stereoscopic "eye-in-hand" camera mounted on a manipulator arm and a *Shared Responsibility* software architecture which involves the user in the working scenarios of the robot. If the object to be grasped is out of the range of the manipulator, the position of AMOS is changed in order to get it closer to the object. In both systems presented above, AVISO and AMOS, "eye-in-hand" cameras are used. In comparison to that, in ROVIS visual information is obtained from a global, fixed, stereo camera system. The advantage of using a global camera over an "eye-in-hand" one is that it provides a global description of the imaged scene, which is more appropriate for manipulator motion planing with obstacles avoidance.

In [56] the visual control of the MANUS arm using the SIFT [59] algorithm is proposed. As said before, the disadvantage of this approach is the need of a 3D model of the object to be grasped (the SIFT-keypoints) and the fact that the SIFT algorithm provides reliable results only for planar textured objects. Recently, SIFT was used in a shared-controlled architecture to simulate ADL tasks, also considered in this thesis, using a visual controlled assistive robot arm [44].

Vision in mobile robotics

In mobile robotics, vision is commonly used for autonomous navigation control, indoor or outdoor, of a mobile robotic platform. Although the vision system proposed in this thesis has as target object recognition for manipulation, concepts from vision for mobile robotics are presented here as applied to general robot vision.

A survey of color learning algorithms and illumination invariance in mobile robotics is given in [94]. The algorithms are presented from the perspective of autonomous mobile robot navigation with stress on inter-dependencies between components and high-level action planning. As an alternative to the color segmentation methods presented in the survey, in this thesis, robustness of color segmentation against varying illumination is achieved through feedback adaptation of the image processing parameters.

The vision system proposed in [13] is one of the first ones to approach the robot vision problem from an image processing architectural point of view. The SRIPPs (*Structured Reactive Image Processing Plans*) image processing system is designed for the robot control architecture FAXBOT of the RHINO mobile robotic platform. The sequence of vision algorithms is built around an image processing pipeline. As a comparison, the architecture of ROVIS is modeled using the *Unified Modeling Language* (UML) [65]. Through the use of UML a better transparency of the system is achieved, as also a simplified development process since the structure of the computer programs that make up ROVIS is implemented graphically.

An interesting "visual attention" computer vision architecture developed for mobile robots is VOCUS (*Visual Object detection with a CompUtational attention System*) [28, 29]. The designed system is based on the cognitive capabilities and neuro-physiology of the human brain. For the recognition of objects two approaches are used:

- "bottom-up" attention, when no a priori information regarding the visualized scene exists;
- "top-down" approach, when a priori information about the objects to be recognized in the scene exists.

The "bottom-up top-down" framework is also used in this thesis for building two methods for image ROI definition. Attention vision architectures have also been studied in [75] with the purpose of optimizing the sensorimotor behavior of a mobile robot.

Vision for mobile manipulation

The systems designed for both navigation and object manipulation are mobile robotic platforms equipped with redundant manipulators. Such hybrid systems, like the mobile robot JL-2 [107] used for field operations, rely on vision to calculate the robot's moving path and also recognize objects to be manipulated.

In recent years, robotic perception in domestic environments has been treated in a large number of publications [12, 47]. The UJI librarian robot [22] was designed to detect IDs of books on a shelf. The vision system of this robot considers only the detection of books IDs, followed by their pick up from the shelf using a special designed gripper and hybrid vision/force control. In comparison to the vision system in [22], ROVIS aims at the recognition and 3D reconstruction of all types of books, placed in cluttered environments.

Care-O-Bot® represents a series of mobile robots designed to assist people in daily life activities [32, 89]. They were developed at Fraunhofer IPA [1] since 1998, currently reaching its 3rd generation. The object recognition system of Care-O-Bot uses a camera sensor and a 3D laser scanner for reconstructing the 3D representation of objects of interest. The objects are taught beforehand to the robot using model images. Also, a laser scanner is used for planning a collision free trajectory of the 7-DoF manipulator arm used in grasping

[1] Fraunhofer-Institut für Produktionstechnik und Automatisierung

and manipulating objects. In ROVIS, the goal is to develop a robust vision system based only on image data from a global stereo camera with the purpose of extracting as much visual information as possible.

In [87, 88], it has been investigated how a mobile robot can acquire an environment object model of a kitchen and more generally of human living environments. For this purpose, range sensing information acquired from a laser scanner, in form of 3D point clouds, has been used. However, such methods are strictly dependent on the range data quality provided by the sensing device. Sensed depth information can have different error values depending on the sensed surface.

A different approach in researching mobile manipulation is found in the BIRON robot (*Bielefeld Robot Companion*), where the focus is on manipulative actions for human-robot interaction [55]. The topic of human-robot interaction, also treated in [14, 15, 52, 96], plays a very important role in robotics, generally concerning safety and particularly regarding recognition and interpretation of human gestures. In this thesis, human-machine interaction is treated as a tool used for sending input commands to the robotic system.

The contributions of this thesis, with respect to the state-of-the art systems presented above, are both theoretical and practical, as summarized below.

The thesis considers the design and implementation of a robot vision system with improved visual perceptual capabilities for the purpose of manipulator path planning and object grasping [73, 72]. The focus of the proposed vision system ROVIS is specifically related to service robotic platforms, namely it treats the various form of interconnections between system components and also the connection to the overall control architecture of the robot. These conceptual elements represent a crucial factor in the well operation of a robotic system [46, 77].

The thesis represents a contribution in the field of feedback control in image processing [83, 70, 66, 49]. Because of the complex environment where ROVIS operates, its robustness with respect to external influences is critical. The role of including feedback structures at image processing level is to improve this robustness. The region based color segmentation algorithm proposed in this thesis represents further research in the field of region segmentation, as a sequel to the proved closed-loop gray level region segmentation from [83]. The case of optimal boundary segmentation has been approached with investigating a new feedback quality measure derived from feature extraction level. The objective of both segmentation methods is to reliably extract 2D object feature points needed for 3D reconstruction. As said before, the investigation of feedback mechanisms for ROVIS are intended for the improvement of the overall robustness of the vision system following the principle of decentralized control [40], where the robustness of a complex system is not achieved by a complex control mechanism, but by controlling subcomponents of it, thus ensuring overall system robustness.

One other important aspect in designing ROVIS is its performance evaluation with respect to traditional vision architectures, where image processing is performed in an

open-loop manner [108]. The algorithms presented in this thesis have been evaluated with respect to traditional ones by appropriate performance metrics used to quantify results from both proposed and compared methods. An overall evaluation of the ROVIS system was made through measuring the performance of its end result, that is 3D reconstruction of objects to be manipulated. These results have been compared with the actual positions and orientations of the objects measured in real world.

The practical aspect of the thesis is represented by the integration of the ROVIS system in the overall control architecture of the service robot FRIEND. ROVIS, sustained by the implemented vision methods, was used in building the visual perceptual capabilities of FRIEND.

1.2. Organization of the thesis

In Chapter 2, an overview of image processing hardware and operations used as building blocks for vision algorithms in ROVIS is given. Chapters 4, 5 and 6 treat the use of these operations in developing robust methods for improving the visual perceptual capabilities of the architecture.

The concept and architectural design of the service robotic vision system ROVIS is given in Chapter 3. The stress here is on core concepts of the system, that is visual data processing on an image ROI and improvement of the vision methods through feedback mechanisms implemented at different levels of image processing as also between various components of ROVIS. The integration of ROVIS within the overall control architecture of the robotic system FRIEND is also presented in this chapter.

Chapter 4 treats the development of two robust image segmentation methods required in developing the image ROI definition systems from Chapter 5 and the object recognition and 3D reconstruction chain from Chapter 6. Two types of novel segmentation operations have been discussed, that is robust region and boundary based segmentation.

The complexity reduction of a scene, through the use of an image ROI, is presented in Chapter 5. Based on the amount of available contextual information, two novel methods for image ROI definition have been proposed, namely one bottom-up approach, based on user interaction, and a second one, top-down, based on camera gaze orientation.

In Chapter 6 a robust object recognition and 3D reconstruction image processing chain is proposed. Again, two types of object recognition methods can be distinguished here, for the case of region based segmented objects and boundary segmented objects, respectively. As before, the robustness of the chain has been improved, where it was possible, with appropriate feedback mechanisms. Also, for performance evaluation, the precision of ROVIS is compared to traditional open-loop image processing approaches.

Finally, conclusions and possible further development regarding the proposed vision system are given in Chapter 7. Results from this thesis have been published in [1, 2, 3, 4, 5, 6, 7, 8, 9].

2. Object recognition and 3D reconstruction in robotics

A robot vision system is composed of different hardware and software components which, together linked, provides visual information to the robot with respect to the surrounding environment.

In this chapter, the basics of image processing used in robotics are given. In fact, only the methods used in designing the ROVIS vision algorithms are presented. Since in robotics the goal of the image processing system is to reconstruct the viewed scene in a 3D virtual environment that can be understood by the robot, the methods discussed in this chapter are explained in the context of stereo vision which deals with the process of visual perception for estimation of depth using a pair of cameras. The algorithms presented here are to be taken as basic image processing operations used in developing the robust methods, presented in Chapters 4, 5 and 6, within the ROVIS architecture.

Also, an introduction to feedback structures in image processing [83], a key concept in the development of ROVIS, is given. Feedback in image processing is presented in the context of robot vision, that is, the improvement of the visual perceptual capabilities of a robot through the inclusion of closed-loop mechanisms at image processing level. A case study of closed-loop gray-level image segmentation is presented.

2.1. Open-loop vs. closed-loop image processing

In a robotic application, industrial or real world, the purpose of the image processing system is to understand the surrounding environment of the robot through visual information. In Figure 2.1 an object recognition and 3D reconstruction chain for robot vision is presented, consisting of low and high levels of image processing operations.

Low level image processing deals with pixel wise operations with the purpose of image improvement and separation of the objects of interest from the background. Both the inputs and outputs of low level image processing blocks are images. The second type of blocks, which deal with high visual information, are connected to low level operations through the feature extraction module which converts the input image to abstract data describing the objects of interest present in the image. For the rest of high level operations both the inputs and outputs are abstract data. The importance of the quality of results coming from low level stages is related to the requirements of high level image processing [36]. Namely, in order to obtain a proper 3D virtual reconstruction of the

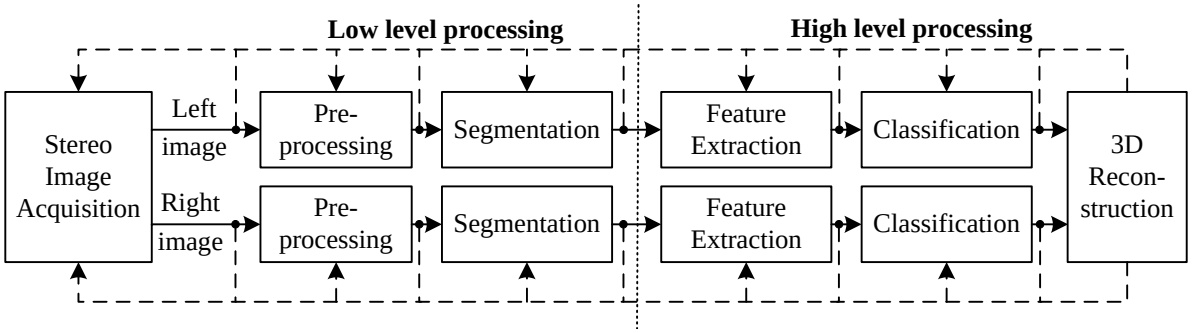


Figure 2.1.: Block diagram of conventional (solid line) and closed-loop (dashed line) object recognition and 3D reconstruction chain in robot vision.

environment at high level stage, the inputs coming from low level have to be reliable.

In Figure 2.1, the solid arrows connecting the processing modules represent a traditional, open-loop, chain in which image processing parameters have constant values. This approach has impact on the final 3D reconstruction result since each operation in the chain is applied sequentially, with no information between the different levels of processing. In other words, low level image processing is done regardless of the requirements of high level image processing. For example, if the segmentation module fails to provide a good output, all the subsequent steps will fail. Also, the usage of feedforward information for optimizing the open-loop image processing chain would fail to provide an optimal outcome since no feedback with respect to the goodness of visual processing is considered in such a structure.

In [83] the inclusion of feedback structures between image processing levels for improving the overall robustness of the chain is suggested. It is a fact that feedback has natural robustness against system uncertainty and ability to provide disturbance rejection, which is a fundamental concept in control theory [71]. The feedback between different image processing stages is represented in Figure 2.1 by dashed arrows connecting high level operations to lower ones. In this approach the parameters of low level image processing are adapted in a closed-loop manner in order to provide reliable input data to higher levels of processing. In [83] two types of feedback loops for image processing purposes have been introduced:

- *image acquisition closed-loop*, in which feedback is used for controlling the image acquisition process, thus ensuring an input image of good quality to the image processing chain; in this case the parameters of the acquisition system are controlled based on feedback information from different stages of image processing;
- *parameter adjustment closed-loop*, which deals with adaptation of image processing parameters according to the requirements of subsequent processing operations.

In this thesis, the principle of closed-loop image acquisition is used in controlling the orientation of the robot's camera system through a visual controller, as described in Chapter 5.3. Also, in order to improve robustness, parameter adjustment closed-loops are

implemented in key places of the ROVIS architecture, as will be explained in Chapters 3 to 6. The left and right images from Figure 2.1 are processed in parallel independent of each other. The possible connections between the image processing chain for the left image with the one processing the right image are not treated here. Such loops are strongly related to the geometry of the stereo camera. Hence the feedback unifying both chains should start directly from the 3D reconstruction module responsible with calculating the 3D positions of the imaged objects.

The design and implementation of feedback structures in image processing is significantly different from conventional industrial control applications, especially in the selection of the pair *actuator variable – controlled variable*. The choice of this pair has to be appropriate from the control, as well as from the image processing point of view. In [83] two types of feedback structures to be used in image processing are introduced, both presented in Figure 2.2.

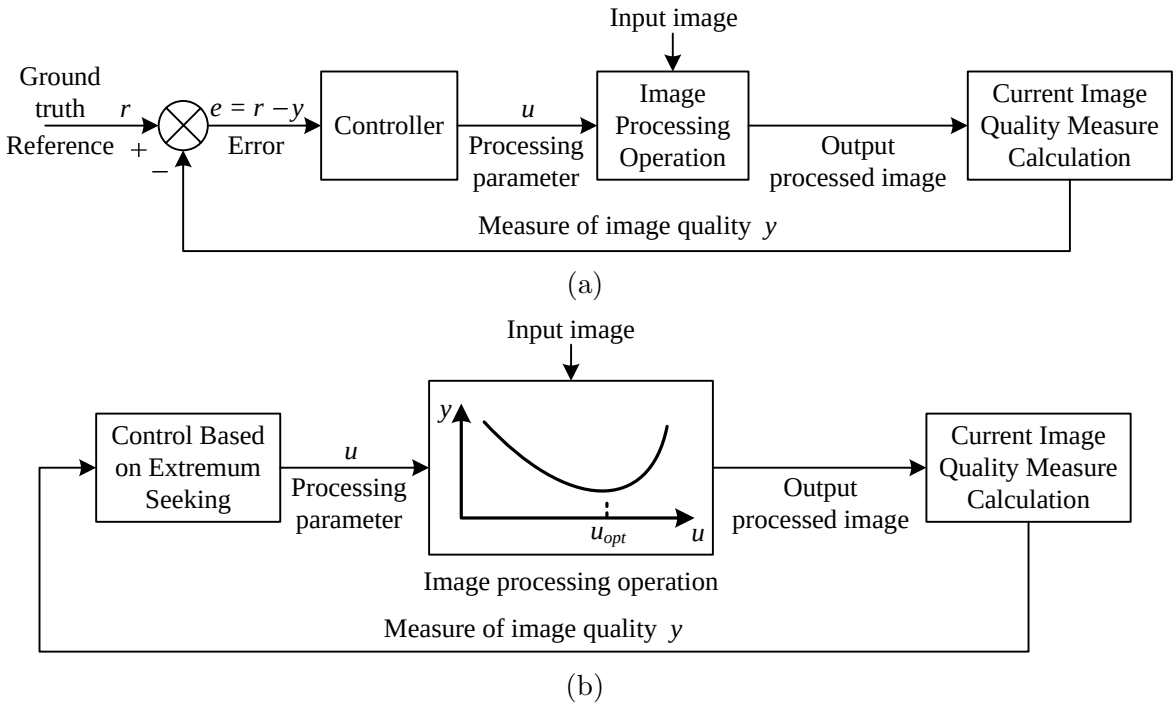


Figure 2.2.: Basic block diagrams of feedback structures used in image processing. (a) Ground truth reference image available. (b) No reference image – control based on extremum seeking.

When a reference value for the closed-loop system is available, in our case a reference image, or *ground truth*, the classical error-based control structure from Figure 2.2(a) is suggested. The ground truth is defined as what should be ideally obtained from processing an image, regardless of the processing operation. The basic principle of the error-based control structure is to automatically drive the values of the image processing parameters according to the output image quality measure, or controlled variable. Using this measure, an error between the reference value and the current output can be calculated. The

resulted error is used in determining the optimal working point of the image processing tuning parameter, or actuator variable. The actuator variable can be calculated using a classical control law, like P (Proportional) or PI (Proportional – Integral).

The second type of feedback structure proposed for image processing applications, seen in Figure 2.2(b), treats the case when no reference value for the closed-loop system exists. Here, the optimal value of the controlled variable is defined as an extreme value, maximal or minimal, of a measure of image quality. The optimal value u_{opt} of the actuator u is chosen using the calculated extreme. The calculation of the optimal output value $y = f(u_{opt})$, within the effective input operating ranges $[u_{low}, u_{high}]$, is achieved using an appropriate extremum seeking algorithm, as the *hill-climbing* method for image processing purposes described in [83], with the pseudo-code presented in Table 2.1. An introduction to extremum seeking control is given in Appendix A.

Table 2.1.: Hill-climbing procedure used in feedback structures for image processing.

1	Update the lowest input value u_{low} as the current state u_c of the search process and calculate $y_c = f(u_c)$.
2	Identify n successors of the current state u_c, $u_c + \Delta u, \ldots, u_c + n \cdot \Delta u$, and calculate $f(u_c + i \cdot \Delta u), i = 1, \ldots, n$.
3	Let $y_m = f(u_m)$ be the maximum (minimum) of $f(u_c + i \cdot \Delta u), i = 1, \ldots, n$.
4	If $y_m > y_c (y_m < y_c)$ update input value u_m as the current state u_c and go to Step 2, otherwise STOP the increasing of the input variable.
5	Current state u_c of the search process is the parameter that provides image processing result of optimal quality.

The advantages of the feedback structures mentioned above are emphasized in this thesis by improving the visual perceptual capabilities of a service robotic system through the vision architecture ROVIS.

2.2. Stereo image acquisition

The basic component in any machine vision system is the image acquisition module. Here, images are acquired from vision sensors and converted in a suitable digital representation for processing. Usually, in robot vision applications, as also in the ROVIS, stereo camera systems are used to understand the robot's surroundings. The advantages of stereo cameras over normal monocameras are discussed below. Image acquisition can be divided for explanation into two categories: *hardware components* and *image representation*, both of them presented in this section.

2.2.1. Hardware components

The starting point in developing vision based applications is the proper choice of dedicated hardware modules that, together combined, provide a reliable base for the machine

vision structure. In the following, common hardware components used in robot vision are described, namely cameras, gaze orientation systems and digital processing elements used for time critical applications.

Stereo camera systems

The most common devices used for image acquisition are the *Charged Couple Device* (CCD) and the *Complementary Metal Oxide Semiconductor* (CMOS) sensors. Their purpose is to convert optical images into electric signals. Extra circuitry is used for converting the analog voltage into digital information. Camera systems build around one image sensor are usually referred to as *monocameras*.

The problem with monocamera systems is that they only provide visual information in form of 2D images and no depth sensing, needed by robotic systems to perceive the world in a 3D form. This problem has been solved with the introduction of stereo cameras composed, as the name suggests, of a pair of image sensors. Knowing the geometrical characteristics, that is relative positions and orientations of the two sensors, geometrical relations between the 3D imaged scene and the 2D data can be derived. In such a way the 3D environment can be reconstructed in a virtual space from 2D information. In order to properly reconstruct the 3D visualized scene the position of the two image sensors relative to each other has to be precisely known.

Although the basic principle of 3D scene reconstruction relies on two image sensors, a variety of systems containing more sensors have been developed. The advantage of such a multisensor camera is the increased precision of depth calculation and a wider field of view coverage. State of the art research using multisensor cameras is 3D reconstruction from multiple images, where a scene is to be reconstructed from a number of n images acquired from n cameras simultaneous. Lately, such systems have been used in reconstructing large areas, as for example cities [113].

Recently, as an alternative to classical stereo vision, a novel type of 3D sensor, which uses frequency modulated pulses of infrared light to measure the distance from the sensor to a surface [90], has been introduced. These types of devices are commonly referred to as *Time-of-Flight* (ToF) cameras, or active cameras. Depth information is retrieved by measuring the time needed for a pulse of infrared light to travel from the image sensor to a surface and back. ToF cameras have been introduced as a faster alternative to depth information calculation. Although the depth precision is relatively high for short distances (maximum 10mm error for a distance below 1m), it decreases proportionally with the distance by a factor of 1%. At the current state of this technology, one major disadvantage of these cameras is low image resolution. This happends because the sensor has to include, along with the visual sensor, extra circuitry for senders and receivers of infrared pulses. Also, the precision of depth information is influenced by the nature of the imaged surface. For example, a black surface reflects back considerably less infrared light than a colored or white surface.

Camera gaze positioning modules

In vision based systems, especially service robotics applications, the coverage of a large field of view is needed. This requirement can be achieved by manipulating the viewpoint of the camera system, a process also known as *gaze positioning*. Since camera parameters are changed in order to facilitate the processing of visual data, this type of system is commonly referred to as *active vision* [93, 41]. Although the term active vision includes the control of several camera parameters (e.g. focus, zoom etc.), in this thesis only the closed-loop orientation of a stereo camera system is considered, as explained in Chapter 5.3.

In robotics there are typically two known approaches for controlling the orientation of a camera:

- *Eye-in-Hand* configuration, where the camera is mounted on the last joint of a general purpose manipulator arm;
- *Pan-Tilt Head* (PTH) configuration, where the Position and Orientation (POSE) of a camera is controlled by a 2-DoF unit; the pan and the tilt represent the angles controlling the orientation of the camera's viewpoint; in aviation, pan and tilt are commonly known as the yaw and pitch, respectively.

For the second approach, both image sensors can be fixed on the same PTH unit, or each sensor can be mounted separately on its own PTH. Mounting the sensors separately introduces additional degrees of freedom, like *vergence*, representing the angle between the optical axes of the two cameras, and *baseline*, representing the distance between the cameras. Vergence control is used to cooperatively move the fixation point of a stereo pair around a scene. As said before, the relation between the two image sensors has to be precisely known in order to reliably control the coordination between both PTH units.

Image processing on dedicated hardware

One very important part of a machine vision system is its computational resources, or the hardware elements on which the image processing algorithms exists. Depending on the type of vision application, the choice for the computing hardware can be made. There are basically two main technologies used for processing digital images:

- *PC based processing*, where the computing power is carried out on traditional computer architectures, usually multiprocessor PCs with a high amount of computational resources; in robotics, and particularly service robotics, PC based processing is one of the most common approach used;
- *Digital Signal Processing* (DSP), which has as focus the optimization of the speed of the image processing operations; DSP algorithms are typically implemented on specialized processors called digital signal processors, or on purpose-built hardware such as *Application-Specific Integrated Circuit* (ASIC); recently *Field-Programmable Gate Arrays* (FPGA) technology made its way into image processing as a powerful, relatively low cost, processing hardware with many promising application fields.

It is worth to mention that lately camera manufacturers have begun to include build-in processing hardware into cameras. For the design of small industrial automation applications, a basic number of image processing functions can be used directly from the camera module.

2.2.2. Image representation

Camera systems usually provide digital electrical signals representing images of the viewed scene. The representation of these signals has to be standardized and suitable for digital processing. A grey level digital image is a two-dimensional function $f(x, y)$ where the value, or amplitude, of f at spatial coordinates (x, y) is a positive scalar quantity whose physical meaning is determined by the source of the image [30]. Mathematically $f(x, y)$ is represented as an $M \times N$ matrix:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}. \tag{2.1}$$

The elements of the matrix 2.1 are called *image pixels*, or *px*. The pixels of $f(x, y)$ take values, named grey level values, in a finite interval:

$$0 \leq f(x,y) \leq K_{gray}, \tag{2.2}$$

where in typical, 8-bit, computer implementation $K_{gray} = 255$.

In case of color images their digital representation is a matrix vector containing three elements:

$$f_{RGB}(x,y) = \begin{bmatrix} f_R(x,y) & f_G(x,y) & f_B(x,y) \end{bmatrix}^T, \tag{2.3}$$

where $f_R(x, y)$, $f_G(x, y)$ and $f_B(x, y)$ represent the primary red, green and blue components of light, respectively. By adding together the three components a palette of colors can be obtained.

In case of stereo cameras the output of the image acquisition system is a pair of synchronized images:

$$\{f_L(x,y), f_R(x,y)\}, \tag{2.4}$$

where $f_L(x, y)$ and $f_R(x, y)$ represent the left and right images acquired from the stereo camera, respectively.

The relationship between the two camera sensors, mounted in a stereo correspondence manner, is described by the camera's vergence and baseline.

The image processing methods explained below are applied in parallel to both the left and the right images, as seen in Figure 2.1.

2.3. Image pre-processing

The image pre-processing stage aims at improving the acquired images or to transform them in order to be suited for the next module, image segmentation. Also, as it will be described in Chapter 5, a pre-processing operation is also the definition of an image ROI on which object recognition algorithms will be applied. Usually, in robot vision, the pre-processing operations are represented by image filtering and color space transformation.

Image filtering

Image filtering is commonly used in the removal of noise from input images and is described, for the case of spatial domain, by the linear operation:

$$g(x, y) = G[f(x, y)], \tag{2.5}$$

where $f(x, y)$ is the input image, $g(x, y)$ is the output image and G is an operator on f defined over a neighborhood of (x, y). In this thesis, a smoothing linear filter [30] was used for enhancing the quality of acquired RGB images.

Color image transformation

The representation of color images in digital computers is made according to standardized color models, also named color spaces or color systems. A color model is essentially a coordinate system where every point represents a specific color. Although in practice there is a number of existing color models to choose from, in this thesis only two of them are used, that is, the RGB (*Red*, *Green*, *Blue*) model and the HSI (*Hue*, *Saturation*, *Intensity*) model.

Color models can be classified into hardware oriented and application specific models. The most common hardware-oriented color model is the RGB model. Most types of camera systems used in robotics use the RGB color space for representing color images provided as output. The RGB model, showed in Figure 2.3(a), is based on a Cartesian coordinate system and the unit cube where colors take values in the interval $[0, 1]$. The primary color components are specified by the 3 axes of the coordinate system. The corners of the RGB cube represent the primary and secondary colors, respectively. The origin of the color cube corresponds to the black value. Between the origin and the outermost corner, which represents the white value, various shades of gray are encountered.

The transformation from the RGB color space to gray level is commonly performed as:

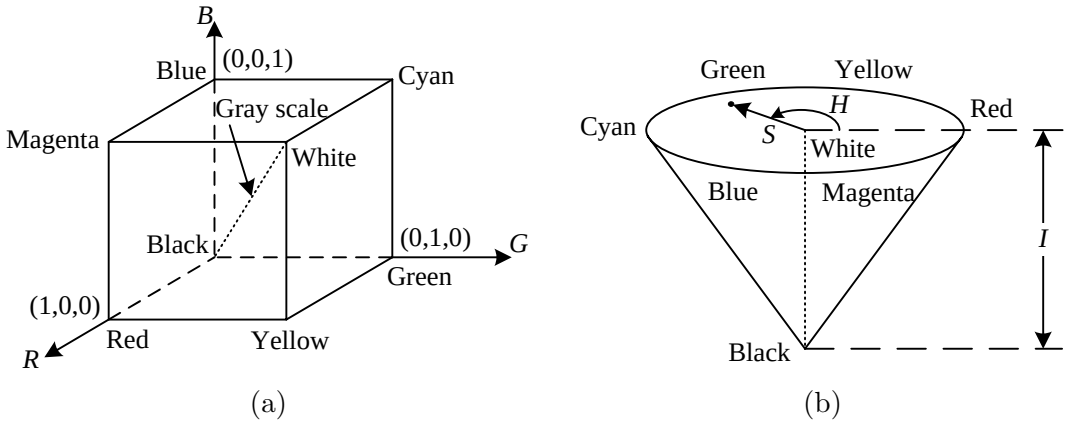


Figure 2.3.: Hardware –RGB– (a) and application –HSI– (b) oriented color models.

$$f(x,y) = c_1 \cdot f_R(x,y) + c_2 \cdot f_G(x,y) + c_3 \cdot f_B(x,y). \tag{2.6}$$

where the values of the coefficients are typically addopted as $c_1 = 0.299$, $c_2 = 0.587$ and $c_3 = 0.114$.

When developing robot vision applications the problem with the RGB model is that it does not describe color in a practical manner, that is, color information is spread in three components. A solution to this problem is the HSI model, represented in Figure 2.3(b). The advantage of this color space is that it decouples the intensity, or brightness, component from the color information, stored in the hue and saturation image planes. In the HSI model the color of a point is given by the hue component defined as the angle H of the color circle in Figure 2.3(b). The hue H takes values in the interval $[0, 2\pi]$, where 0 corresponds to the red color.

The saturation S represents the purity of a color, or the amount of white added to a pure color. S is defined as the radius of the color circle in Figure 2.3(b). Finally, the intensity I specifies the brightness of a point and is defined as the vertical axis of the color cone in Figure 2.3(b). The transformation from the RGB to the HSI color model is governed by a set of three equations:

$$H = \begin{cases} H \text{ if } B \leq G, \\ 360 - H \text{ if } B > G, \end{cases} \tag{2.7}$$

where

$$H = cos^{-1} \left\{ \frac{1/2 \cdot [(R-G) + (R-B)]}{\sqrt{[(R-G)^2 + (R-B)(G-B))]}} \right\}, \tag{2.8}$$

$$S = 1 - \frac{3}{(R+G+B)}[min(R,G,B)], \tag{2.9}$$

$$I = \frac{1}{3}(R + G + B). \tag{2.10}$$

In a computer, an HSI image is represented similar to its RGB counterpart, as:

$$f_{HSI}(x, y) = \begin{bmatrix} f_H(x, y) & f_S(x, y) & f_I(x, y) \end{bmatrix}^T, \tag{2.11}$$

where $f_H(x, y)$, $f_S(x, y)$ and $f_I(x, y)$ represent the hue, saturation and intensity components, respectively.

2.4. Image segmentation

Image segmentation is often one of the most difficult stages in the image processing chain from Figure 2.1. It refers to the process of partitioning a digital image into subregions, or sets of pixels, suitable for further computer analysis. The goal of image segmentation is to separate objects of interest from the background by assigning a label to every pixel in the image such that pixels with the same label share a specific visual characteristic.

According to the output result, segmentation is classified in two distinct approaches: region based and boundary based segmentation. Both types will be explained here as an application to a gray level image followed by the color image segmentation approach. In this thesis the output of the segmentation step is regarded as a binary image containing foreground, or objects (black pixels 1s), and background (white pixels 0s). The process of image segmentation is also encountered with the name *binarization*.

Following, two image segmentation approaches will be discussed: traditional, open-loop, segmentation and a novel, closed-loop, segmentation method introduced in [83].

2.4.1. Open-loop image segmentation

Region based segmentation

Region based image segmentation techniques are aimed at grouping pixels according to common image properties like intensity values, texture or spectral profiles that provide multidimensional image data.

The most common way to perform region based segmentation is histogram thresholding. If the image pixels values from a histogram can be separated by a *global threshold* T_G, then the background pixels in the output binary image are represented by the pixels in the input image with a value lower than T_G and, respectively, the foreground pixels by the ones with a value higher or equal to T_G, as:

$$t_G(x, y) = \begin{cases} 1, & \text{if } f(x, y) \geq T_G, \\ 0, & \text{if } f(x, y) < T_G, \end{cases} \tag{2.12}$$

where $t_G(x, y)$ is the output binary image.

A requirement when segmenting images with only one threshold value is that the input image has to contain only one object and a uniform background. One way around this problem is to define a threshold interval $T = [T_{low}, T_{high}]$ as:

$$t(x, y) = \begin{cases} 1, & \text{if } f(x, y) \in T, \\ 0, & \text{if } f(x, y) \notin T, \end{cases} \tag{2.13}$$

where $f(x, y)$ is the pixel value at image coordinates (x, y). T_{low} and T_{high} are the low and high thresholding boundaries applied to the histogram of image $f(x, y)$.

An automatic approach for histogram thresholding is the so-called *adaptive threshold* which thresholds the image based on a moving window, or mask. The optimal threshold value T_{opt} is calculated based on the mean pixels value of the mask. A more complicated automatic thresholding technique is the so-called Otsu method which makes use of the inter class variance between the object pixels and the background [74].

Boundary based segmentation

The purpose of boundary based segmentation is to extract the edges of the objects in an image. This is commonly done by detecting sharp local changes between the intensity of the pixels in an image. The output of this operations, also called *edge detection*, is a binary image containing as foreground pixels the edges in the image, or the places where intensity changes abruptly.

The basic principle of edge detection is to locally calculate the image gradient through partial derivatives of order one or two. The gradient of an image $f(x, y)$ at location (x, y) is defined as the vector:

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix}. \tag{2.14}$$

The calculation of the gradient is made using a mask shifted on the input image. The resulted gradient image is then thresholded using relation 2.12. The problem with using such an edge detector lies in the difficulty of choosing the appropriate threshold value. If the threshold value is set too low, then the binary output image will contain false edges, also called *false positives*. On the other hand, if the threshold value is too high, real edges will be suppressed, edges also called *false negatives*.

The global thresholding of the gradient image has been considered in the development of the canny edge detector [16]. Canny is widely used due to its performance regarding time and quality of the calculated edges. The method represents an optimal edge detection algorithm designed to achieve three objectives:

- *optimal edge detection*: all edges in the image should be found, as close as possible to the real edges;

- *optimal edge points localization*: the position of the obtained edges should be as close as possible to the real edges;
- *optimal edge point response*: the calculated edge should be as thin as possible, namely the detector should not identify multiple edges where only a single edge exists.

The canny edge detector involves three sequential steps. At the beginning the input image $f(x, y)$ is convolved with a *Gaussian smoothing filter*:

$$G(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{2.15}$$

where $G(x, y)$ is a Gaussian function with standard deviation σ. This type of filtering suppresses noise in the input image, since the first derivative of a Gaussian used in calculating the image gradient is susceptible to noise present on raw unprocessed image data.

The second step of the canny edge detection algorithm is to calculate the image gradient magnitude, $M(x, y)$, and direction (angle), $\alpha(x, y)$:

$$M(x, y) = \sqrt{g_x^2 + g_y^2}, \tag{2.16}$$

$$\alpha(x, y) = tan^{-1}\left[g_x/g_y\right], \tag{2.17}$$

where g_x and g_y are the horizontal and vertical directions of the image gradient, respectively.

The obtained edges are thinned using *non-maximum suppression*, that is, four types of filter masks are used to specify a number of discrete orientations of the edge normal: horizontal, vertical, $+45°$ and $-45°$.

Finally, the obtained gray level image is binarized using a technique named *hysteresis thresholding* which uses two thresholds: a low T_L and a high T_H threshold. The pixels above T_H are considered "strong" edge pixels and the one below T_L false edges. The pixels belonging to the interval $[T_L, T_H]$, named "week" edge pixels, are considered edges if they are connected to the already detected "strong" pixels.

According to [82], the low threshold can be expressed as a function of the high threshold as:

$$T_L = 0.4 \cdot T_H, \tag{2.18}$$

Color based segmentation

One natural way to segment color images is through the HSI color model. This model retains color information on separate image planes, that is the hue $f_H(x, y)$ and saturation $f_S(x, y)$ images.

In the hue image, color information is represented as pixel values belonging to the interval $[0, 2\pi]$, or $[0, 359]$. Having in mind that color is represented by the angle H, defined on the unit circle from Figure 2.3(b), each pixel value in the interval $[0, 359]$ corresponds to a particular hue. Because in some computer implementations images are stored using 8 bit arrays (255 pixel values), the hue interval has been divided by two to fit in this representation. In this case, the hue varies in the interval $[0, 179]$.

In order to differentiate between object colors, the hue component was divided into color classes which take values in the interval $[T_{low}, T_{high}]$:

$$C_l \in [T_{low}, T_{high}], \tag{2.19}$$

where l represents the number of the color class and T_{low} and T_{high} the minimum and maximum color values across the object's pixels. A pixel is considered as belonging to one color class if its hue value resides in one of the object color class values.

The application of the histogram thresholding method from Equation 2.20 to the hue plane image $f_H(x, y)$ results in a color segmentation method which separates a specific colored object from the background:

$$t_H(x, y) = \begin{cases} 1, & \text{if } f_H(x, y) \in C_l, \\ 0, & \text{if } f_H(x, y) \notin C_l, \end{cases} \tag{2.20}$$

where $t_H(x, y)$ represents the binary thresholded hue plane image. For the sake of clarity, an object color class C_l is referred in this thesis as the *object thresholding interval* $[T_{low}, T_{high}]$.

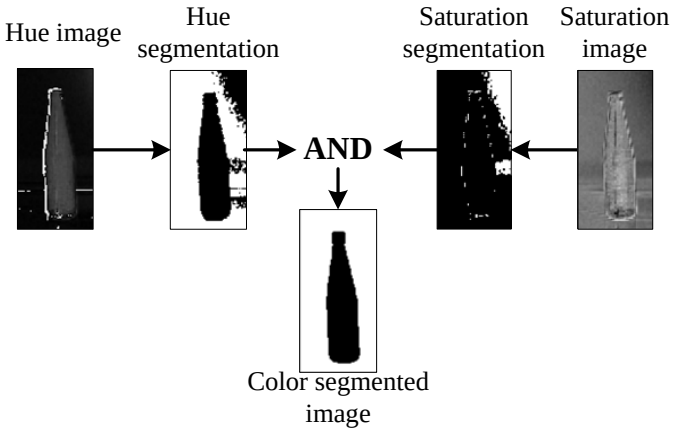
The saturation component is used in color segmentation as a mask that isolates further regions of interest in the hue image. Typically, the saturation image is thresholded using Equation 2.12 with a threshold equal to 10% of the maximum value in the saturation image [30]. The result is the binary image $t_S(x, y)$. The final color segmentation result is a binary image obtained through a pixel level logical **AND** operation between t_H and t_S:

$$t(x, y) = t_H(x, y) \textbf{ AND } t_S(x, y), \tag{2.21}$$

where $t(x, y)$ is the final binary image output of color segmentation. The truth table of the logical **AND** operation is given in Figure 2.4(b). An example of color segmentation using the described procedure can be seen in Figure 2.4(a).

2.4.2. Closed-loop image segmentation

The image segmentation step from Figure 2.1 plays a crucial role in the 2D recognition and 3D reconstruction of a robot's environment. The operations following segmentation can provide reliable results only if the input segmented image is of good quality, that is with well segmented objects.



(a)

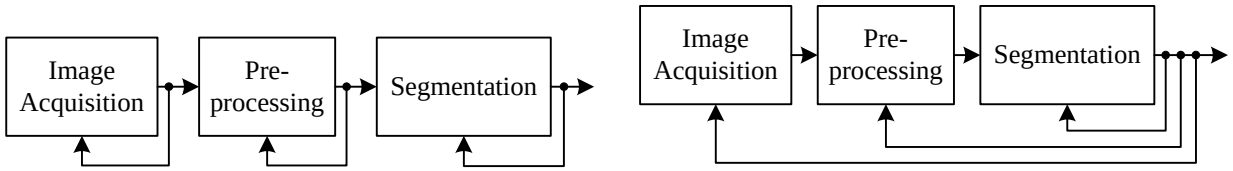
p	q	p **AND** q (also $p \cdot q$)
0	0	0
0	1	0
1	0	0
1	1	1

(b)

Figure 2.4.: (a) Color segmentation example. (b) Logical **AND** operation.

The above presented segmentation methods provide good results when they are used in constant reference conditions, like, for example, constant illumination. Here, the reference is represented by the conditions for which the segmentation parameters were manually tuned. If these conditions are changed segmentation will fail to produce reliable results. If, for example, illumination varies, constant segmentation parameters will not be able to properly extract the objects of interest.

A solution for the above problem is to automatically adjust the segmentation parameters. One possibility for this is to use the feedback mechanisms presented earlier in Chapter 2.1. In [83] two feedback structures for control of image segmentation are proposed. In Figure 2.5(a) sequential closed-loops at different levels of image processing are introduced. Feedback information of each processing stage is used here for improving the robustness of that specific stage. A second type of closed-loop for improvement of image segmentation is the cascade control structure from Figure 2.5(b), where feedback information for different loops is measured at the same image processing stage. In this case the "inner" control loops provide improvement of the processing result to a certain level while its further improvement is achieved in "outer" loops.



(a) (b)

Figure 2.5.: Sequential (a) and cascade (b) control structures for control of image segmentation.

In [83], a measure of output binary image quality is proposed as a feedback variable for the control structures from Figure 2.5. The so-called two-dimensional (2D) entropy measure, defined in Equation 2.22, is to be used in both region and boundary based

segmentation. The 2D entropy aims at quantifying the degree of connectivity between object pixels.

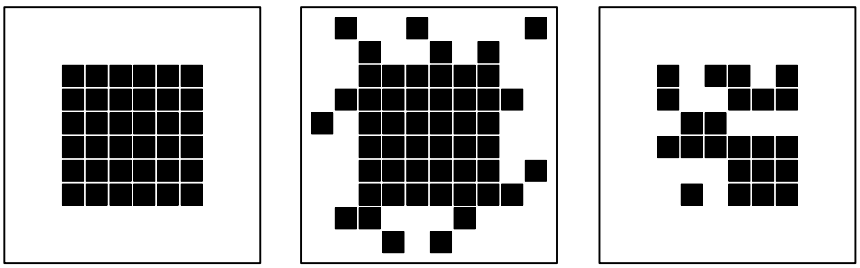
$$S_{2D} = -\sum_{i=0}^{8} p_{(1,i)} \log_2 p_{(1,i)}, \tag{2.22}$$

where $p_{(1,i)}$ is the relative frequency, that is, the estimate of the probability of occurrences of a pair $(1, i)$ representing a foreground pixel surrounded with i foreground pixels:

$$p_{(1,i)} = \frac{number\ of\ black\ pixels\ surrounded\ with\ i\ foreground\ pixels}{number\ of\ foreground\ pixels\ in\ the\ image}. \tag{2.23}$$

The 2D entropy S_{2D} can be considered as a measure of disorder in a binary segmented image since, as demonstrated in [83], *the higher the 2D entropy, the larger the disorder (noise, breaks) in a binary image is.* Hence, the goal of the feedback control structures would be the minimization of S_{2D}.

The relation between the 2D entropy 2.22 and a region based segmented image is best understood on the synthetic images from Figure 2.6, where three types of segmented images are shown: ideal, noisy and broken, respectively. The results of Equation 2.22 on the three binary images are presented in Table 2.2. As can be seen, the 2D entropy has the minimum value for the case of the ideally segmented image.



(a) (b) (c)

Figure 2.6.: Possible connectivity of object pixels in region based segmentation. (a) Ideal. (b) Noisy. (c) Broken.

Table 2.2.: The 2D entropy from Equation 2.22 of the synthetic images from Figure 2.6.

	Ideal	Noisy	Broken
2D entropy S_{2D}	1.3921	2.3628	2.7499

A similar demonstration as above, ending with same conclusions, is also given in [83] for the case of boundary segmented images. The benefit of the measure in Equation 2.22 has been demonstrated in [83, 31, 84, 86, 85] for the case of two industrial image processing applications, that is, improvement of character recognition on metallic surfaces and improvement of corner detection in images taken from a ship welding scenario.

In Chapter 4, two closed-loop segmentation methods, based on inclusion of feedback control in image processing, are proposed. Their purpose is to improve the robustness of vision systems in service robotics.

2.5. Feature extraction

Feature extraction, also known as representation and description, is the intermediate operation between low and high level image processing stages. From input binary images, attributes, or features, are extracted. These features should describe the interdependency between the segmented pixels. This process is also known as the transformation of the input segmented image into a set of features.

One straightforward purpose of feature extraction is the classification of the objects present in the imaged scene. For this reason, the chosen features for describing the objects must be invariant to translation, rotation, scale, or mirroring. The first step in feature extraction is to extract the boundary of the contours from the raw binary image and convert them into a form suitable for analysis, a process also known as contour extraction.

Contour extraction

Depending on the segmentation type, region or boundary based, the objects in a binary image can be represented by blobs of foreground pixels, for the case of region segmentation, or from connected edge pixels, for the case of boundary segmentation. The principle behind contour extraction is to order the pixels on the boundary of segmented objects in a clockwise, or counterclockwise, direction. The procedure is also referred to as *boundary* (*border*) *following* [30].

A popular method for contour extraction is the so-called *chain codes*. Chain codes describe a boundary by a connected sequence of straight-line segments of specific length and direction. It is typically based on 4- or 8-connectivity of the segments. In this type of representation, also known as a *Freeman chain code*, the direction of each segment is coded as a sequence of directional numbers, from one pixel to the next [30]. An example of an 8-directional chain code of the simple object boundary from Figure 2.7 is:

0 0 0 0 6 0 6 6 7 7 6 4 5 6 6 4 4 4 4 4 4 2 4 2 2 2 2 2 0 2 2 0 2

Such a digital boundary can be further approximated by a polygon. The purpose of polygonal approximation is to transform the extracted chain code into a shape that captures the essence of the boundary and uses the fewest possible number of segments. A popular method used in image processing for polygonal approximation is boundary description by a *minimum-perimeter polygon* [30]. From the calculated polygon a number of features can be extracted, such as its *area*, *perimeter*, *diameter*, *major* and *minor axis* together with their *eccentricity* (ration of major to minor axis), *curvature* etc.

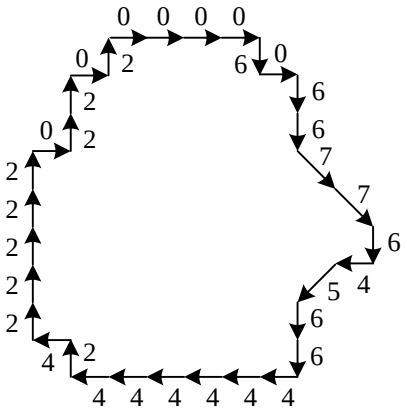


Figure 2.7.: 8-directional chain-coded boundary of a segmented object.

The extracted boundary, or polygon, should be represented for classification by a set of descriptors invariant to linear transformations like translation, rotation, scale, or mirroring.

Moment invariants

A set of invariant object descriptors are a set of seven coefficients proposed by Hu [38]. These coefficients are derived from the moments of the object boundary extracted with an appropriate contour extraction method. In case of a digital image intensity function $f(x, y)$, the moment of order $(p + q)$ is:

$$m_{pq} = \sum_{x} \sum_{y} x^p y^q f(x, y), \tag{2.24}$$

where x and y are pixel coordinates in the considered image boundary region. The central moments μ_{pq} are defined as:

$$\mu_{pq} = \sum_{x} \sum_{y} (x - \bar{x})^p (y - \bar{y})^q f(x, y), \tag{2.25}$$

where $p, q = 1, 2, 3, ...$, $\bar{x} = m_{10}/m_{00}$, $\bar{y} = m_{01}/m_{00}$. In this thesis, for object recognition, two invariant moments are used:

$$\begin{cases} I_1 = \eta_{20} + \eta_{02}, \\ I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2, \end{cases} \tag{2.26}$$

where η_{pq} is the normalized central moment:

$$\eta_{pq} = \mu_{pq} \cdot \mu_{00}^{-1-\frac{p+q}{2}}. \tag{2.27}$$

Hough transform

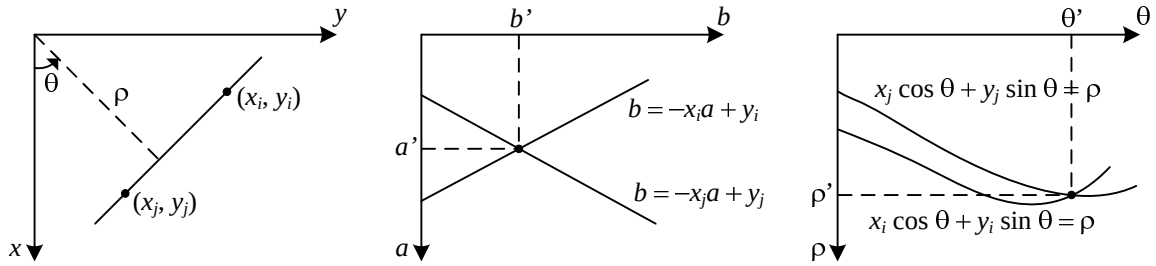
One problem when using boundary based segmentation is that very often the obtained contour edges are not connected (e.g. small breaks between the edge pixels). This phenomenon happens due to noise in the input image, non-uniform illumination and other effects that introduce discontinuities in the intensity image.

The *hough transform* [37] is a method used in linking edge pixels based on shape. Although any type of shape can be represented by the so-called *generalized hough transform*, in practice, because of computational expenses, shapes like lines, circles and ellipses are used. In this thesis, the hough transform is used in combination with the canny edge detector for finding boundaries of textured objects, as explained in Chapter 4.

The principle of the hough transform for lines detection, represented in Figure 2.8, is based on the general equation of a straight line in slope-intercept:

$$y_i = ax_i + b, \tag{2.28}$$

where (x_i, y_i) is a point on the line.



(a) (b) (c)

Figure 2.8.: Hough transform principle. (a) $xy-$image plane. (b) ab parameter space. (c) $\rho\theta$ parameter space.

Through point (x_i, y_i) an infinite number of lines pass, all satisfying Equation 2.28 for different values of a and b. If, instead of the $xy-$image plane, the line equation is represented with respect to the $ab-$plane from Figure 2.8(b), also named *parameter space*, then the equation of a single line for a fixed pair (x_i, y_i) is obtained:

$$b = -x_i a + y_i. \tag{2.29}$$

If a second point (x_j, y_j) is collinear with the point (x_i, y_i) in the $xy-$image plane then, in parameter space, the two corresponding lines intersect at some point (a', b'), as presented in Figure 2.8.

One problem with using the parameter space ab is that a, the slope of a line, approaches infinity as the line approaches the vertical direction. A solution to this problem is to use the normal representation of a line:

$$xcos\theta + ysin\theta = \rho. \tag{2.30}$$

Using relation 2.30 the lines in the $xy-$image plane are represeted by sinusoidal curves in the $\rho\theta-$parameter space from Figure 2.8(c). The intersection of the sinusoidal curves (ρ', θ') represents collinearity of points in the $xy-$image plane.

The parameter space $\rho\theta$ is subdivided into so-called *accumulator cells*. An accumulator cell $A(i, j)$ corresponds to a quantification of the point (ρ_i, θ_j) in parameter space coordinates. Initially the accumulator cells are set to zero. For every foreground pixel (x_k, y_k) in the $xy-$image plane, the parameter θ is varied along the $\theta-$axis. For each value of θ, the corresponding ρ is obtained using Equation 2.30. If a specific θ_p results in a solution of ρ_q, then the accumulator cell $A(p, q)$ is increased with value 1:

$$A(p, q) = A(p, q) + 1. \tag{2.31}$$

At the end of the transformation, a number n in $A(i, j)$ means that n points in the $xy-$image plane lie on the line $xcos\theta_j + ysin\theta_j = \rho_i$. The more foreground pixels lie on a line the higher the respective accumulator value is. Since the accumulator represents collinearity for all the foreground pixels in a binary image it is meaningful for representation to threshold it, namely to consider as lines only the ones which have an accumulator value higher than a specific threshold T_{HG}. In Chapter 4 of this thesis, an algorithm for automatic calculation of the optimal threshold T_{HG}, based on the results of feature extraction, is proposed.

2.6. Classification

The final stage of 2D image processing is the classification of the extracted features. In classification, a set of features is usually named a *pattern vector*. The field of feature classification includes a broad range of decision-theoretic approaches aimed at labeling the image features to one or more distinct classes. An algorithm that fulfils this process is commonly referred to as a *classifier*. According to how class knowledge is specified, classification can be separated in two categories:

- *supervised classification* where object classes are a priori specified by an analyst;
- *unsupervised classification* where input data is automatically clustered into sets of prototype classes; the number of desired classes is usually specified.

The simplest approach for classification is the so-called *Minimum (Mean) Distance Classifier*, which computes the distance between a measured, unknown, pattern vector and the mean of a set *prototype vectors*. A prototype vector is composed of a number of training features a priori specified:

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x}_j, \quad j = 1, 2, \ldots, W, \tag{2.32}$$

where $\mathbf{x}_j$ is a pattern vector, N_j is the number of pattern vectors from class ω_j and W is the number of pattern classes.

In this thesis, the Euclidean distance is used in assigning the class membership of an unknown pattern vector $\mathbf{x}$, that is $\mathbf{x}$ is assigned to the class which has the closest prototype to it. Since the training prototype vectors are given a priori, the method belongs to the supervised classification category.

Although in literature a large number of powerful classification methods can be found, in this thesis the stress is on improving the overall robot vision image processing chain from Figure 2.1. The complexity of classification is hence maintained at a medium level. In a number of robotic systems [68, 42] the robustness of image processing is achieved using powerful classification algorithms. A drawback of this approach is further 3D reconstruction, since features obtained from image segmentation are used for defining the attributes of the object in the virtual 3D space. Motivation for robust image segmentation will be given in Chapter 4.

2.7. 3D reconstruction

According to the results of classification, a decision is made regarding how the extracted object features will be used in the 3D reconstruction phase. For example, if an object is classified as a bottle, then the object's features used in 3D reconstruction will be the top and bottom coordinates of the object and its diameter. On the other hand, if an object is classified as a book, then the features important for 3D reconstruction are its four corners.

A classical approach to 3D reconstruction is the so-called *epipolar geometry* [34], which refers to the geometry of stereo vision. The principle behind epipolar geometry relies on the fact that between an imaged point in the real 3D world and its projection onto 2D images exist a number of geometrical relations. These relations are valid if the cameras are approximated using the *pin hole camera model* [34]. This model refers to an ideal camera with its aperture described as a point and no lenses are used to focus light. Knowing the relative position of two cameras with respect to each other, the imaged 3D point can be reconstructed in a 3D virtual environment using triangulation.

The position of a camera in the real world is described by a *projection matrix* obtained through the process of *camera calibration* which calculates the POSE of the camera with respect to a known reference coordinate system. For the case of a stereo camera two projection matrices are used, one for the left camera lense Q_L and one for the right lense Q_R. A projection matrix is composed of two types of parameters:

- *intrinsic parameters* C_{int}, which describe the internal characteristic of the camera, that is focal length, intersection of the optical axis with the image plane, pixel aspect

ration and pixel skew;

- *extrinsic parameters* C_{ext}, representing a homogeneous transformation describing the POSE of the camera with respect to a reference coordinate system to which the reconstructed 3D points are reported.

When both the intrinsic and extrinsic camera parameters are known, the full camera projection matrix can be determined as

$$Q = C_{int} \cdot C_{ext}. \tag{2.33}$$

In this thesis, the 3D reconstruction module is considered as a black box which requires as input the object type, obtained through classification, and its extracted features.

3. ROVIS machine vision architecture

Integrating visual perceptual capabilities into the control architecture of a robot is not a trivial task, especially for the case of service robots which have to work in unstructured environments with variable illumination conditions. This is also the case of the machine vision architecture ROVIS (*RObust machine VIsion for Service robotics*) with which the service robotic system FRIEND is equipped.

In machine vision systems, more particular service robots, an important role is played not only by the image processing algorithms itself, but also by how the visual processed information is used in the overall robot control architecture. This process has high complexity since image processing involves the management of a large quantity of information which has to be used in high level action planning. A good candidate for modeling large scale systems, like the control architecture of a service robot, is the *Unified Modeling Language* (UML) [65] which wraps together several graphical language tools for modeling object-oriented problems. A short description of UML can be found in Appendix B.

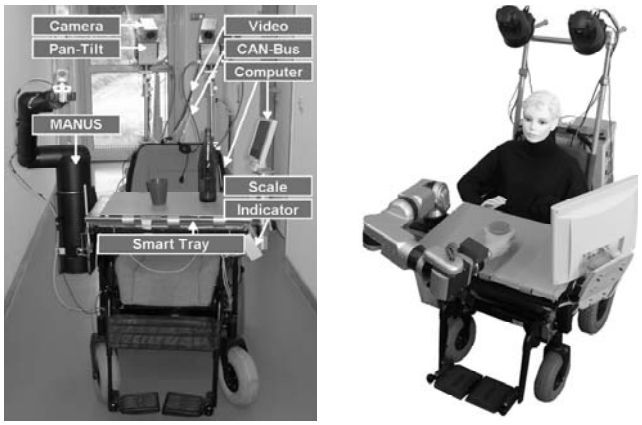
In this chapter, the concept of the vision system ROVIS, modeled with the help of UML, is presented together with its intergration into the overall control architecture of the robotic system FRIEND. The stress here is on the concept of ROVIS and on the structure of the information flow within the vision system. To begin with, as comparison, the vision systems of previous FRIEND robotic platforms are presented.

3.1. FRIEND I and II vision systems

The robotic systems FRIEND I and II (*Functional Robot with dexterous arm and user-frIENdly interface for Disabled people*) are service robots developed at the Institute of Automation (IAT) from University Bremen. The research started back in 1997 when the building of the first FRIEND prototype began.

Basically, all FRIEND robots consist of a manipulator arm mounted on an electrical wheelchair and various sensors needed to understand the surrounding environment for the purpose of manipulator path planning and object grasping. The first FRIEND [64], presented in Figure 3.1(a), was equipped with a MANUS arm, while the second version, FRIEND II [104] (see Figure 3.1(b)) was equipped with a 7-DoF arm with functional specifications given by IAT. Another key component of both robots is the *Human-Machine Interface* (HMI) used to communicate with the user of the robotic platform. The importance of the HMI is related to the overall robot control architecture MASSiVE (*MultiLayer Architecture for SemiAutonomous Service-Robots with Verified Task Execution*), presented

in Chapter 3.4.3. MASSiVE has as core concept the integration of the cognitive capabilities of the user in the working scenarios of the robot [61, 62, 63]. For example, the user can be asked by the system to assist it at specific operational stages where autonomous task execution fails (e.q. the object of interest was not detected, hence the system will ask the user to manually control the movement of the manipulator arm in order to bring it to the grasping position) [80]. This concept of "human-in-a-loop" was also used in other assistive robotic systems presented in [100, 23, 109, 25].



(a) (b)

Figure 3.1.: IAT FRIEND I (a) and II (b) assistive robots.

The first experiments involving vision in FRIEND were based on the visual servoing principle [21]. The robotic arm MANUS was equipped with an "eye-in-hand" camera used for detecting a marker placed on the object to be grasped [51].

The "eye-in-hand" camera setup was replaced with a pair of pan-tilt-head (PTH) zoom cameras mounted on a rack behind the user of the robotic system [104]. Although the two cameras form a stereo vision system, they were used in a visual servoing manner where no camera calibration is needed. In this second case, the manipulator arm was equipped with an active marker which was tracked in the input image, alongside with the tracking of features of the object of interest. Furthermore, the usage of color as a feature to track was introduced as a replacement to the artificial marker used before. Objects were divided into color classes and detected as separate entities in the environment [104, 103]. The implemented visual servoing control structure is displayed in Figure 3.2.

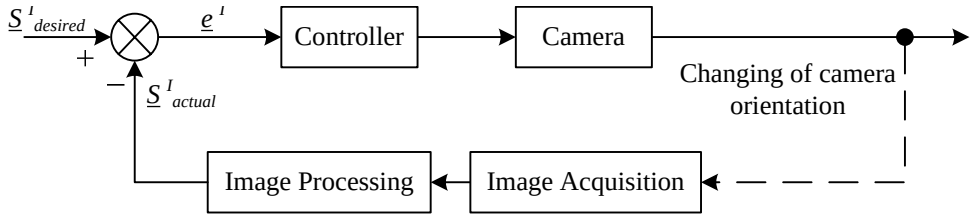


Figure 3.2.: Block diagram of the control structure for adjustment of camera parameters in the FRIEND I system.

In visual servoing with an "eye-in-hand" camera, the goal of the closed-loop system is to minimize the control error e^I represented as the difference between the current position of the tracked image feature $S^I_{actual} = (u, v)^T$ and the coordinates of the image center $S^I_{desired} = (u_0, v_0)^T$. For the case of static cameras mounted behind the user, the error is measured with respect to the position of the active marker mounted on the manipulator arm: $S^I_{desired} = (u_{arm}, v_{arm})$ [105]. The control error can be expressed as:

$$e^I = [u - u_i, v - v_i]. \tag{3.1}$$

where (u, v) represents pixels coordinates in the input image I and:

$$(u_i, v_i) \in \{(u_0, v_0), (u_{arm}, v_{arm})\}. \tag{3.2}$$

The control signal is calculated using a proportional controller and the inverted image Jacobian matrix J^{-1}. This matrix describes the relationship between pixel motion in images and changes in camera orientation [105]. A comprehensive survey on visual servoing can be found in [39, 92].

The development of the FRIEND II robot, depicted in Figure 3.1(b), represents the next big step towards the concept of the vision architecture ROVIS described in this thesis. Because of the high complexity of FRIEND, the visual servoing principle was replaced with a "look-and-move" strategy which separates machine vision from manipulator path planning and object grasping control. Also, the MANUS manipulator was replaced with a 7-DoF AMTEC® arm for improving object manipulation performance. Details about the components and algorithms used in FRIEND II can be found in [104].

One drawback of the FRIEND II vision system was the lack of a vision architecture to sustain and manage the large amount of image processing operations used in visual robot control. The vision algorithms were implemented as sequential functions in the MASSiVE architecture.

Another existing problem in FRIEND II is represented by the lack of image processing robustness with respect to external influences, like variable illumination conditions. The vision system of FRIEND II required a constant illumination to reliably detect objects of interest. Also, because of the fixed color classes, only objects which were a priori learned by the robot's control system could be recognized, thus making the functionality of FRIEND rigid with respect to the working scenarios. In order to overcome these problems the machine vision system ROVIS is introduced.

3.2. The ROVIS concept

The goal of ROVIS is to recognize objects of interest and reconstruct them in a 3D virtual environment for the purpose of manipulative motion planning and object grasping [73].

Although the basic concept of ROVIS is not derived from the neuro-biological func-

tioning of the human brain, it is inspired from how a human person visually analyzes a scene, namely cognitive psychology [69]. This process is depicted in the environmental setting from Figure 3.3, where a typical all-day-living scene is found. When a human visualizes a scene he/she is not analyzing the whole visual field in a single moment of time but focuses his/her attention to several objects present in the environment, for example a book shelf at moment t_i or a cup of coffee at moment t_{i+K}. In computer vision terms, the focus of attention of a human on a particular area in the visualized scene can be interpreted as a *Region of Interest* (ROI) in an image.

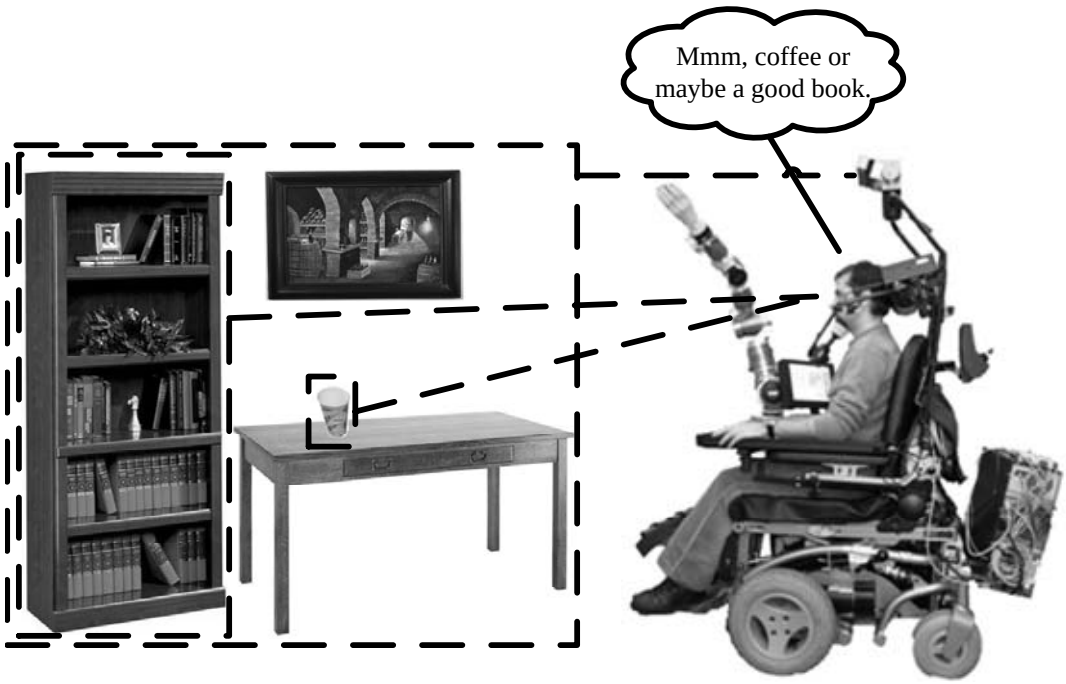


Figure 3.3.: Human focus of attention in a complex scene.

The ROVIS architecture has been developed starting from the above description and the inclusion of feedback structures at image processing level.

3.2.1. Classification of objects of interest

In service robotics applications, a large number of different objects with different characteristics are found. This can be seen in Figure 3.4, where example scenes from support scenarios of FRIEND are shown. Details regarding the support scenarios are given in Chapter 3.4. In ROVIS, the objects are saved in a virtual 3D model of the surroundings, named *World Model*. From the image processing point of view, the objects of interest are classified into two categories:

- *Container objects*, which are represented by relatively large objects that have a fixed location in Cartesian space (e.g. fridge, microwave, book shelf, tables, library desk etc.);

- *Objects to be manipulated*, which can be found anywhere in the scene, inside or outside container objects (e.g. bottles, glasses, meal-trays, books etc.).

(a) (b)

Figure 3.4.: Typical service robotic scenes from the FRIEND support scenarios. (a) Activities of daily living. (b) Library.

In FRIEND, grasping an object to be manipulated from a container is represented, for example, by grasping a bottle from a fridge.

Since objects to be manipulated come in various characteristics (e.g. different shapes and colors), their detection has to be made without use of a priori information regarding their structure. In Chapter 6, two robust object recognition methods that cope with lack of a priori knowledge regarding objects are proposed. Both methods rely on the robust segmentation algorithms presented in Chapter 4.

In order to plan a collision free manipulator motion, container objects have to be reliably detected. Bearing in mind that the container objects in the FRIEND environment are a permanent feature of the scenarios, the SIFT method [10] is used for their localization and 3D reconstruction. This method uses a model image to train a classifier off-line. During on-line system operation, the SIFT algorithm searches for the model image in the scene through a matching based algorithm. Once the model image has been detected, its Position and Orientation (POSE) can be reconstructed. Knowing the position of the model image and the geometry of the container, its POSE can be reconstructed. The POSE of containers is one way to define the image ROI, as explained in Chapter 5.

The different objects of interest are represented in ROVIS by *object classes*. Whenever 2D object recognition and 3D reconstruction is activated, relevant information of object classes, involved in the robot's operation, are made available in the World Model. This information is specified via object class characteristics that are encoded in an extensible ontology. This ontology is depicted in Figure 3.5, where the objects involved in the ADL scenario of FRIEND, as well as in the Library scenario, are pointed out. As an example, for the case of the fridge, which is a part of the ADL scenario, the characteristics *IsContainer*, *HasShelves* and *HasDoor* will be made available. For the tray with meal (meal-tray) the knowledge about its components *Plate*, *Spoon* and *Lid* is supplied.

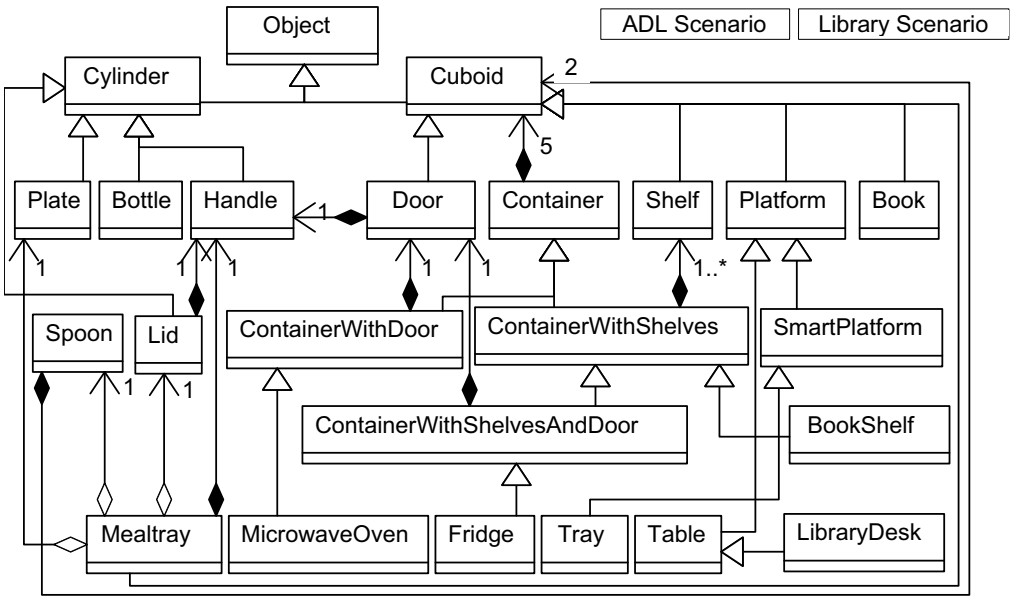


Figure 3.5.: Hierarchical ontology of objects involved in service robotic support scenarios (ADL and Library).

The coded object class information is used in classifying segmented objects, improve segmentation and final 3D reconstruction, as will be explained in Chapter 6. The extracted object features are used together with object classes to construct a virtual 3D environment. In ROVIS, feature extraction is divided into two categories:

- *feature extraction for object classification*, which deals with the extraction of those attributes needed to recognize certain objects in the 2D image plane;
- *feature extraction for 3D reconstruction*, represented by the extraction of features from 2D images than can describe the 3D shape of the imaged objects; for this second type of feature extraction, objects attributes from synchronized stereo images are acquired, together with the geometrical relationship between the stereo camera lenses.

Object classes provide a description of how extracted object features are to be used in 3D reconstruction, that is, the positions of the feature points of an object.

3.2.2. ROVIS block diagram

Following the above reasoning, the vision architecture ROVIS has been developed, with its block diagram presented in Figure 3.6. Arrows connecting the blocks illustrate the flow of information through the ROVIS system as well as the connections of the ROVIS components with the external modules, the HMI and other reactive operations in the robotic system. The HMI handles input commands from the user, or patient, to the FRIEND robot and subsequently to ROVIS. Depending on the dissabilities of the patient (e.g. in case of spinal cord injuries, which vertebra is fractured), different input devices

are to be used. If the patient still has some motoric capabilities in its upper limbs, than it can send commands to FRIEND through a hand joystick. For patients which are disabled from the neck down, a chin joystick is used as input device. Alternative solutions like speech recognition and *Brain Computer Interface* (BCI) are also implemented for patients with no motoric abilities. These devices are used to control a cursor on a display monitor. Different buttons on the display signify different robotic commands.

As can be seen from Figure 3.6, there are two main ROVIS components: *hardware* and *object recognition and reconstruction chain*, also referred to as the image processing chain. The connection between ROVIS and the overall robot control system is represented by the World Model, where ROVIS stores the processed visual information. The robustness of ROVIS with respect to external influences (e.g. complex scenes, or variable illumination conditions) resides in two key aspects:

- automatic calculation of an image ROI on which further image processing operations are applied;
- inclusion of feedback structures within vision algorithms and between components of ROVIS for coping with external influences.

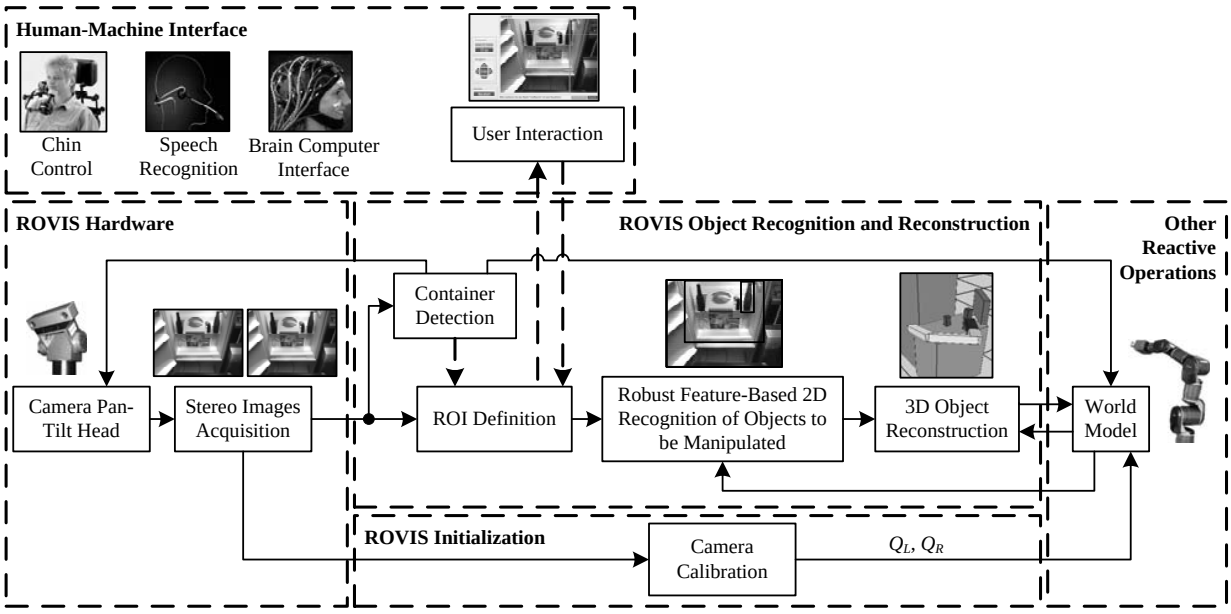


Figure 3.6.: Block diagram of ROVIS, the robust vision architecture for service robotics.

The ROVIS hardware consists of a stereo-camera system mounted on a PTH. The camera views the scene in front of the service robot. The viewing angle of the sensors can be changed through the pan-tilt control so that the container required for a particular working scenario can be detected in the image. This is illustrated in Figure 3.6 by the feedback from *Container Detection* to the *Camera Pan-Tilt Head* block.

The vision system is initialized through the ROVIS *Camera Calibration* procedure [4], which calculates the left and right camera projection matrices, Q_L and Q_R, respectively.

These matrices describe the homogeneous transformation between the robot's reference coordinate system W, located at the base of the manipulator arm, and the left C_L and right C_R coordinate systems of the lenses of the stereo camera, respectively. In this thesis, the reference coordinate system will be named as the World coordinates. As it will be explained in Chapter 6, the projection matrices are used by the 3D Object Reconstruction module to calculate the POSE of the objects to be manipulated with respect to the world coordinates. The calculated calibration data is further stored in the World Model.

The ROVIS object recognition and reconstruction chain consists of a sequence of image processing operations used in the extraction of the features needed for both the 2D recognition and 3D reconstruction of objects. One main feature of ROVIS is to apply the vision methods on the image ROI rather than on the whole image. This is motivated by the observation that people focus their visual attention on the region around an object when they grasp it, as illustrated in Figure 3.3.

In Fig. 3.7 the interconnections of ROVIS with other components is illustrated using UML use cases. ROVIS is connected with two other use cases: the HMI and the system's World Model. The user of FRIEND is modeled in Fig. 3.7 as an actor. He interacts with ROVIS through the HMI. The *Environment*, or scene, in which FRIEND operates is modeled also as an actor connected to the vision architecture. The *Sequencer*, detailed in Chapter 3.4, is modeled as the requester of visual information. The predefined task knowledge with which the Sequencer plans sequences of operations is formally specified and verified a priori in a scenario-driven process [79]. It is flexibly applicable in different situations and environments due to the usage of object classes, as detailed below. The visual data processed by ROVIS is finally stored in the World Model.

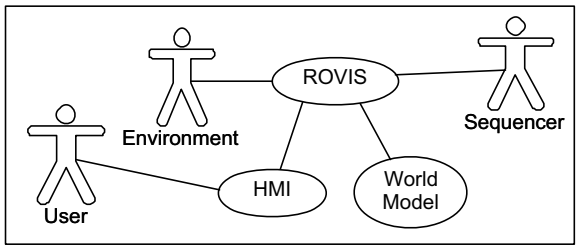


Figure 3.7.: ROVIS general use case diagram.

Another important requirement for ROVIS is the automatic construction of the object recognition and 3D reconstruction chain, which puts together sequences of image processing operations needed for object detection. The model of this process is depicted in Fig. 3.8. The five types of basic image processing operations, or primitives, are modeled as five use cases: *Image Pre-processing*, *ROI Segmentation*, *Feature Extraction*, *Object Classification* and *3D Reconstruction*. The ROI definition algorithms and the camera calibration methods are considered pre-processing operations. In order to achieve a high robustness of the vision system with respect to external influences, the five types of image processing methods are connected to an extra use case which models feedback structures

within image processing operations. Depending on the type of objects that have to be recognized, appropriate object recognition operations are called. For example, for the case of uniform colored objects, region based recognition is to be used for object detection [30]. On the other hand, the detection of textured objects is performed via boundary based object recognition [30]. The dynamic image processing chain is automatically constructed by the *Algorithms Executer* which connects the vision methods needed by a specific scenario.

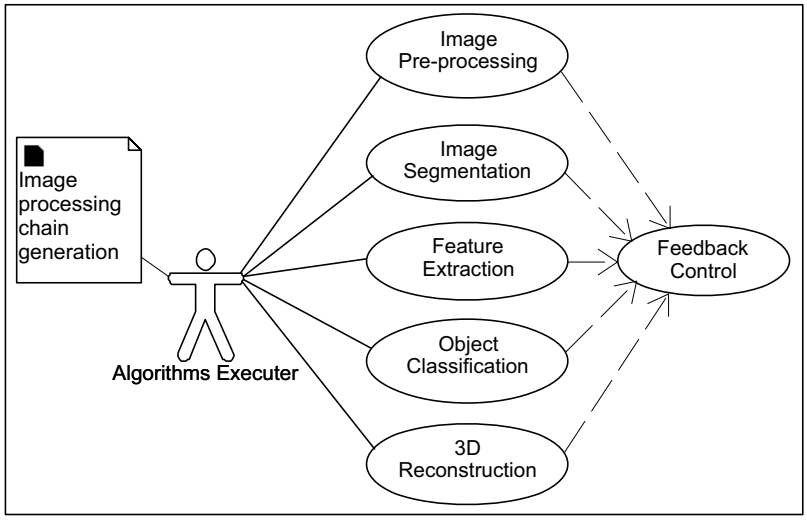


Figure 3.8.: UML modeling of basic image processing primitives.

3.2.3. Interest image areas in ROVIS

An image ROI can be defined for two cases which differ with respect to the level of a priori knowledge about the location of the object to be manipulated within the image. In the first case only a partial knowledge about the object environment is available. For example, as explained in Chapter 3.2.1, in the FRIEND system the available information through object classes is of the form: "the object is in the fridge" or "the object is on the shelf". Starting from the reconstructed 3D POSE of the detected container, the container region in the image is obtained using 3D to 2D mapping. If the container is not in the Field of View (FOV) of the stereo camera, top-down ROI definition through camera gaze orientation is used, as explained in Chapter 5.3. In this case the camera orientation is changed in a visual feedback manner until the container is positioned in the middle of the imaged scene. The recognition of the container is done either using the robust object boundary detection method from Chapter 4.2, or by SIFT model based recognition. The resulting image region enclosing the container, in which the object of interest is located, represents the image ROI. Hence, in this case, the defined ROI encloses all the objects present in the container and not just the object of interest. For example in the ADL scenario, where one of the task of the manipulator is to fetch a bottle with a drink from the fridge, such situation corresponds to a user's command "I want a drink".

The second possible case regarding ROI definition is the case where precise information on the object's position within the image is available through the HMI. For example, the user can locate the object of interest by using a particular action, such as clicking on the displayed image using a special input device, like a chin joystick, as illustrated in Figure 3.6. For the case of ADL scenario, starting from the user's command "I want this drink" and an interest image point defined by the user, the size of the rectangular image ROI is automatically adjusted in order to fully bound the object of interest, as it will be explained in the closed-loop image ROI definition algorithm from Chapter 5.

3.2.4. Robustness in ROVIS through feedback mechanisms

In order to improve the robustness of the ROVIS system, the inclusion of feedback structures in and between the various processing components has been suggested.

The development of an overall feedback strategy for controlling ROVIS would be to complex to implement and analyze. The solution to the control problem is the application of the *decomposition technique* [40] to the vision architecture. The decomposition technique is normally used when developing control methods for large complex plants where the control of the overall process would be extremely difficult due to the large number of variables. Thus, the vision system from Figure 3.6 is not treated as a whole process that has to be controlled, but as a composition of different subsystems that can be individually controlled (e.g. control of ROI definition, control of object recognition etc.). The overall robustness of the system can be achieved by developing robust subcomponents of ROVIS. In ROVIS two types of such closed-loops are introduced:

- feedback structures within image processing operations;
- feedback loops between the various components of ROVIS.

Feedback structures within image processing operations

The application of control in image processing deals with the inclusion of feedback loops within image processing operations to improve their robustness with respect to external influences. In [6, 7], two closed-loop image processing algorithms, used in the FRIEND system for object recognition, are introduced. The purpose for the inclusion of feedback loops in the vision operations of ROVIS is to automatically determine the optimal working points of the parameters of these operations, thus achieving system robustness with respect to external influences.

In [83], two types of closed-loops for image processing purposes are proposed, both detailed in Chapter 2. The basic principle of feedback in image processing is to automatically adjust the values of the processing parameters according to the output image quality measure, or *controlled variable*. Using this measure an error between the reference values and the current output can be calculated. The resulted error is used in determining the optimal working point of the image processing tuning parameter, also called *actuator*

variable. The stress in [83] is on the importance of the choice of the pair *actuator variable – controlled variable* for the success of the vision application.

The novel vision algorithms presented in Chapters 4, 5 and 6 are based on the concept of including feedback control within image processing operations in order to improve their robustness.

Feedback loops between ROVIS components

This second type of closed-loops is used for setting a synchronization method between the various components of the vision system. As an example, in Figure 3.6 a loop between the image ROI definition algorithms and the control unit of the camera gaze orientation system can be seen. Also, another loop is introduced from image ROI definition to the HMI component, loop which actually represents a direct feedback to the user of the robotic system. This concept of *human-in-the-loop* was also treated in [99]. As said before, for user interaction, different devices like chin control, speech recognition, BCI or a hand joystick can be used.

3.3. Architectural design

The image processing flow from Figure 3.6 is implemented as ROVIS operations that can be activated by the Sequencer (see Chapter 3.4), from the overall control architecture of the service robot. The Algorithms Executer is responsible for putting together the proper image processing methods within the operations.

Besides providing object class characteristics during task execution, the task planner within the Sequencer also operates on the basis of object classes and plans context-related operations. Among others, the following class-based categories of ROVIS operations can be activated by the Sequencer:

- *AcquireObjectBySCam*: Determine the object's location and size via the stereo camera (SCam) system. This operation is used to determine single objects (e.g. a handle), or containers where other objects are placed (e.g. fridge, table, gripper);
- *AcquireGrippedObjectBySCam*: Determines the gripping location and size of the object in the gripper via SCam;
- *AcquireObjectInContainerBySCam*: Determine location and size of an object in a container via SCam.

These ROVIS operations are used during execution of a task, but also within the *initial monitoring* process, which is performed in the Sequencer after task activation by the user. Initial monitoring is the procedure which organizes the supply of scenario-related object characteristics in the World Model according to the object anchoring principle [78]. This sets the basis for distinguishing between the handling of indefinite objects of a certain object class (e.g. *a bottle*) within the ROVIS operations or the handling of a definite

instantiation of a specific object class (e.g. *the small green bottle*). The difference between indefinite and definite objects is a runtime decision during anchoring, where a connection is established between symbolic object characteristics and concrete sub-symbolic data, like with the values *small* for the size and *green* and color of the bottle in this example.

Consequently, the pre-structured task knowledge, object classes and their characteristics allow to build universal operations in ROVIS as well as dynamic image processing chain construction according to a given scenario and context.

3.3.1. Software architecture

The overall UML software structure of ROVIS can be seen in Figure 3.9. The ROVIS CORE, which contains the Algorithms Executer, is the main part of the architecture. Using the object oriented programming concept of polymorphism [95], the set of image processing methods can be accessed by the Algorithms Executer through a pointer to the *CBaseAlgorithm* class from which all the vision methods are derived. Further, the Algorithms Executer dynamically constructs the image processing chain. In order to distinguish between normal functions and the operations required by the Sequencer, we will name the last ones *skills*. The servers providing such skills are called *skill servers*, such as the *ROVIS Skill Server*. The ROVIS skill server acts as an interface from the vision system to other components. Through it, the Sequencer calls the image processing operations made available by the skill server interface *IROVISSkillServer*. This server is only one of the skill servers used by the Sequencer in task planning. The ROVIS hardware is represented by the stereo camera and PTH unit accessed by the two hardware server interfaces: *ICameraHardwareServer* and *IPTHHardwareServer*.

The five types of image processing primitives from Figure 3.8 are implemented in five base classes derived from the common *CBaseAlgorithm* class, as seen in Figure 3.10. Here, an extra class is added for the camera calibration algorithms. The package *Feedback Structures* models the closed-loop control algorithms in two separated classes: *CClassicControl*, for the classical error-based control methods, and *CExtremumSeekingControl*, for control based on extremum searching. Two types of basic vision algorithm classes can be distinguished:

- traditional, open-loop, image processing methods;
- base classes of the algorithms which are using feedback structures.

The process of developing vision algorithms in ROVIS is depicted in Figure 3.11. The methods are developed and tested with the help of the *ROVIS Graphical User Interface* (GUI) from Figure 3.12. The GUI has a direct connection to the image processing methods, thus simplifying the development of the algorithms. No adaptation from the implementation and testing platform to the on-line execution is needed, that is, the vision algorithms are created and developed in only one place.

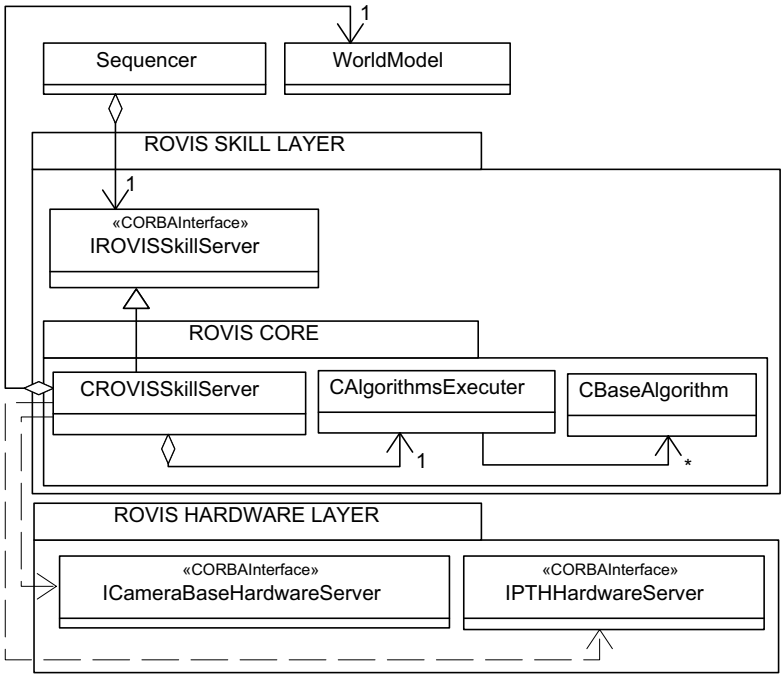


Figure 3.9.: ROVIS software architecture.

3.3.2. Execution of vision algorithms

In Fig. 3.13, the components involved in the execution of a typical skill are presented. In the center of the diagram is the ROVIS skill server interface which binds all the components together. For a better understanding of the process, a skill example, which provides as result the 3D position of an object, is considered: *AcquireObjectBySCam*. The input arguments of this skill are the task-related object names which are used to extract object-related information as provided during initial monitoring or previous executions. Based on this information, the dynamic generation of the image processing chain by the Algorithms Executer takes place. This process is started by the Sequencer via skill call. First, the necessary hardware is actuated. For the case of the vision system, the stereo camera field of view is changed and the stereo images pair acquired. Within the skill, the Algorithms Executer combines vision algorithms with the help of their basic properties which reside in the algorithms base class. These are the properties *Name*, *Description*, *Category*, *InputType* and *OutputType*. The above properties are used when selecting appropriate algorithms with respect to the given object class, as well as its a priori known characteristics. In case of recognition of definite objects, a priori known characteristics are concrete data sets (e.g. color, shape descriptors, etc.) that are used to parameterize the vision algorithms. The ROVIS methods are controlled via three functions called by the algorithms executer: *run*, *stop* and *pause*. These three functions modify the *Status* attribute of the algorithm. After the object of interest has been detected, its 3D position is saved in the World Model.

The overall structure of a skill, like *AcquireObjectBySCam*, can be seen in the flowchart

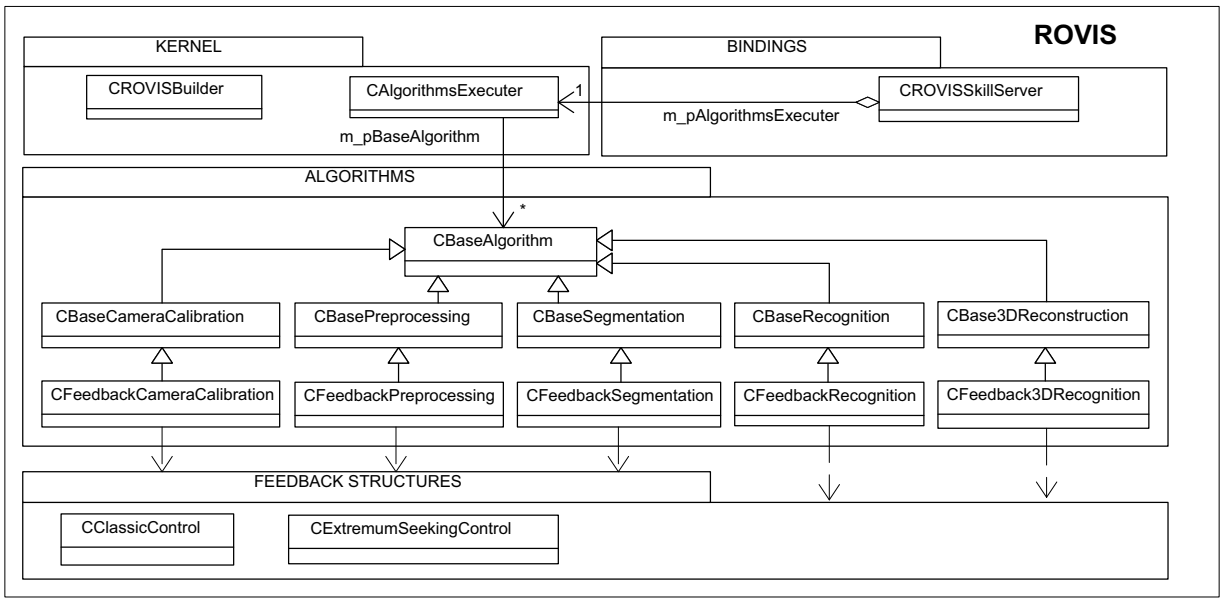


Figure 3.10.: Organization of the ROVIS architecture core.

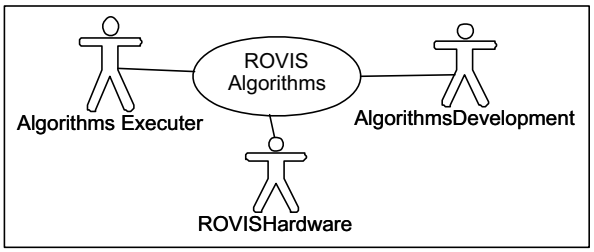


Figure 3.11.: ROVIS vision algorithms development process.

from Fig. 3.14. The start and end information messages of the skill are written in a log file for later debugging. Also, at the beginning, an extra process is started to check the incoming commands to the skill (e.g. stop or pause). This process is terminated at the end of the structure. The skill can be executed in two ways:

- *Normal execution*, which includes the actuation of the vision hardware and the construction of the image processing chain,
- *Simulative execution*, used by the Sequencer to test task planning capabilities.

Just before the end of the skill, the encountered exceptions are properly handled.

After defining the ROVIS architectural principle, the next task that has to be fulfilled is the integration of the proposed vision system within the overall control structure of the service robot, that is the FRIEND assistive robotic platform.

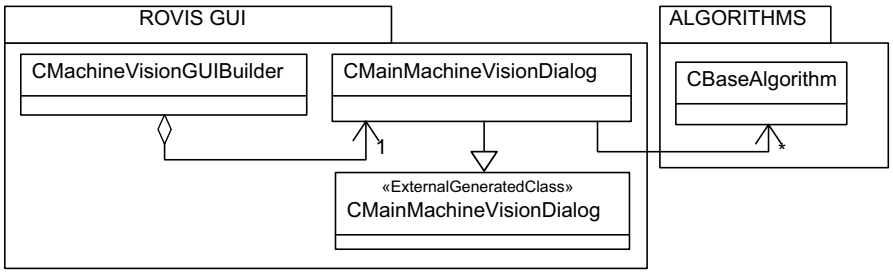


Figure 3.12.: ROVIS Graphical User Interface implementation and connection to vision algorithms.

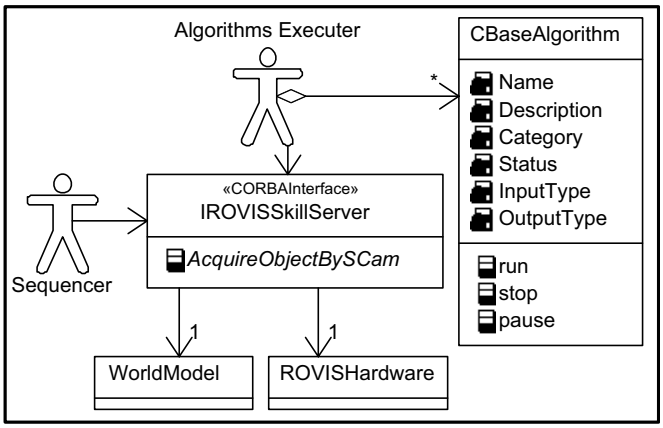


Figure 3.13.: Architectural design of a skill in ROVIS.

3.4. ROVIS integration in a service robotic system

The ROVIS architecture is used to implement the visual perceptual capabilities of FRIEND, a service robotic platform designed to assist disabled and elderly people in their daily life and professional life activities. The system, shown in Figure 3.15, is the result of more then a decade's work in the field of service robotics done at The Institute of Automation, University of Bremen. FRIEND is the 3rd generation of assistive robots designed at the institute, after FRIEND I [64] and FRIEND II [104], with their vision systems detailed in Chapter 3.1. The realization of the robotic system involved an interdisciplinary cooperation between different fields of research ranging from computer vision and robotics to neurorehabilitation.

The robotic system enables the disabled users (e.g. patients which are quadriplegic, have muscle diseases or serious paralysis due to strokes or other diseases with similar consequences for their all day living independence) to perform a large set of tasks in daily and professional life self-determined and without any help from other people like therapists or nursing staff.

The capabilities of FRIEND have been demonstrated in different scenarios where a large number of consecutive action sequences are performed. These sequences, necessary

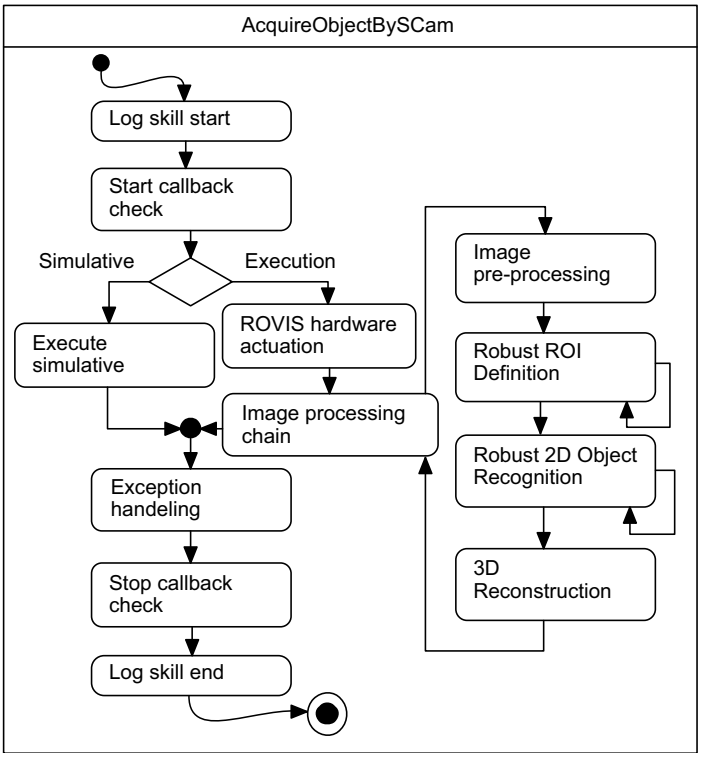


Figure 3.14.: Basic ROVIS skill structure.

to fulfil the demands of the robot system's user, are semantically described as robot object handling methods like "pour and serve a drink", "prepare and serve a meal", "fetch and handle a book". In order to plan such actions, reliable visual perception, given by ROVIS, is needed to determine the POSE of the objects in the FRIEND environment.

3.4.1. ROVIS hardware components in FRIEND

In FRIEND, the various hardware components that make up the system can be classified into four parts:

- *sensors*, required for environment understanding;
- *actuators*, performing actions requested by the user;
- *input-output devices*, needed for human-robot interaction;
- *computing system* where data processing and task planning takes place.

In this section the hardware components relevant to the vision system ROVIS will be discussed.

Stereo camera system: The main sensor component of FRIEND is the global vision module, represented by a Bumblebee® 2 stereo camera system [118] used for environment understanding. The camera is equipped with two 1/3" Sony® progressive scan CCD ICX204 sensors that provide two synchronized 1024x768px RGB color images at a

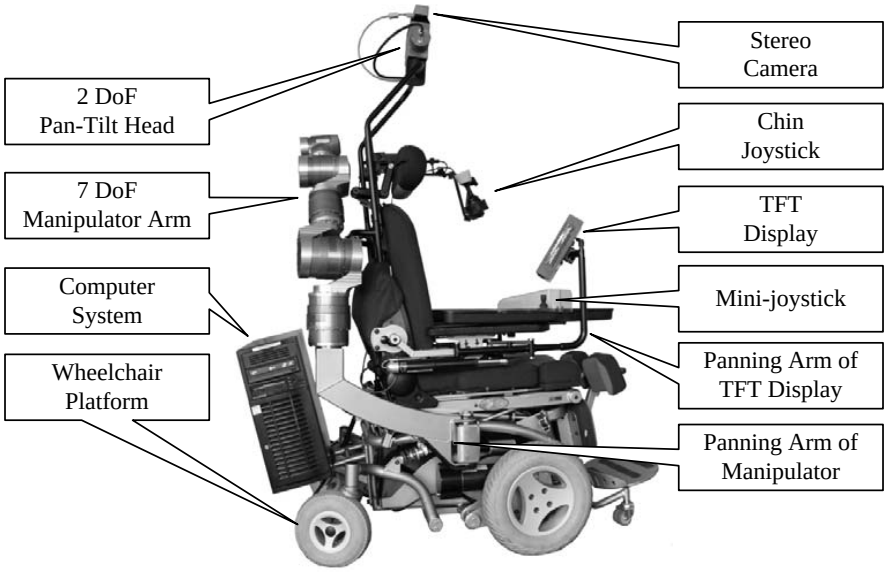


Figure 3.15.: 3rd generation of the assistive robot FRIEND.

maximum framerate of 20 *Frames Per Second* (FPS) and a $4.65\mu m$ square pixels. The imaging sensors have a focal length of 6mm with 43° *Horizontal Field Of View* (HFOV) and a distance of 120mm between the two lenses. Also, the Bumblebee® 2 camera is pre-calibrated against distortions and misalignment. The conversion from the analog image signal to digital images is done through a 12-bit *Analog to Digital Converter*. Serial communication between the camera and the computing system is implemented using a 6-pin IEEE-1394a FireWire interface. Various parameters of the stereo camera (e.g. exposure, white balance, shutter speed, etc.) can be set either to automatic or manual adjustment. The camera system is mounted on a PTH module behind the user, above its head, and views the scene in front of the robotic system including the manipulator and the tray which is mounted on the wheelchair in front of the user. The viewing angle of the camera can be changed by modifying the gaze orientation of the PTH unit.

On-line recalibration system: One major problem in the FRIEND system design is the shuddering, or vibration, of the upper part of the wheelchair during operation (manipulator arm and global vision system). This involves a change in the position of the camera with respect to the world coordinate system, found at the base of the manipulator arm. This change can produce grasping errors because of false 3D object reconstruction. In order to cope with this problem, an extra vision system that supervises the position of the manipulator's base with respect to the ground was added. The system consists of a Sony® monocamera and a visual pattern, or marker, mounted at the base of the robot arm. Within this coordinate system only the visual pattern will shudder, being mounted at the base of the robot arm. The camera is mounted on the lower part of the wheelchair, thus remaining at a constant parallel position with the ground. The tracked position of the marker is used on-line to recalibrate the global stereo camera system.

Camera pan-tilt gaze orientation module: An important actuator used by ROVIS is the gaze orientation module of the global vision system. This module is composed of a Schunk® 2-DoF *Power Cube* servo-electric PTH unit. The covered field of view of the PTH is 1180° and 180° in the pan and tilt directions, respectively. For positioning and velocity control it uses two incremental encoders with a resolution of 2000Inc/Rotation. The communication between the PTH and the main computing device is performed via a CAN bus interface.

Processing system: The computing system is represented by a standard PC computer with 8GB of RAM and two Intel XEON® QuadCores microprocessors, each working at a speed of 2.33GHz. The high computing power has been chosen so in order for the system to cope with the large amount of information data that has to be processed, especially from the vision system and motion planning algorithms of the manipulator. The computer is mounted at the backside of the wheelchair, behind the user, as can be seen from Fig. 3.15.

3.4.2. Support scenarios

The capabilities of the FRIEND robot are materialized into three support scenarios. From these scenarios, one deals with activities of daily living and the remaining two with re-integration of the user into working environments, as described bellow.

ADL – Activities of Daily Living: The ADL scenario enables the user to prepare and serve meals or beverages. It represents the types of activities that a person performs in a domestic environment. Besides the robot FRIEND, the elements included here are typical household objects like refrigerator, microwave oven, bottles, glasses or mealtrays. The task of the ROVIS architecture is to reliably recognize these typical household objects for proper path planning of the manipulator arm and appropriate object grasping. For manipulation reasons during eating, a special mealtray and spoon were designed.

Working at a library service desk: A second support scenario developed for FRIEND is a professional life scenario where the user is working at a library desk equipped with a laser scanner for reading IDs of books and customer IDs. The task of the FRIEND user is to handle outgoing and returned books, as well as other tasks at a library desk. A positive aspect of the library scenario is that the user has to interact with people, thus making his recovery and reintegration in professional life easier. In order to successfully achieve the required tasks, that is books handling, their locations have to be precisely calculated. Taking into account the variety of books (e.g. different textures, colors and sizes), the image processing algorithms behind books recognition can rely on no a priori knowledge except their rectangular shape. Also, the proposed vision system has to recognize the library desk and its components (e.g. laser scanner for reading the ID of the grasped book).

Functional check of workpieces: The third support scenario takes place in a rehabilitation workshop. Here, the user has to perform quality control tasks. These tasks have been proven to positively influence the disabled person in the rehabilitation process. Such

a task is checking of electronic keypads for public phones for malfunctioning. For this purpose a special workshop desk containing different smart tools has been built. The keypads are placed into a keypad magazine from which the user can extract only one at a time by pushing a button which will eject the keypad. The vision task is to detect the 3D position of the electronic keypads and the workshop desk on which the keypads are mounted. When a keypad is localized, the manipulator can grasp it and move it in front of the user in order to allow him to perform a visual check. After visual check, the keypad will be inserted into a special test adapter for verifying its functionality.

In this thesis, the main focus is to reliably determine the POSE of the objects to be manipulated in the FRIEND scenarios, that is meal-trays, bottles, glasses, books, etc.

3.4.3. Overall robot control architecture

The robust control of a complex robotic platform like FRIEND can only be achieved with an appropriate control architecture which separates the different levels of processing (e.g. image processing, manipulator control, task planning etc.) into abstraction layers linked together by a core module which acts as a system manager. The software architecture used for controlling the FRIEND robot, presented in Figure 3.16, is entitled MASSiVE and it represents a distributed control architecture which combines reactive behavior with classical artificial intelligence based task planning capabilities [63, 61, 62]. MASSiVE is modeled under the *shared control framework*, signifying a constant interconnection with the user of the robot. Such approaches, where the cognitive capabilities of the user are used to improve the capabilities of the robot, have also been encountered in other architectures [99, 23].

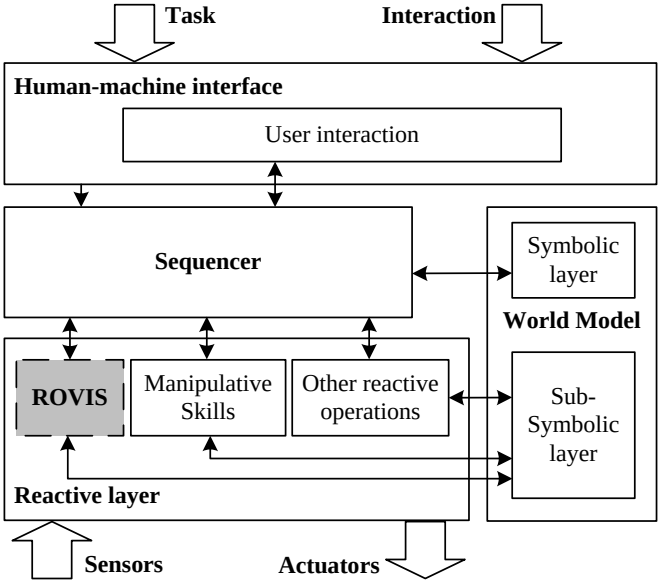


Figure 3.16.: MASSiVE overall control architecture of the service robot FRIEND.

The MASSiVE architecture interacts with the user through the HMI which operates

at user interaction level. The HMI is interconnected with the Sequencer, the core of MASSiVE. The Sequencer plays the role of a *Discrete Event Controller* (DEC) that plans sequences of operations by means of predefined tasks knowledge [63]. The user commands are acquired with the help of different input methods, such as speech recognition, chin control, or BCI [60, 101], and translated further into machine language for interpretation [80]. The processing algorithms that converts a user request into robot actions resides in the Reactive Layer. Here, the data collected from different sensors, such as the stereo camera, are processed in order to "understand the environment". The data is further converted into actions by the available actuators, such as the 7-DoF manipulator. As said before, the sequence of operations needed to perform a specific task is generated by the Sequencer module which also calls the ROVIS vision methods. Throughout the functioning of the system, the computed data is shared between the modules with the help of the World Model. In MASSiVE, the World Model defines the information produced and consumed by the operations in the Reactive Layer. The software interconnection between the processing layers is implemented using the *Common Object Request Broker Architecture* (CORBA) [102]. During on-line system operation task parameters can be viewed with the help of a GUI available on a display system mounted on the wheelchair in front of the user. The vision system acts on the commands sent by the Sequencer and performs appropriate tasks needed for reliable object recognition and subsequent 3D reconstruction of the object to be manipulated.

ROVIS is integrated as a reactive module within MASSiVE. As displayed in Figure 3.16, ROVIS is placed inside the Reactive Layer from where it provides visual information for the Sequencer which further activates the manipulative skills. ROVIS communicates with the Sub-Symbolic layer of the World Model, where it outputs the reconstructed 3D environment. This information is used further by the manipulative operations for path planning and object grasping [73]. On the other hand, for performing the reconstruction task, ROVIS uses from the World Model necessary information such as features of an object class needed for object classification.

3.4.4. Functional analysis of workflow

The *system functional analysis* represents the verification and validation of the developed vision system. For this purpose, a message-driven approach involving sequence diagrams is used in the analysis.

In UML language, sequence diagrams are used to graphically represent the functional flow of information and the behavior of a system. In Figure 3.17, a simplified sequence diagram of the behavior of ROVIS is shown. This behavior is encountered when during object recognition and reconstruction. The user of FRIEND starts the process by selecting a specific support scenario. After, the control is taken by the Sequencer who plans the necessary sequence of actions needed to fulfill the requested user scenario. After the list of necessary robotic actions is generated, the control is further given to the ROVIS architecture for 2D objects recognition and 3D reconstruction. As said before, the first

step in the ROVIS image processing chain is the definition of the image ROI. This is modeled in Figure 3.17 through *user interaction* and *camera gaze control*.

For the case of user interaction, the control of object recognition is given to the user of FRIEND who defines an interest point on the input left camera image, as seen in Figure 5.2. The algorithm for defining the ROI through user interaction is detailed in Chapter 5.2. For the second case, *camera gaze orientation*, the control is given to the 2-DoF PTH unit for changing the Field Of View (FOV) of the stereo camera system. The FOV change is sequenced by the calculation of the image ROI. In Chapter 5.3 two algorithms for image ROI definition using camera reorientation are detailed. After the image ROI is defined, the object recognition methods are applied and the 3D positions of the objects of interest are calculated and saved in the World Model. Finally, the control of the process is goes back to the Sequencer component which further calls the manipulative skills of the 7-DoF manipulator arm.

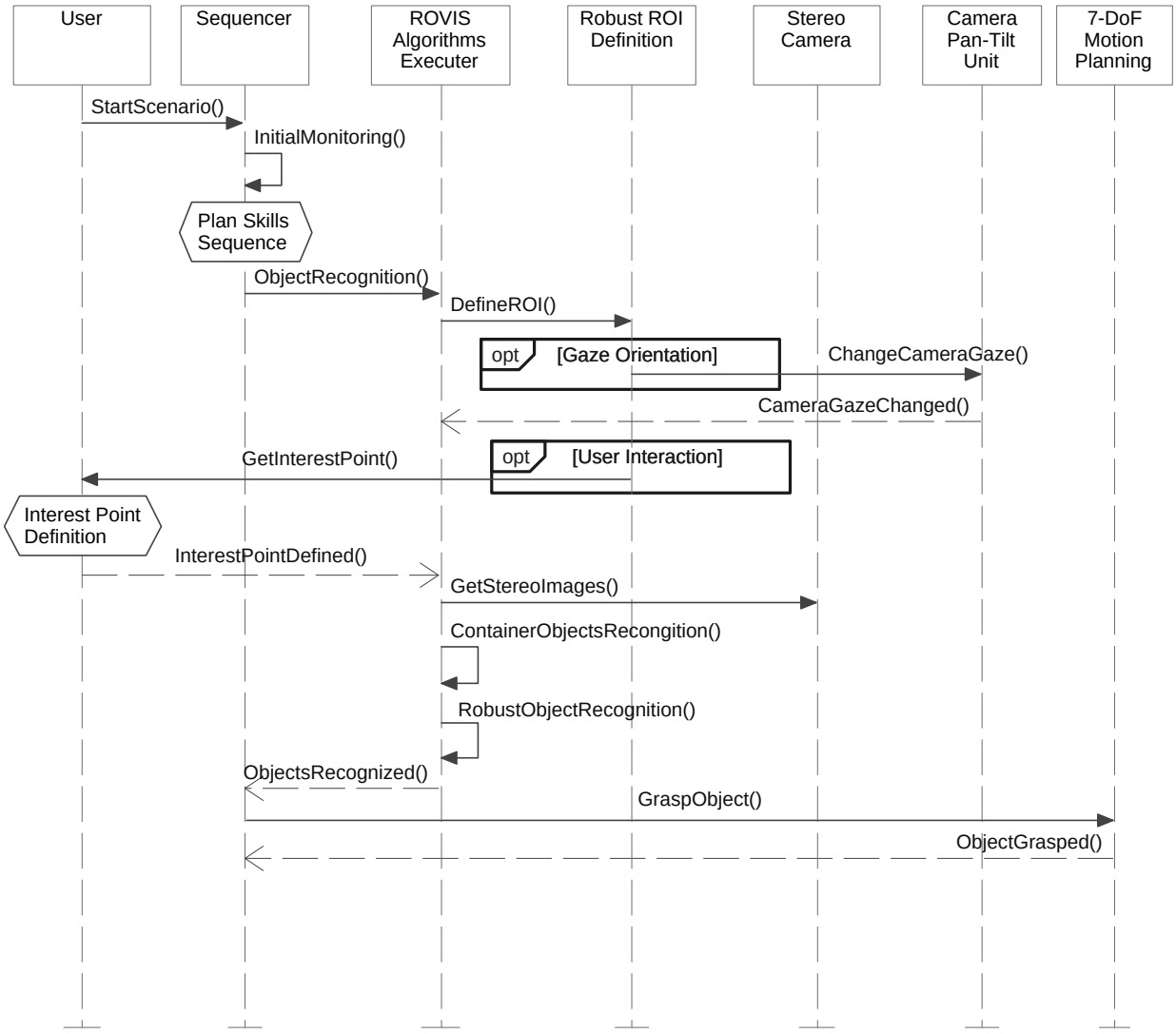


Figure 3.17.: Sequence diagram of ROVIS operations involved in environment understanding.

4. Robust image segmentation for robot vision

One crucial step in the image processing chain from Figure 2.1 is image segmentation. Its purpose is to reduce the visual information from the input image in order to make it suitable for further processing. As seen in Figure 1.1, the results of image segmentation directly influences the performance of object classification and the precision of 3D object reconstruction. Hence, reliability of image segmentation is a key requirement in robot vision applications.

Motivation for robust segmentation in robot vision

In this thesis, image segmentation is used to classify and extract 2D object features in order to reconstruct the 3D Position and Orientation (POSE) of an object. An example emphasizing the importance of reliability of image segmentation is shown in Figure 4.1, where the "3D object reconstruction problem" is represented. In Figure 4.1(a), a scene from the Activities of Daily Living (ADL) scenario of the FRIEND robotic system has been imaged under artificial and daylight illumination conditions, respectively. The artificial illuminated scene will be referred to as the reference scene. From this scene, the constant values of the object thresholding interval C_l (see Equation 2.19) are calculated. For the sake of clarity, only the left stereo image is shown, original and segmented, respectively. Figure 4.1(b) represents the segmentation results obtained using Equation 2.21. The thresholding interval $C_l = [35, 65]$ has been determined by manually segmenting the reference image. As can be seen, in case of reference artificial illumination, the chosen C_l interval performs well, but in the second case of daylight illumination, the constant value of C_l outputs a false segmentation result. This happens because colors varies with respect to illumination. Although the object can be recognized in both images, reliable extraction of object feature points, needed for 3D reconstruction, can be made only on a well segmented image. Hence, in Figure 4.1(c), only feature points obtained from good segmentation provide an optimal 3D reconstruction. In case of erroneous segmentation the deviation of the feature points from the ideal values, represented in Figure 4.1 by the object's center of mass, corresponds to a deviation in the 3D reconstructed object.

In robot vision, the above example suggests also the importance of quality of image segmentation over object classification power. In the example, although the object can be easily classified using powerful state of the art machine learning methods, like *Neural Networks* (NN) or *Support Vector Machines* (SVN) [43], its 3D position can not be calculated precisely, that is, the algorithms concentrate on the recognition of objects in 2D images without taking into account the need of good object segmentation used for

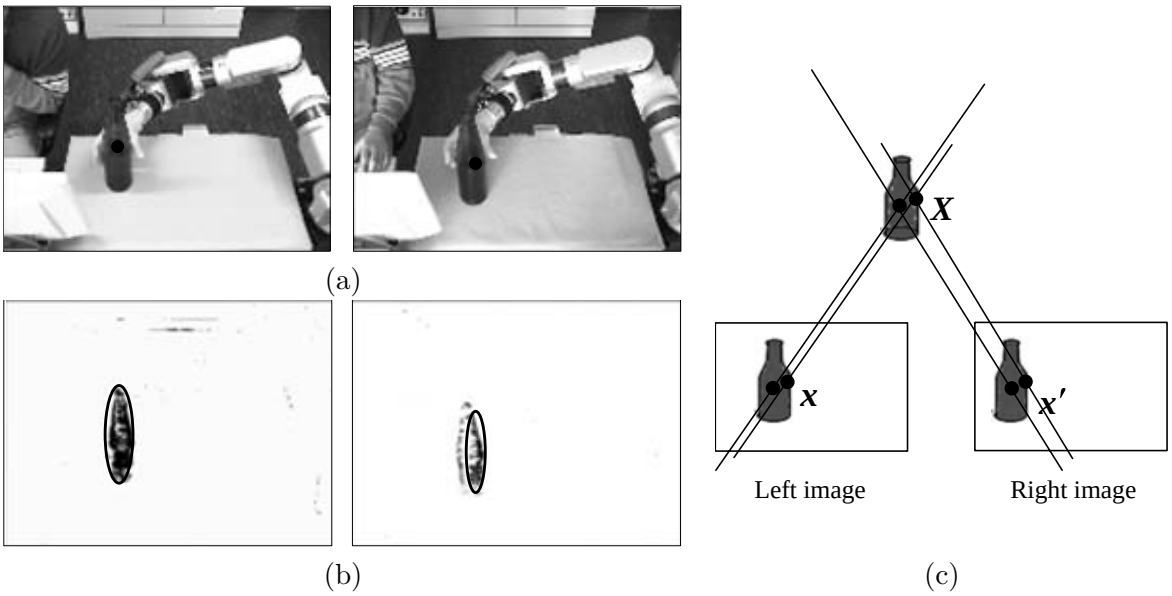


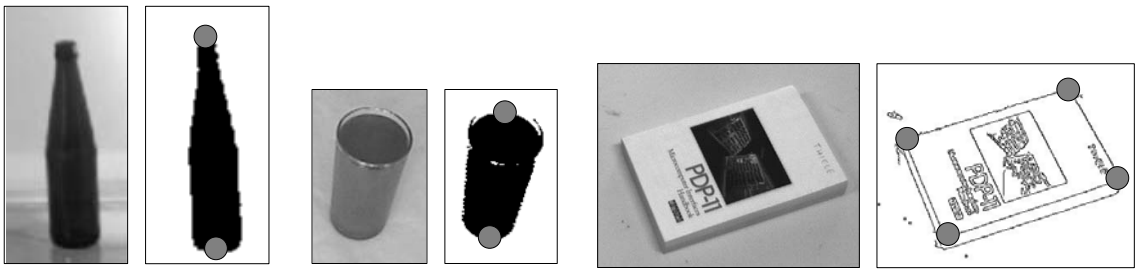
Figure 4.1.: 3D object reconstruction principle. (a) Images acquired under artificial and daylight illumination conditions, respectively. (b) Color segmentation and feature point extraction using the thresholding interval $C_l = [35, 65]$. (c) 3D reconstruction of the segmented objects in (b), respectively.

subsequent object feature point extraction and 3D reconstruction [18].

For robotic object manipulation, proper segmentation is also needed for the extraction of the geometrical shape of the imaged object. The shape is used for calculating the optimal object grasping point required by the manipulator arm to plan its movement. Depending on the characteristics of the object, different segmentation methods are better suited to extract its shape. In Figure 4.2, the segmentation and feature points extraction of three objects is illustrated. For uniformly colored items, the segmentation is performed by grouping together pixels based on their similarity, as for the bottle and glass from Figure 4.2(a,b). The segmentation output represents blobs of pixels which separates the object from the background, as explained in Chapter 2.4. If the imaged items are textured, segmentation based on detecting their boundaries is a better choice. Such methods evaluate sharp transitions between neighboring pixels in order to emphasize the object's edges, as seen in Figure 4.2(c) for the case of a book.

Also, the choice of feature points extraction is strictly related to the nature of the images item. For example, the optimal feature points of a bottle or a glass are represented by their top and bottom, as seen in Figure 4.2(a,b). On the other hand, the feature points of a book are represented by its four corners, as illustrated in Figure 4.2(c).

In this chapter, two techniques for robust image segmentation in robot vision are proposed. The goal is to cope with variable illumination conditions and scene uncertainty. The main idea of the approach is to use classical image processing techniques enhanced by including feedback control at low-level image processing, an idea also tackled previously in the computer vision community [66, 76, 70]. In contrast to these publications, this



(a) (b) (c)

Figure 4.2.: Feature points extraction based on different segmentation types.

thesis approaches the inclusion of feedback control in image processing from the point of view of robotic manipulation, where precise 3D reconstruction is crucial for optimal object grasping. The first method, discussed in Chapter 4.1, is intended to control the parameters of region based segmentation using quality information measured from the obtained binary image. The second method, presented in Chapter 4.2, involves adaptation of parameters of boundary segmentation with the purpose of reliably extracting boundary object features. The robust segmentation algorithms proposed in this chapter will be used extensively in the rest of the thesis for implementing the visual perceptual capabilities of the FRIEND robot.

4.1. Robust region based segmentation

The color based segmentation algorithm presented in this chapter aims at detecting "unknown" uniformly colored objects in variable illumination conditions. The term unknown denotes the fact that no a priori knowledge regarding object characteristics (e.g. color, shape etc.) is used to drive the parameters of image segmentation. This comes from the fact that the robot operates in complex, cluttered, scenes. The algorithms presented here will be used in Chapter 5 for defining an object's ROI and also for recognition of objects of interest in Chapter 6.

4.1.1. Evaluation of color information

Since the goal of the algorithm presented here is to segment uniformly colored objects, it makes sense firstly to investigate the nature of color representation. As discussed in Chapter 2.2, although color images are usually stored under the RGB color model, this representation is inappropriate for color object recognition, since it contains color information in all its three channels. In contrast to the RGB representation, a number of color models have been introduced with the purpose of separating the color, or *chromatic*, information from the intensity, or *achromatic*, information. One such model is the HSI color space presented in Chapter 2.3. In this model, a pixel is characterized by color (represented by its hue and saturation) and intensity.

If a closer look is taken at the color cone from Figure 2.3(b), it can be observed that saturation is defined as the radial distance from the central axis and takes values between 0 at the center and 255 at the extremities. The value 255 represents the maximum value of usually used in computer implementations, where gray level pixel information is stored in an 8bit representation. In other words, saturation represents the purity of a color. A color is defined by its hue, where hue is an angle with values in the interval $[0, 360]$. For $S = 0$, the color information is undefined, that is, color will be represented as shades of gray ranging, from black to white as one moves from 0 to 255 along the intensity axis I. On the other hand, if saturation is varied from 0 to 255 the perceived color changes from a shade of gray to the most pure color represented by its hue. This phenomenon can be seen in Figure 4.3 for the case of the red color ($H = 0$). In Figure 4.3, both the saturation and intensity values are varied in the interval $[0, 255]$. When saturation is near, 0 all pixels, even those with different hues, look similar. As saturation increases, they get separated by their color values. In the human visual system this phenomenon can be encountered when we look at objects under poor illumination conditions (e.g. a red glass becomes gray when illuminated only by the moon's light). Also, even if saturation is high, a color is close to a gray value if the intensity of the image is low, as can be seen in Figure 4.3.

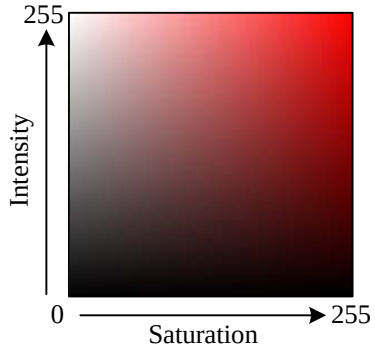


Figure 4.3.: Saturation (S) and intensity (I) variation in the interval $[0, 255]$ for the case of $H = 0$ (red color).

From the above description, the relationships between the hue, saturation and intensity components of an image can be described by the following statements:

- low saturation S decreases the hue H value (if $S = 0$ then the value of H gets undefined: $S = 0 \rightarrow H = \varnothing$);
- low intensity I decreases the hue H and saturation S values (if $I = 0$ then both H and S values get undefined: $I = 0 \rightarrow H = \varnothing, S = \varnothing$).

Usually, lower values of saturation and intensity are encountered in images acquired under poor illumination conditions. Following the above reasoning, *color can be used in a segmentation method only when the values of saturation and intensity are high enough.* Thus, for lower values of S and I a segmentation method based only on intensity values

should be applied. The automatic switching between the two methods is realized based on a switching mechanism which uses the S and I values to evaluate the amount of color information present in the image:

$$H_{ev} = (1 - K_{ev})\frac{S}{S_{max}} + K_{ev}\frac{I}{I_{max}}, \tag{4.1}$$

where S_{max} and I_{max} represent the maximum value of saturation and intensity, respectively. In most computer implementations $S_{max} = 255$ and $I_{max} = 255$. The color evaluation parameter varies in the interval $H_{ev} \in [0, 1]$. *The higher the color information in an image is, the higher the evaluation parameter H_{ev} is.* The coefficient $K_{ev} \in [0, 1]$ signifies a scaling factor needed to enforce the importance of each component in the relation. There are cases when, although a scene is good illuminated, it contains achromatic objects represented by gray surfaces. It has been discovered that for images with such objects, the value of saturation decreases, whereas the intensity stays high. Keeping in mind this fact, the contribution of saturation in Equation 4.1 should be higher than the one of intensity. For this reason, the value of the scaling factor K_{ev} has been heuristically set to $K_{ev} = 0.32$. The switching between intensity and color segmentation is made according to the switching threshold T_{sw} as:

$$\begin{cases} \text{Intensity segmentation if } H_{ev} < T_{sw} \\ \text{Color segmentation if } H_{ev} \geq T_{sw} \end{cases} \tag{4.2}$$

After a number of trial and error experiments, it has been established that $T_{sw} = 0.3$. Color based segmentation is also referred to as hue-saturation segmentation, since color is stored in these image planes.

The switching between intensity and color segmentation is graphically shown in Figure 4.4. Based on the value of H_{ev} either one of the methods may be called. In the following sections, the two segmentation algorithms, intensity and color based, will be discussed. Since intensity segmentation requires only one gray level plane, it is simpler than color segmentation, which needs both the hue and saturation components. For this reason, the proposed closed-loop intensity segmentation methods will be explained first.

4.1.2. Closed-loop intensity segmentation

The goal of robust intensity segmentation is to segment achromatic objects or images acquired from poor illuminated scenes. One major drawback of this approach, in comparison to color segmentation, is that objects are classified based only on a relative low number of shades of gray. Although, as said before, in many cases when illumination is poor, intensity based segmentation is the only valid approach.

The approach used in this thesis to design a robust method for intensity segmentation, which also sets the guidelines for the robust color based method, is to control the double thresholding operation from Equation 2.13 in such a way that no a priori information

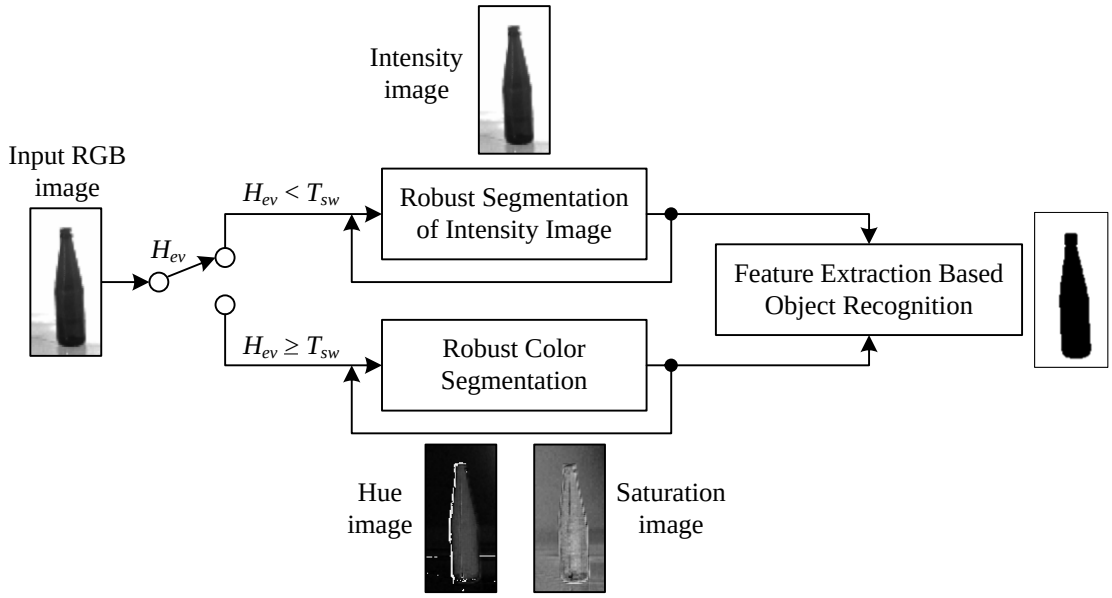


Figure 4.4.: Switching between intensity and hue-saturation segmentation.

regarding the gray level values of the object of interest is required. The reason not to use any a priori information lies on the fact that gray level values vary with respect to illumination changes, hence, although the method would work fine on predefined, reference, illumination contitions, it would fail to produce a reliable result when illumination changes. The principle of closed-loop segmentation, introduced in [83], can be seen in Figure 4.5 applied to the double thresholding operation. Based on an input image and on initial thresholding parameters, the segmentation result is analyzed and further, in a closed-loop manner, automatically adapted to the optimal working point, that is to the *optimal segmentation result*. In such a feedback control system, the control signal, or actuator variable, is a parameter of image segmentation and the controlled variable a measure of feature extraction quality.

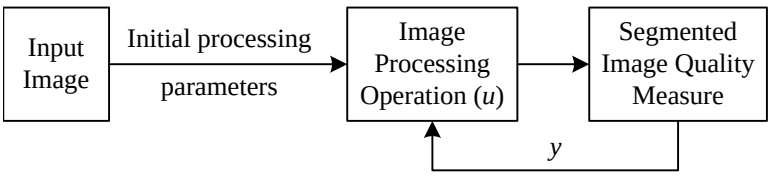
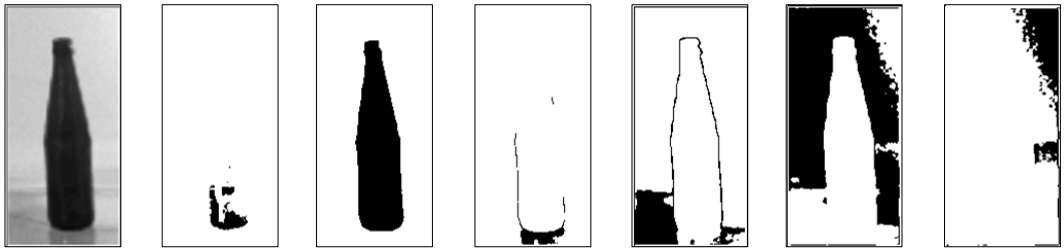


Figure 4.5.: Closed-loop intensity based segmentation principle.

The motivation for adjusting the threshold operator 2.13 is exemplified in Figure 4.6, where different results of segmentation of a bottle are shown. As evident, only the correct choice of object thresholding interval $[T_{min}, T_{max}] = [27, 52]$ yields a segmented image of good quality, containing a whole, well segmented object. In contrast, an incorrect choice of the thresholding interval causes segmentation failure. As shown in Figure 4.6, other intervals, which lay outside the interval of the object's pixel values, yield images with seg-



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

Figure 4.6.: Image segmentation corresponding to different thresholding intervals. (a) Gray level input image. (b) $[T_{min}, T_{max}] = [0, 25]$, (c) $[T_{min}, T_{max}] = [27, 52]$, (d) $[T_{min}, T_{max}] = [60, 85]$, (e) $[T_{min}, T_{max}] = [165, 190]$, (f) $[T_{min}, T_{max}] = [195, 220]$, (g) $[T_{min}, T_{max}] = [225, 250]$.

mented background pixels (noise) and without any segmented object pixels, respectively.

As explained in Chapter 2.1, the process of designing a control system for image processing differs significantly from classical control application, but still exhibits similar design phases like process studying, decision on controlled and actuator variables, design of the control configuration and of the controller and finally testing. These phases will be further detailed for the case of robust intensity segmentation.

Choice of the actuator variable

Taking into account the observations made from Figure 4.6, the purpose of the control system is to control the thresholding operation 2.13. A suitable actuator variable for this process is an increment value u_i added to the initial intensity thresholding interval $[T_{min}, T_{max}]$ in order to drive the segmentation operation to its optimal result. For maintaining a lower system complexity, the values of the thresholding interval have been linked together as:

$$T_{max} = T_{min} + K_{iw}, \tag{4.3}$$

where K_{iw} is a constant value denoting the *intensity thresholding interval width*. Having in mind Equation 4.3, the expression of the actuator variable may be now written as:

$$[T_{min} + u_i, T_{max} + u_i], \tag{4.4}$$

where u_i represents the threshold increment to the initial thresholding value $[T_{min}, T_{max}]$. In order to maintain a consistence of the algorithm description, Equation 4.4 can be rewritten as:

$$[T_{min} + u_i, (T_{min} + K_{iw}) + u_i]. \tag{4.5}$$

By combining the two values of the thresholding interval using Equation 4.3, a singular actuator variable, u_i, is obtained for the proposed segmentation control system.

Choice of the controlled variable

In order to automatically adjust the actuator variable so that the current quality of segmented image is driven to the desired, or optimal, value a controlled variable has to be defined. The chosen controlled variable has to be appropriate from the control as well as from the image processing point of view [83]. From the image processing point of view, a feedback variable must be an appropriate measure of image segmentation quality. Two basic requirements for control are that it should be possible to calculate the chosen quality measure easily from the image and the closed-loop should satisfy input-output controllability conditions. Input-output controllability primarily means that for the selected output (controlled variable) an input (actuator variable) which has a significant effect on it must exist.

Reliable object recognition and 3D reconstruction can only be achieved with a segmented image of good quality. A binary segmented image is said to be of good quality if it contains all pixels of the object of interest forming a "full" (unbroken) and well shaped segmented object region. Bearing in mind the qualitative definition of a segmented image of good quality given above, the quantitative measure of segmented image quality in Equation 4.6 has been proposed:

$$I_m = -log_2 p_8, \; I(0) = 0, \tag{4.6}$$

where p_8 is the relative frequency, that is, the estimate of the probability of a segmented pixel surrounded with 8 segmented pixels in its 8-pixel neighborhood:

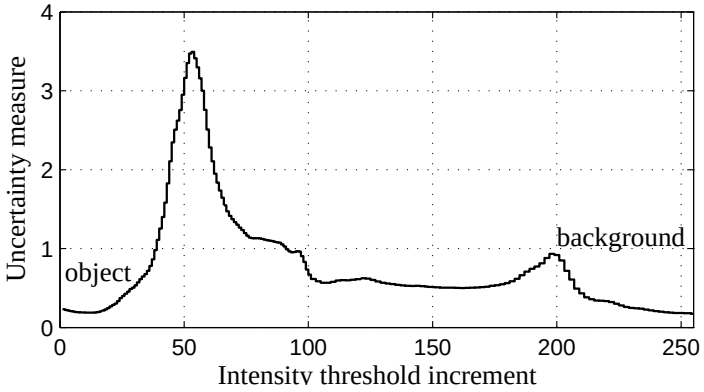
$$p_8 = \frac{number\ of\ segmented\ pixels\ surrounded\ with\ 8\ segmented\ pixels}{total\ number\ of\ segmented\ pixels\ in\ the\ image}. \tag{4.7}$$

Keeping in mind that a well segmented image contains a "full" (without holes) segmented object region, it is evident from Equation 4.7 that a small probability p_8 corresponds to a large disorder in a binary segmented image. In this case, a large uncertainty I_m, defined by Equation 4.6, is assigned to the segmented image. Therefore, the goal is to achieve a segmented image having an uncertainty measure I_m as small as possible in order to get reliable segmentation result.

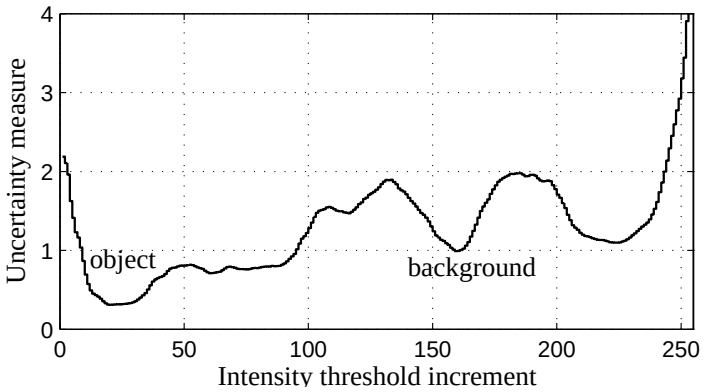
Input-output controllability

In order to investigate the input-output controllability of the image segmentation system when considering the thresholding interval increment u_i as the input (actuator) variable and the proposed uncertainty measure I_m as the output (controlled) variable, the thresholding of the image from Figure 4.6(a) was done. The initial thresholding interval

was set to $[0, K_{iw}]$, where $K_{iw} = 20$. To this interval the increment u_i was added as in Equation 4.5. For each segmented image corresponding to the increment $u_i \in [0, 255]$, the uncertainty measure I_m was calculated. The resulting input-output characteristic is presented in Figure 4.7 for two different input intensity images. As can be seen, the uncertainty I_m is sensitive to the chosen actuator variable across its effective operating range. Also, it is clear that each input value is mapped to at most one output value and that it is possible to achieve the minimum of I_m, which corresponds to the segmented object image of reference good quality, by changing the thresholding boundaries. The satisfaction of these prerequisites for successful control action to be performed demonstrates the pair "threshold increment u_i – uncertainty measure I_m" as a good "actuator variable – controlled variable" pair.



(a)



(b)

Figure 4.7.: The uncertainty measure I_m of segmented pixels vs. intensity threshold increment u_i for two different input intensity images.

The input-output characteristics shown in Figure 4.7 is characterized by a number of local minimas from which only one represents the desired object of interest. As an example, in the diagram, the minimum corresponding to the object and the one representing the background have been displayed. Based on the above discussion, it can be said that the original problem, that of finding the optimal object threshold interval that provides

a segmented object image of good quality, appropriate for subsequent object feature extraction, can be interpreted and converted to the problem of finding the minimas of the uncertainty I_m of the object region in the binary segmented image. Given the proper effective operating range $[u_{low}, u_{high}]$, with $u_{low}, u_{high} \in [0, 255]$, the optimal threshold increment $u_{i\ opt}$ can be expressed as:

$$u_{i\ opt} = \arg\min I_m(u_i). \tag{4.8}$$

In Chapter 6.1, characteristics like the ones in Figure 4.7 will be used for extracting different objects present in the imaged scene of the FRIEND robot.

Control structure design

The minimum corresponding to optimal segmentation is calculated using the extremum seeking method presented in Table 2.1. The input image from which the curve in Figure 4.7 has been generated contains only one object. The diagram has two local minimas, representing the object and the background, respectively. Keeping this in mind, it is important to choose appropriate effective input operating ranges $[u_{low}, u_{high}]$. Because of noise in the input data and also for not getting stuck in local minimas, the feedback optimization method is not applied directly on the calculated characteristic, but on a curve smoothed using a moving average filter [45]:

$$C_j = \frac{\sum_{i=-(m-1)/2}^{i=(m-1)/2} X_{j+i}}{m}, \tag{4.9}$$

where C is the smoothed characteristic, X is the input data, j is the index into the input data and i is the index into the sliding window m.

The reference value of the chosen controlled variable is not explicitly known in the presented system. However, the selection of an image quality measure whose minimal value corresponds to the image of good quality has been suggested for the controlled variable. Hence, the optimal value of the chosen controlled variable is achieved through feedback optimization using the extremum seeking algorithm from Table 2.1, as shown in Figure 4.8. Here, in principle, the feedback information on the segmented image quality is used to choose the optimal value $u_{i\ opt}$ of the actuator variable u_i, that is, to drive the current segmented image to one with reference optimal quality.

4.1.3. Closed-loop color segmentation

The main disadvantage in using only intensity information for segmentation is the relatively low number of shades of gray that can be used in distinguishing between different objects and their background. Hence, it may be impossible to differentiate between two

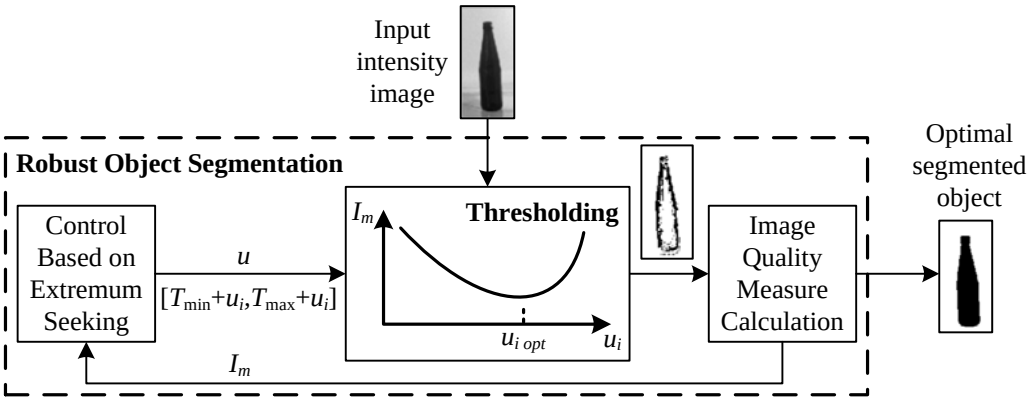


Figure 4.8.: Closed-loop intensity segmentation control structure.

objects even when they have different colors. This phenomenon occurs when intensity is the same for different objects. In order to overcome this problem, color information may be used when its quantity is enough to reliably distinguish between colored objects, namely $H_{ev} \geq T_{sw}$.

The principle of color segmentation in the HSI color space is similar to intensity based segmentation, with the difference that, instead of using one gray level plane, two planes are used to generate the binary segmented image. In the following, a closed-loop color based segmentation method, derived from the principle of robust intensity segmentation presented above, is proposed. The use of color in the algorithm denotes the information extracted from images and not a priori color class information, classically used when segmenting colored objects.

Choice of the actuator and controlled variables

As before, the first step in developing a closed-loop image processing algorithm is the selection of the actuator-controlled variables pair. Since in color segmentation two gray level planes are used, the complexity of the control system increases with the factor two.

If we take a look at Figure 2.3(b), where the HSI color model is represented, it can be seen that color depends on the hue H and saturation S components. Bearing in mind that for intensity segmentation the intensity information I was used in calculating the output binary segmented image, in the current case both H and S are involved in the segmentation process.

The hue color circle has been depicted separately in Figure 4.9. The color class C_l of an object is determined by the angle H from the reference red axis of the color circle. Since a real object might contain pixels with different color values it is more evident to define an object color class as a set of more hue values. On the color circle from Figure 4.9, an object color class C_l will be defined as the interval:

$$C_l = [T_{min}, T_{max}], \tag{4.10}$$

or:

$$C_l = [T_{min}, (T_{min} + K_{hw})], \tag{4.11}$$

where K_{hw} is a constant angular value representing the width of the object color class interval. In this thesis $K_{hw} = 60$. $[T_{min}, T_{max}]$ is the thresholding interval of the hue image plane. The goal of the control system for this case is to control the position of the H angle in order to obtain reliable object segmentation in a large spectrum of illumination conditions. Since object color values vary with illumination, no a priori knowledge of the object of interest color class could be used.

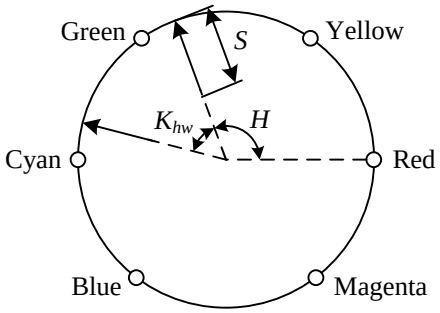


Figure 4.9.: Definition of an object color class on the hue-saturation components.

As discussed in Chapter 2.3, the saturation component S is involved in color based segmentation as a mask that rejects the pixels which carry less color information. In traditional color segmentation, the output color segmented image is calculated as a logical **AND** operation between the hue and saturation segmented images, as pointed out in Equation 2.21. The problem with this traditional approach is that the processing of the two image planes is done in an open-loop manner and no information regarding the quality of the binary segmented image is used in improving the segmentation. For obtaining a reliable segmentation, the length of the S component is controlled, in conjunction with controlling the H angle. The H angle gives the object color class C_l. S is varied from the radius of the color circle, 255, to its center. In principle, the goal of the control system is to determine the values of hue and saturation that provides optimal image segmentation. This process involves two steps:

- *determine the optimal color class C_l;*
- *for the calculated color class C_l determine the optimal saturation thresholding interval.*

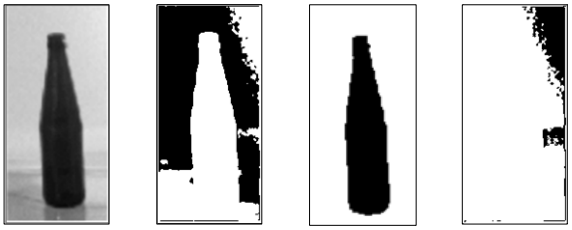
The actuator variables involved in the color based segmentation process are similar to the one in Relation 4.5, with the difference that here we have two variables, one for hue and one for saturation.

For determining the optimal color class, the hue angle H has to be automatically adjusted to its optimal value. Similar to intensity based segmentation, this is achieved by

using an actuator variable defined as the increment value u_h added to the initial object thresholding interval:

$$[T_{min} + u_h, (T_{min} + K_{hw}) + u_h], \tag{4.12}$$

where u_h is an increment value added to the initial color interval $[T_{min}, T_{max}]$. In Figure 4.10, segmentation results for different object color intervals can be seen. The value of the saturation thresholding interval was manually set to the optimal value of $[74, 255]$. As can be seen, only one object color interval corresponds to the segmentation result of good quality.



(a) (b) (c) (d)

Figure 4.10.: Color segmentation results for different object color intervals and constant optimal saturation segmentation. (a) Input image. (b) $[T_{min}, T_{max}] = [21, 81]$. (c) $[T_{min}, T_{max}] = [104, 164]$. (d) $[T_{min}, T_{max}] = [258, 381]$.

The actuator variable for controlling the saturation component is defined as the increment u_s which represents the distance from the radius (maximum saturation value $S_{max} = 255$) to the center of the color circle:

$$[S_{max} - u_s, S_{max}], \quad u_s \in [0, S_{max}]. \tag{4.13}$$

Color segmentation results for different saturation segmentation intervals can be seen in Figure 4.11. The hue image segmentation interval was manually set to $[104, 164]$. As can be seen, only a combination of optimal segmented hue and saturation images can provide a good, fully, segmented object.

Since the goal of robust color segmentation is to obtain good segmented objects with well connected pixels, the feedback variable to be used is the same as for the intensity case, that is the uncertainty measure I_m, defined in Equation 4.6. Starting from I_m, the optimal values $u_{h\ opt}$ and $u_{s\ opt}$ can be determined.

Input-output controllability

The input-output controllability has been investigated as for the case of intensity segmentation. The initial color angle H was set to 0, representing the red color. This corresponds to the initial color thresholding interval $[T_{min}, T_{max}] = [0, 60]$. Also, the initial saturation segmentation interval was set to its maximum value $[0, 255]$.

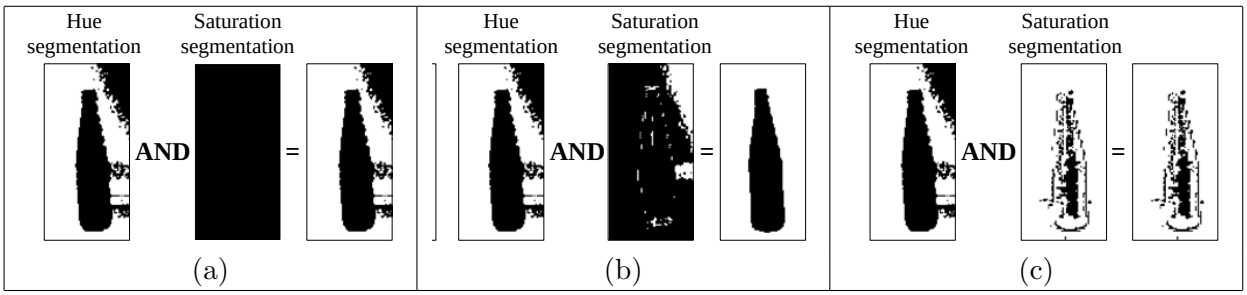


Figure 4.11.: Color segmentation results for different saturation segmentation intervals and constant optimal hue segmentation. (a) $[T_{min}, T_{max}] = [0, 255]$. (b) $[T_{min}, T_{max}] = [74, 255]$. (c) $[T_{min}, T_{max}] = [183, 255]$.

To the initial hue angle $H = 0$, the increment value u_h was added as in Equation 4.12. The hue angle was varied in the interval $[0, 360]$. Further, for each value of u_h, the saturation interval was varied in the interval $[0, 255]$. This was done by adding the increment u_s as $[255 - u_s, 255]$. The resulting input-output characteristic represents the variation of the uncertainty measure I_m with respect to the hue and saturation variation. In Figure 4.12(a), the input-output characteristic obtained by varying the object color class increment u_h can be seen. For the sake of description clarity, the saturation variation is displayed in Figure 4.12(b) only for a number of minimas in the hue characteristic corresponding to green, blue and red objects, respectively. The curves in the diagrams have been processed using the smoothing filter from Equation 4.9.

The optimal color segmentation parameters, represented by the pair $\{u_{h\ opt}, u_{s\ opt}\}$, are described by the minimal uncertainty measure I_m in the hue and saturation characteristics simultaneously. Again, as for the case of intensity segmentation, the problem of finding the optimal segmentation is represented by finding the minimum of the uncertainty measure I_m in both the hue and the saturation curves.

Because of the shape of the characteristic in Figure 4.12(a), which has different local minimas corresponding to different colored objects present in the input image, the search for the optimal segmentation parameters is linked to the proper definition of the effective operating ranges $[u_{low}, u_{high}]$ of the extremum search algorithm from Table 2.1. The case of the saturation characteristic in Figure 4.12(b) is simpler since the optimal saturation thresholding parameters correspond to the global minimum, hence the operating range for this case is $[0, 255]$.

Control structure design

For controlling the color segmentation process, the cascade control structure from Figure 4.13 has been proposed.

The inner-loop from Figure 4.13 is responsible for finding the optimal value of the hue thresholding interval $u_{h\ opt}$ represented by the object color class C_l. On the other hand, the objective of the outer-loop is to find the optimal value of the saturation threshold

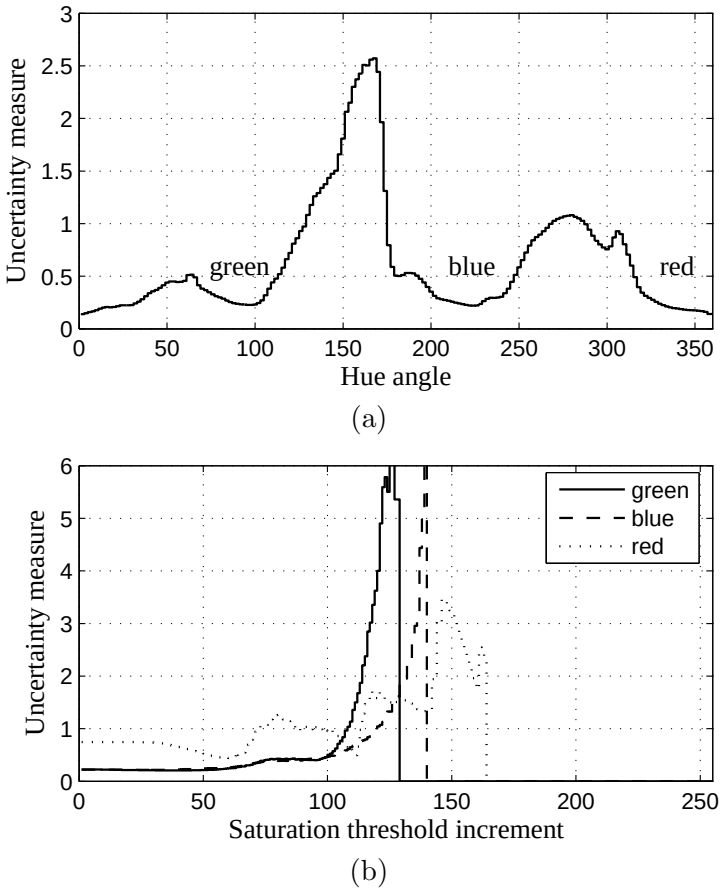


Figure 4.12.: The uncertainty measure I_m of segmented pixels vs. hue angle H (a) and vs. the saturation threshold increment u_s corresponding to the minimas of u_h representing green, blue and red objects, respectively (b).

increment $u_{s\ opt}$.

Both closed-loop structures presented in Figure 4.13 represent feedback optimization mechanisms like the one illustrated in Figure 4.8 for the case of intensity based segmentation.

The effective operating range $[u_{low}, u_{high}]$ can be manually chosen on a specific color section of the hue circle, in the interval $[0, 360]$. Also, as it will be discussed in Chapter 6, the effective operating range can be automatically determined for the purpose of finding multiple objects present in the input image.

A pseudo-code description of the proposed closed-loop color based segmentation method is given in Table 4.1.

In Figure 4.14, the variation of the optimal color segmentation parameters $[u_{h\ opt}, u_{s\ opt}]$ over a dataset of images acquired in illumination conditions ranging from 15lx to 1000lx can be seen. This corresponds to an illumination interval ranging from a dark room lighted with candles (15lx) to the optimal lighting level of an office (500lx) and above.

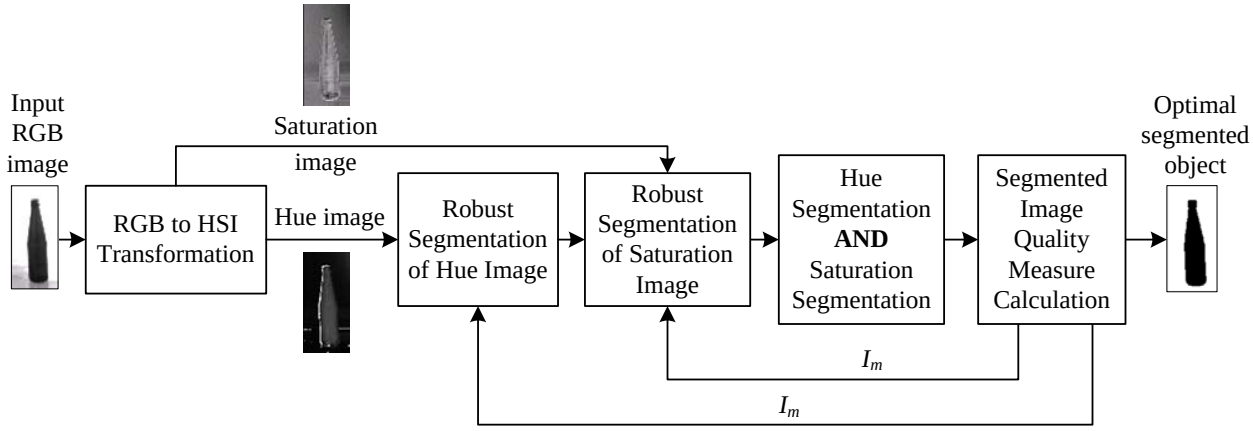


Figure 4.13.: Cascade closed-loop control structure for color segmentation.

Table 4.1.: Pseudo-code of the robust color segmentation algorithm.

Initialize $i = 0$, $j = 0$;
for $u_{low} : 1 : u_{high}$ **do**
1. Threshold the hue image $f_h(x, y)$ with thresholding interval $[u_h, u_h + K_{hw}]$;
2. Store the thresholded image in $t_h(x, y)$;
for $u_s = 0 : 1 : 255$ **do**
3. Threshold the saturation image $f_s(x, y)$ with thresholding interval $[u_s, 255]$;
4. Store the thresholded image in $t_s(x, y)$;
5. Combine the hue and the saturation segmentation results as $t(x, y) = t_h(x, y)$ **AND** $t_s(x, y)$;
6. Calculate the uncertainty measure $I_m(i, j)$ of the binary image $t(x, y)$; $j = j+1$;
end for
7. $i = i + 1$;
end for
Find min $I_m(i, j)$.

The characteristics in Figure 4.14 show the optimal value of color segmentation parameters, calculated using the proposed closed-loop method, as illumination varies in the interval $[15, 1000]$. The nonlinear variation of the uncertainty measure in Figure 4.14 comes from the fact that the segmentation is influenced not only by the intensity of illumination, but also by the position and type of illuminant. For example, shades can be produced by positioning the illuminant on one side of the imaged object. In this case the shades can be erroneously segmented as object pixels.

4.2. Robust boundary based segmentation

The recognition of textured objects, like books, is mainly done through methods that detect their boundaries, or edges [30]. Such a method is the *canny edge detector* which

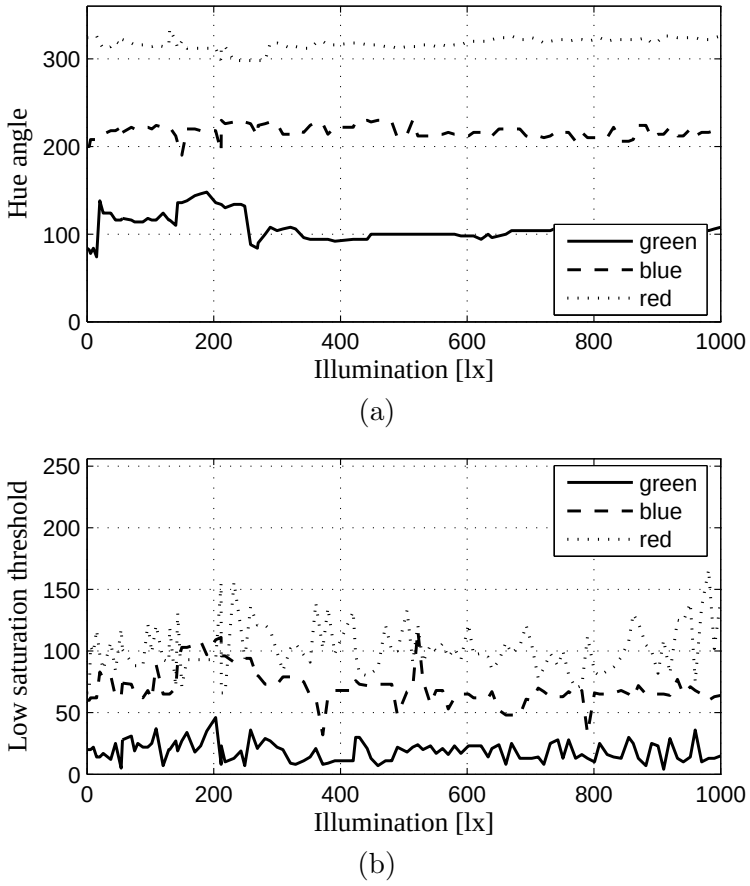
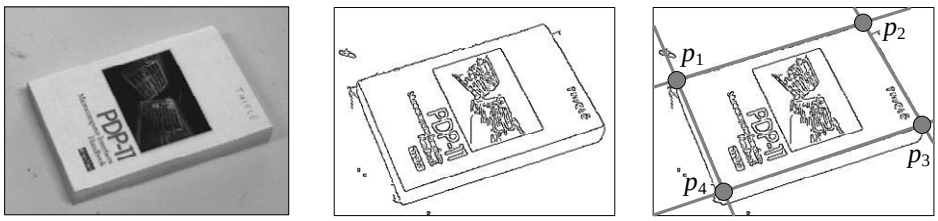


Figure 4.14.: Variation of hue (a) and saturation (b) threshold increments over a number of images acquired in various illumination conditions.

aims at classifying as foreground object pixels the ones that lie on the edges of objects. Pixels are considered as edges if they lie on sharp local changes in the intensity of an image. The output of segmentation is a binary image where foreground object pixels have the value 1 (black) and background pixels the value 0 (white), as seen in Figure 4.15(b). One main drawback of pure, raw, edge segmentation is that often breaks between edge pixels are encountered. For the case of line edges, a way around this problem is to evaluate the collinearity of binary edge pixels. This evaluation can be performed using the so-called *hough transform* [37] which converts the raw edge pixel data to a parameter space suitable for collinearity analysis. In order to distinguish between raw edge lines and lines calculated with the hough transform, the latter will be referred to as *hough lines*. In Figure 4.15(c), the gray lines represent the extracted hough lines, whereas the gray circles the extracted 2D object feature points. As convention, the numbering of the feature points is made in a clockwise manner.

In this chapter a robust, closed-loop, boundary object detection method based on the canny detector and the hough transform, both explained in Chapter 2.4, is proposed. The idea of the method is to adjust the parameters of canny and hough transform to



(a) (b) (c)

Figure 4.15.: Boundary feature point extraction. (a) Input image of a book. (b) Edge segmented image. (c) Hough lines and feature point extraction.

the optimal operational points based on the evaluation of the detected lines in the input image.

Open-loop object boundary segmentation

In this thesis, boundary segmentation is considered as the combination of raw edge segmentation and hough transform. Edge segmentation is performed using the canny edge detector [16] applied on the intensity image. Basically, the canny algorithm is performed through two main steps: filtering of the input intensity image with the derivative of Gaussian of a scale σ and thresholding the filtered image by the so-called *hysteresis thresholding*. Gaussian filtering aims at noise reduction where the degree of smoothing is determined by the value of σ. The binary edge detected image is calculated with the help of low T_L and high T_H thresholds, aiming at detecting strong and weak edges, where the weak edges are included in the output image only if they are connected to strong edges. The low threshold can be expressed as a function of the high threshold as:

$$T_L = 0.4 \cdot T_H. \tag{4.14}$$

An example of a Canny edge segmented image can be seen in Figure 4.15(b).

One drawback of using only raw edge detection for boundary object extraction is that very often the obtained contour edges are not connected, that is, they have small breaks between the edge pixels. This phenomenon happens due to noise in the input image, non-uniform illumination and other effects that introduce discontinuities in the intensity image [30]. The hough transform [37] is a method used in linking edge pixels based on shape. Although any shape can be expressed by the so-called generalized hough transform, in practice, because of computational expenses, shapes like lines or ellipses are used. In this thesis, the goal is to extract the lines that bound a book. These lines are calculated by estimating the collinearity of raw edge pixels. The hough transform maps the binary edge pixels to the so-called *accumulator cells*. Initially, the accumulator cells are set to 0. For every foreground pixels that lies on a line, a specific cell of the accumulator is increased. *The higher the number of pixels that lie on a line, the higher the values of the corresponding accumulator cell is.* Since the value of the accumulator entries reflects the

collinearity for all foreground edge pixels, it is meaningful to threshold it, so to consider as hough lines only the ones which have an accumulator cell value higher than a specific *hough threshold* T_{HG}. In Figure 4.15(c), the gray lines represent the detected hough lines.

A crucial requirement for reliable object manipulation using visual information is the robust extraction of object feature points used for 3D reconstruction. This requirement is strictly related to the quality of boundary segmentation. A boundary segmented image is said to be of good quality if the calculated object boundaries lie on its real boundaries. The extension of the boundary segmentation algorithm presented in here employs the idea of inclusion of feedback structures at image processing level to control the quality of the segmented image. The idea behind this approach is to change the parameters of image ROI segmentation in a closed-loop manner so that the current segmented image is driven to the one of reference quality independently of external influences.

The values of canny and hough transform thresholds are usually used as constant values which poses problems in variable illumination conditions. This problem is exemplified in Figure 4.16, where object feature points of a book are extracted using the constant boundary segmentation parameters. The parameters are determined in reference artificial illumination conditions. As can be seen from Figure 4.16, the feature points are reliably extracted for the case of artificial illumination. When the same constant parameters are used for recognizing the object in changed illumination (e.g. daylight) the output result is incorrect. In the next paragraphs, a closed-loop method for automatic adjustment of these thresholds, as illumination during image acquisition changes, is introduced.

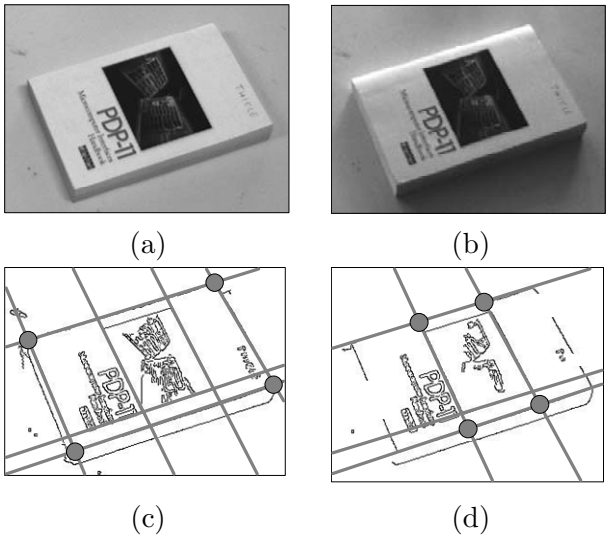


Figure 4.16.: Image of the same scene acquired under artificial - 458 lx (a) and daylight - 143 lx (b) illumination conditions. (c) and (d) object feature points extraction using constant boundary segmentation parameters.

Feedback control structure and actuator variables

In Figure 4.17 the block diagram of the proposed cascade closed-loop boundary segmentation method is displayed. In the presented system, the reference value of the chosen controlled variable is not explicitly known, since the goal is to develop a method able to detect objects independent of their sizes, color, or texture information. The objective of the control structure from Figure 4.17 is to find the maximum value of the controlled variable y. This is achieved through a feedback optimization process using an appropriate extremum seeking algorithm, as will be further explained.

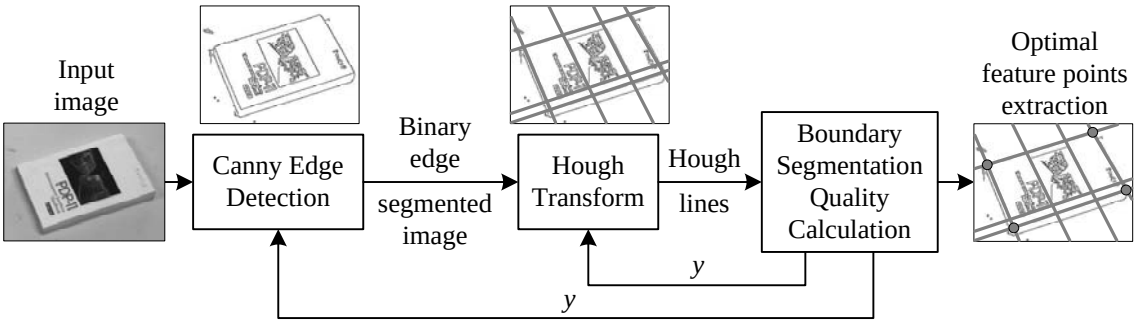
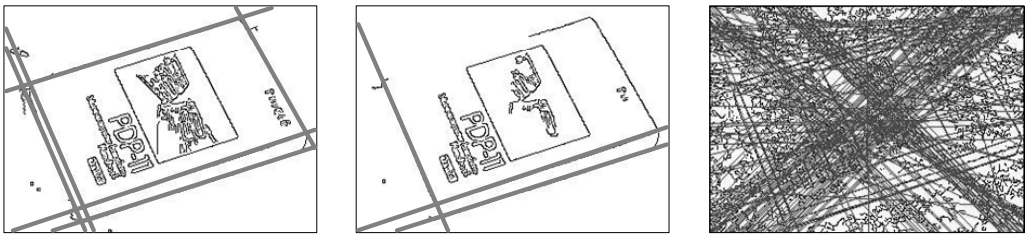


Figure 4.17.: Cascade control structure for robust boundary object detection.

In closed-loop image processing, actuator variables are those parameters that directly influence the image processing result. Since the boundary segmentation method used in ROVIS is composed of the canny edge detector and the hough transform, the actuators are chosen as the parameters that most strongly influence these operations. The result of canny edge detection is dependent on the choice of low T_L and high T_H thresholds. For the sake of clarity, T_L is considered to be a function of T_H, as shown in Equation 4.14. For the rest of this thesis, the canny thresholds will be referred only to T_H. On the other hand, the hough transform is strongly influenced by the value of the accumulator threshold T_{HG}.

The outer-loop from Figure 4.17 is responsible for finding the optimal threshold of the canny edge detector, according to the feedback variable y. The output binary edge image represents the input to the inner-loop of the control structure. In Figure 4.18, different detected hough lines are shown, for different values of canny parameters. The values of the hough threshold T_{HG} is constant. As can be seen, the correct number of lines describing the object of interest is extracted only for the case of optimal canny threshold ($T_H = 100$). The other examples yield either a too low (Figure 4.18(b)) or too high (Figure 4.18(c)) number of hough lines.

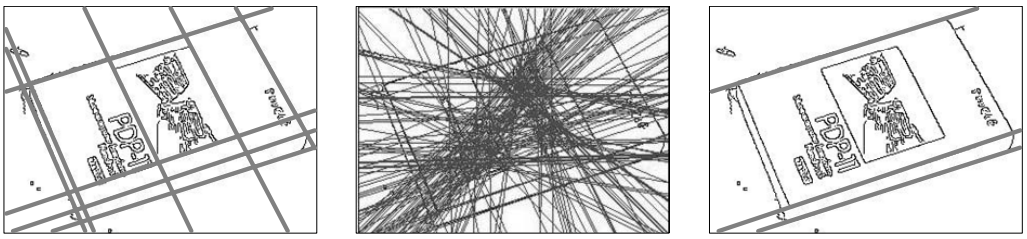
The goal of the inner-loop is to find the optimal working point of the hough transform parameters. As described in Chapter 2.5, the hough transform is capable of obtaining real edges even when the segmented edge is broken, a process also known as edge linking. In order to emphasize on the correct choice of hough transform threshold T_{HG}, in Figure 4.19 different lines detection results are shown for different values of the hough threshold.



(a) (b) (c)

Figure 4.18.: Lines detection corresponding to $T_{HG} = 50$ and different values of the canny threshold. (a) $T_H = 100$. (b) $T_H = 240$. (c) $T_H = 20$.

Again, as seen from the previous example, only a proper choice regarding the threshold value can output a reliable number of object lines.



(a) (b) (c)

Figure 4.19.: Lines detection corresponding to $T_H = 100$ and different hough transform threshold. (a) $T_{HG} = 65$. (b) $T_{HG} = 30$. (c) $T_{HG} = 100$.

Choice of the feedback variable

In order to control the T_H and T_{HG} thresholds, a measure of boundary segmentation quality has to be defined. This quality measure is to be used as the controlled variable in the proposed closed-loop system. A good boundary segmented image is one where the detected hough lines lie on the real object's edges. The feedback variable y from Figure 4.17 should be a description of the obtained lines in the image. Since y is a measure of lines quality dependent on the application, it should describe the lines based on a *contextual model* represented by the optimal combination of lines which reliably describes the shape of an object, here also called *reference*, or target, model.

In a large number of model-based vision systems [19] an object model refers strictly to a 3D prototype representation. This approach is relatively rigid and relies on a good performance of image segmentation for detecting the 3D model. In this thesis, the term reference model signifies the shape signature of an object in the 2D image plane. The feedback controlled variable will therefore represent a description of the target object based upon a set of object features derived from the detected hough lines.

Object boundary detection is used in the Library support scenario of the FRIEND system for the detection of books (Chapter 6.2), as well as in detecting container objects

for image ROI definition (Chapter 5.3). A book reference model is represented as a combination of parallel and perpendicular lines, forming the standard shape of a book. The construction of the candidate solutions represent a combinatorial expansion of the angles between detected lines in the input image. The angle of a line, ν, is measured with respect to the x axis of the 2D image plane, as seen in Figure 4.20. Ideally, between two parallel lines the difference in their angles should be 0, whereas for perpendicular lines $\pi/2$. Considering the camera's viewing angle and possible image distortions, the decision of classifying two lines as parallel or perpendicular has been done by introducing two offsets. Two lines are considered to be parallel if the difference in their angles with respect to the x axis is smaller then 12°, or $0.209rad$. Likewise, two lines are considered perpendicular if the angles difference varies in the small interval $[\pi/2-0.209, \pi/2+0.209]$.

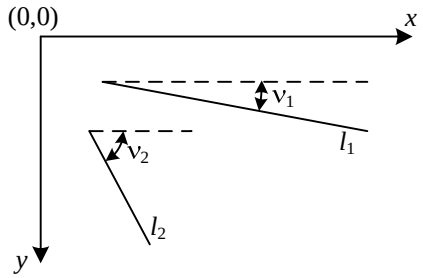


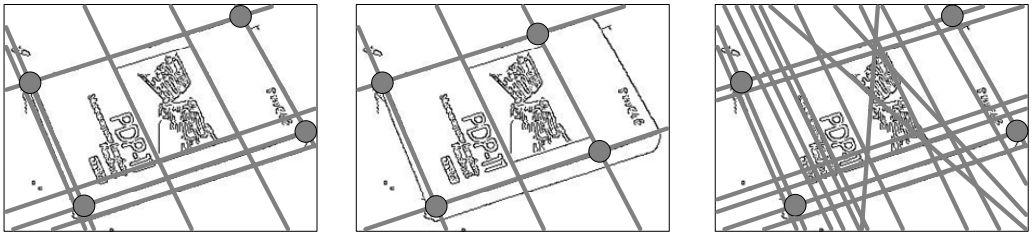
Figure 4.20.: Lines representation for quality measurement.

After grouping lines into parallel pairs, they are tested for perpendicularity. If one line pair is perpendicular to another one it is considered as a *candidate object solution* and added to the candidate solutions vector $N_{\#}$. The quality measure for robust boundary segmentation can be calculated based on the candidates solutions vector $N_{\#}$. The equation of the proposed measure is:

$$y = \begin{cases} e^{N/N_{max}} \cdot \sum_{n=1}^{N_{\#}} \frac{N_O(n)}{P_{ROI}}, & \text{if } N \leq N_{max}, \\ 0, & \text{if } N > N_{max}, \end{cases} \tag{4.15}$$

where N represents the total number of hough lines, $N_{\#}$ the number of candidate solutions and $N_O(n)$ the number of foreground pixels covered by the hough lines of the n^{th} object, normalized with the perimeter of the image, P_{ROI}. Having in mind the computational burden of the hough transform, the maximum number of lines allowed in an image is set to a constant value N_{max}. The exponential term in Equation 4.15 is introduced in order to force feature extraction with a minimum amount of hough lines. Hence, y decays to zero when the number of hough lines increases. In Figure 4.21, the value of the controlled variable y for different boundary segmentation results can be seen.

As can be seen from Figure 4.21, *the higher the value of the quality measure y is, the better the image segmentation quality is.* To investigate the system's input-output



(a) (b) (c)

Figure 4.21.: Different values of the feedback variable y for different segmentation results. (a) Ideally segmented $y = 0.5728$. (b) Undersegmented $y = 0.4821$. (c) Oversegmented $y = 0.5183$.

controllability when considering the thresholds T_H and T_{HG} as the actuator variables and the measure y as controlled variable, boundary segmentation was applied on the image from Figure D.2(a) (see Appendix D). The value of T_H was varied in the interval $[0, 255]$, whereas the value of T_{HG} in the interval $[0, 100]$. For each combination of thresholds $\{T_H, T_{HG}\}$, the controlled variable y was measured. The input-output result can be seen in Figure 4.22. Optimal boundary segmentation corresponds to the combination of thresholds which maximize the variable y. It can be observed from Figure 4.22 that different combinations of thresholds yield the same value of the quality measure. This is because the same optimal feature extraction result can be achieved with different values of T_H and T_{HG}.

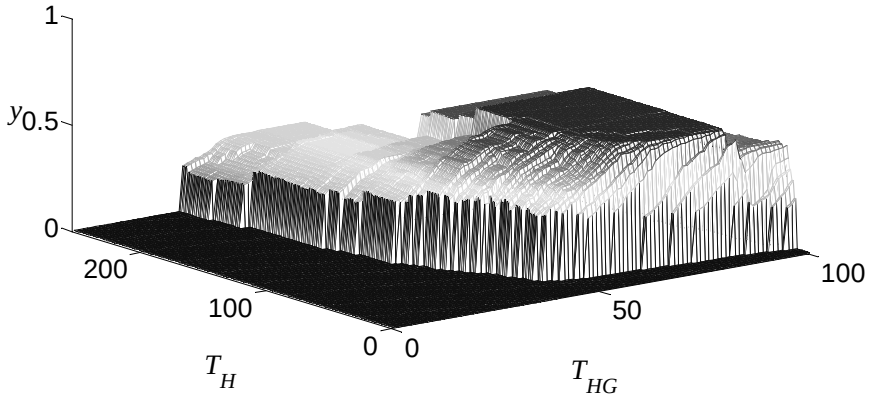


Figure 4.22.: The quality measure y vs. canny T_H and hough transform T_{HG} threshold.

Feedback control design

The block diagram of the proposed control structure for robust boundary segmentation is illustrated in Figure 4.17. The objective of the control structure is to find the maximum value of the controlled variable y. This is achieved through a feedback optimization process using an appropriate extremum seeking algorithm. Since, as said before, optimal

feature extraction is achieved for different values of thresholds, the extremum seeking algorithm stops when the gradient of the surface in Figure 4.22 reaches the value 0.

Feedback optimization on the whole range of canny and hough transform parameter space requires a high amount of computation power. In order to reduce it, a method for determining the effective operating ranges of the thresholds has been set. For canny edge detection, the operating range $[u_{C\ low}, u_{C\ high}]$ is determined by examining the amount of segmented edge pixels, represented as:

$$R_{o2t} = \frac{number\ of\ segmented\ edge\ pixels\ in\ image}{total\ number\ of\ pixels\ in\ image}. \tag{4.16}$$

The edge segmented image is considered to be noisy if the ratio R_{o2t} exceeds the heuristically determined value 0.8. Hence, the value of the lowest canny threshold $u_{C\ low}$ is represented by the value where $R_{o2t} < 0.8$. On the other hand, the highest canny threshold $u_{C\ high}$ is always the maximum gray level value found in the input intensity image.

The operating ranges of the hough transform are determined from the already calculated operating ranges of the canny edge detector, that is from the $u_{C\ low}$ and $u_{C\ high}$ values. The binary segmented image corresponding to $u_{C\ low}$ contains the maximum number of segmented object pixels, hence the maximum number of hough lines. The high boundary $u_{HG\ high}$ is calculated in an iterative manner by decreasing the threshold T_{HG} applied to accumulator cells until the number of detected lines is equal or bigger than 4 ($N \geq 4$). 4 is the minimum number of lines needed to form an object.

In Table 4.2, a pseudo-code of the proposed feedback optimization algorithm for boundary segmentation is given. Since both the canny and the hough transform are discrete operations, the feedback optimization process is performed using a step increment of value 1.

Table 4.2.: Pseudo-code of the robust boundary segmentation algorithm.

Initialize $i = 0$, $j = 0$;
for $T_H = u_{C\ low} : 1 : u_{C\ high}$ **do**
    1. Obtain the canny binary edge detected image and store the result in $t_C(x, y)$;
    2. Calculate the accumulator array of the hough transform;
    for $T_{HG} = u_{HG\ low} : 1 : u_{HG\ high}$ **do**
        3. Threshold the accumulator array and get the hough lines N;
        4. Combine the obtained hough lines and get the candidate solutions vector $N_{\#}$;
        5. Calculate the quality measure y;
        6. Store the value of $y(i, j)$ corresponding to the current pair of canny and hough transform parameters; $j = j + 1$;
    end for
    7. $i = i + 1$;
end for
find max $y(i, j)$.

The main problem raised by the "hill climbing" algorithm from Table 4.2 is the high

computation time required to calculate the optimal values of T_H and T_{HG}. In order to overcome this problem, the genetic algorithms optimization structure from Figure 4.23 is proposed. A brief introduction to genetic algorithms [67] is given in Appendix C. Genetic algorithms were chosen for finding the optimal values of canny and hough transform thresholds because they exhibit high optimization speed and low probability to get stuck in local maximums.

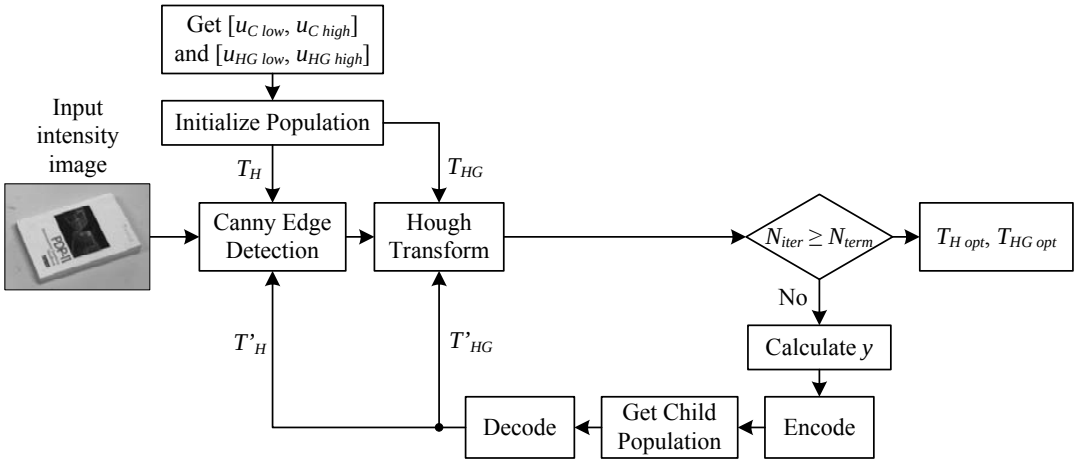


Figure 4.23.: Block diagram of genetic algorithms optimization for robust boundary segmentation.

First, the effective operating range is calculated, defined by the intervals $[u_{C\ low}, u_{C\ high}]$ for canny and $[u_{HG\ low}, u_{HG\ high}]$ for hough transform. Further, the initial population needed for genetic optimization is selected by randomly choosing different parameter pairs $\{T_H, T_{HG}\}$. The obtain pair is referred to as individuals of the population, as explained in Appendix C. For each individual, the boundary quality measure y is calculated. After this procedure is repeated for the whole population, a new child population is generated based on the calculated quality measure. The selected individuals are encoded as binary strings and new individuals are generated by the use of crossover and mutation [67]. After decoding the binary string, the procedure is repeated with the new parameter set (T'_H, T'_{HG}). It has been observed that the proposed method converges to an optimal solution after an average number of three iterations $N_{iter} = 3$. Hence, the stopping criteria for the feedback loop in Figure 4.23 is the maximum iterations number N_{iter}. The output of the optimization process is the pair of optimal canny and hough thresholds $\{T_{H\ opt}, T_{HG\ opt}\}$.

5. Image Region of Interest (ROI) definition in ROVIS

In robot vision, using an image Region of Interest (ROI) in which image processing algorithms are to be applied has a series of advantages. One of them is the reduction of the scene's complexity, that is the reduction of the object's search area from the whole imaged scene to a smaller region containing the object(s) to be recognized. As explained in Chapter 3, ROI definition represents in ROVIS a pre-processing stage. Its aim is to provide as input to the object recognition chain a subarea in the image where the search for the object(s) of interest takes place.

In this chapter two approaches used for defining the image ROI are presented, each of them depending on the amount of contextual knowledge information available:

- *bottom-up ROI definition*, where image-based criteria is used in defining a ROI from groups of pixels that are likely to belong together;
- *top-down ROI definition*, which uses a priori scene information learned from examples.

Following, the computer vision definition of an image ROI will be given. The bottom-up approach will be explained in the context of the ROVIS architecture, where user interaction is used for defining interest points in the input image. Further, two top-down ROI definition methods will be explained together with how knowledge regarding the imaged scene is integrated in the algorithms. This knowledge comes either from the detection of containers boundaries, or from the recognition of natural SIFT markers placed on containers [10]. The definition of a container object has been given in Chapter 3.2.1.

Both approaches presented in this chapter are a core concept of the ROVIS architecture. The object recognition chain described in Chapter 6 is based on the optimal detection of the image ROI.

5.1. Definition of the image ROI

Although the meaning of ROI is strictly dependent on the application, a common accepted definition is, as the name suggests, *a part of the image for which the observer of the image shows interest* [17]. The interest region is not only dependent on the image, but also on the observer itself. In [81], ROIs are classified in two types: *hROIs* (*h*uman identified ROIs) and *aROIs* (*a*lgorithmically detected ROIs). In human perception, context-dependent

sequences of eye movements fixate the hROIs. A medium of three eye fixations per second are generated by a human subject during active looking. These eye fixations are intercalated by rapid eye jumps, called saccades, during which vision is suppressed. Only a small set of eye fixations, hROIs, are usually required by the brain to recognize a complex visual input [81]. The aROIs are generated automatically by image processing algorithms that usually intend to detect and localize specific features in an image (e.g. color, spacial frequency, texture information etc.).

From the computer vision point of view, the definition of the image ROI can be derived from Figure 5.1. For the sake of clarity, only one ROI is assumed to exist in an image at a specific moment of time. On an input image $f(x, y)$, with its coordinate system located at the "top-left" corner, the ROI is defined by a vector of four elements: $[x, y, w, h]$. (x, y) is the ROI's 2D image coordinate point, taken as the left-upper point of the ROI. The other 2 elements of the vector are the ROI's width w and height h. For the rest of the thesis, the ROI of an image $f(x, y)$ will be referred to as:

$$ROI(f|x, y, w, h). \tag{5.1}$$

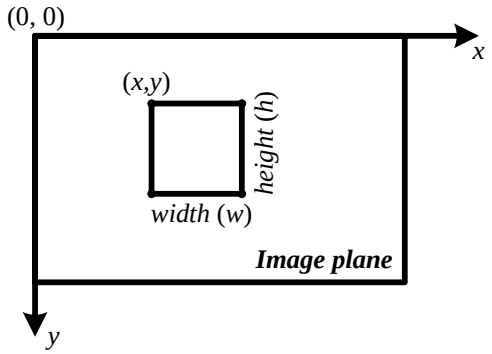


Figure 5.1.: ROI inside the 2D image plane.

5.2. Bottom-up image ROI definition through user interaction

In bottom-up image processing, information extracted at pixel level is used in building the image ROI. In the next paragraphs, a method for defining the image ROI for uniformly colored objects to be manipulated is presented. Within the ROVIS architecture, the bottom-up approach is sustained by the definition of an interest point in the image, $pt_{int}(x, y)$. This point is used as a starting position for adjusting the parameters of the initial ROI to its optimal values, that is, to surround the desired object. For the case of the FRIEND robot, interest point definition is performed through the *Human-Machine Interface* (HMI) component, as explained in Chapter 3. Such a ROI, which bounds only one object to be manipulated, corresponds to the user command "I want this object", as explained in Chapter 3.2.3.

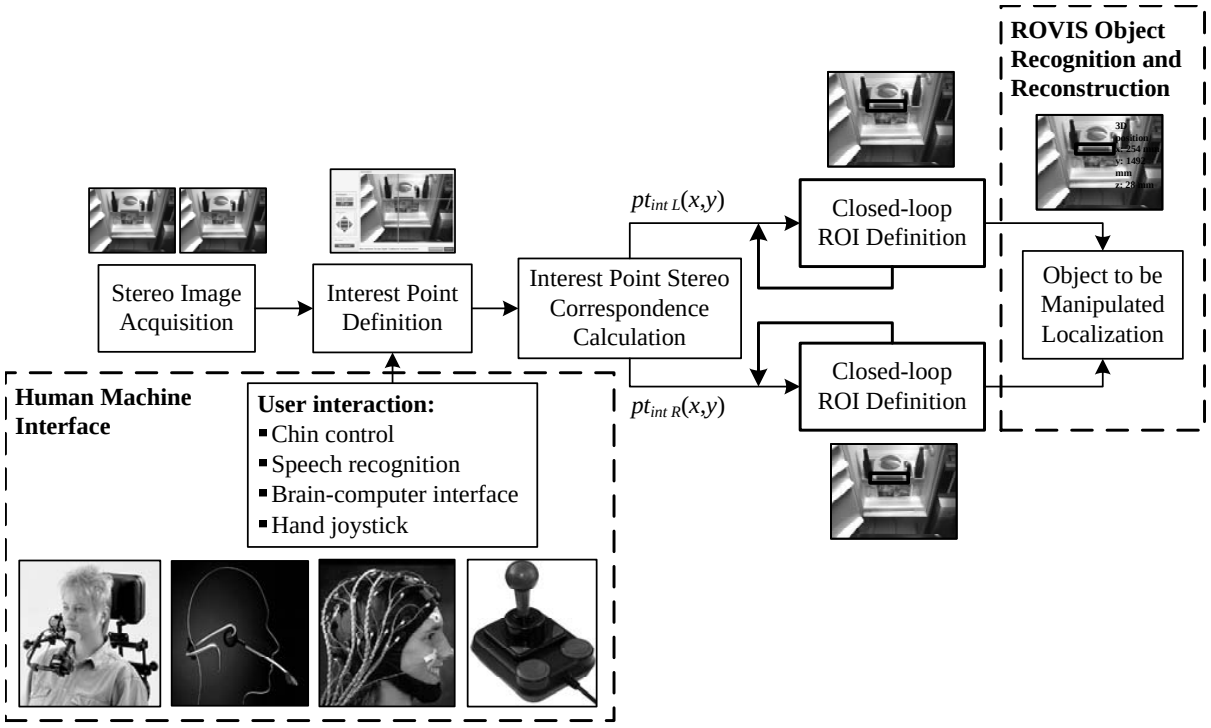


Figure 5.2.: Block diagram of robust ROI definition through user interaction.

In Figure 5.2, the block diagram for ROI definition through the HMI is illustrated. The starting point of the algorithm is the acquisition of a stereo images pair, followed by the definition of the interest point. For reducing at maximum the need for user interaction, the interest point is defined through the HMI only once, on the left image of the stereo pair, as $pt_{int_L}(x, y)$. For the right image, the corresponding interest point $pt_{int_R}(x, y)$ is calculated using epipolar geometry. The result of the calculations provides a stereo interest point pair for a pair of stereo images:

$$\{pt_{int_L}(x, y), pt_{int_R}(x, y)\}. \tag{5.2}$$

The points pair in Equation 5.2 is provided as input to the closed-loop ROI definition algorithm proposed in this chapter. The algorithm calculates the pair of stereo ROIs:

$$\{ROI(f_L|x, y, w, h), ROI(f_R|x, y, w, h)\}. \tag{5.3}$$

The final ROIs in Equation 5.3, optimally bounds the desired object in both left and right images, respectively.

Interest point definition

The connection between the user of FRIEND and ROVIS is moderated by the HMI. Depending on the motoric capabilities of the user, he/she interacts with the robotic system

through several input devices, like chin joystick, speech recognition, *Brain-Computer Interface* (BCI) or hand joystick. Using such a device the user can provide commands to the robot by controlling a cursor on a display monitor. In a similar manner, the controlled cursor can be used to define the interest point $pt_{int_L}(x, y)$. After the interest point is defined, the proposed closed-loop ROI definition algorithm is used to bound the object of interest automatically in both stereo images.

One very important aspect of the proposed method is its robustness with respect to the given image interest point. In order not to constrain the user with a tiring task, as defining a precise point on the object of interest, the method presented here must function when the interest point is given not on the object but on its vicinity. This signify that *the algorithm must be robust with respect to ill defined interest points.* As will be further seen, this robustness is achieved through feedback mechanisms included in the image processing algorithms involved in bottom-up ROI definition.

5.2.1. Problem statement

The goal of the proposed algorithm is to calculate a pair of ROIs that will perfectly bound the object to be manipulated in the input stereo images pair. The formulation of the problem and the proposed solution will be described for the case of a single mono input image, the stereo approach being achieved by applying the algorithm twice, on the left and right image, respectively. The problem that has to be solved is how to bound the desired object to be manipulated with a ROI starting from an ill defined interest point, as plastically represented in Figure 5.3. In Figure 5.3, a uniformly colored, region based segmented, object is ideally bounded by a ROI.

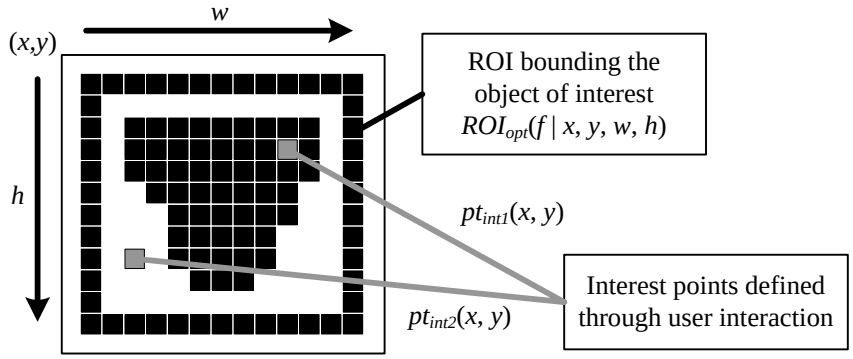


Figure 5.3.: Definition of an optimal image ROI of a segmented object obtained from two interest points, respectively.

In order to illustrate the image ROI definition problem, in Figure 5.4, the intermediate steps for defining the optimal ROI for the segmented object in Figure 5.3 have been displayed. The two presented cases, A and B, correspond to the selection of the interest points $pt_{int1}(x, y)$ on the object and $pt_{int2}(x, y)$ outside of it, respectively. Starting from each of the interest points and using an initial ROI_0, the algorithm must achieve the

optimal ROI that will bound the object:

$$ROI_0(f|x_0, y_0, w_0, h_0) \longrightarrow ROI_{opt}(f|x, y, w, h). \tag{5.4}$$

ROI_{opt} is defined as the rectangle that bounds the segmented object and has its edges at a one pixel distance from the object itself.

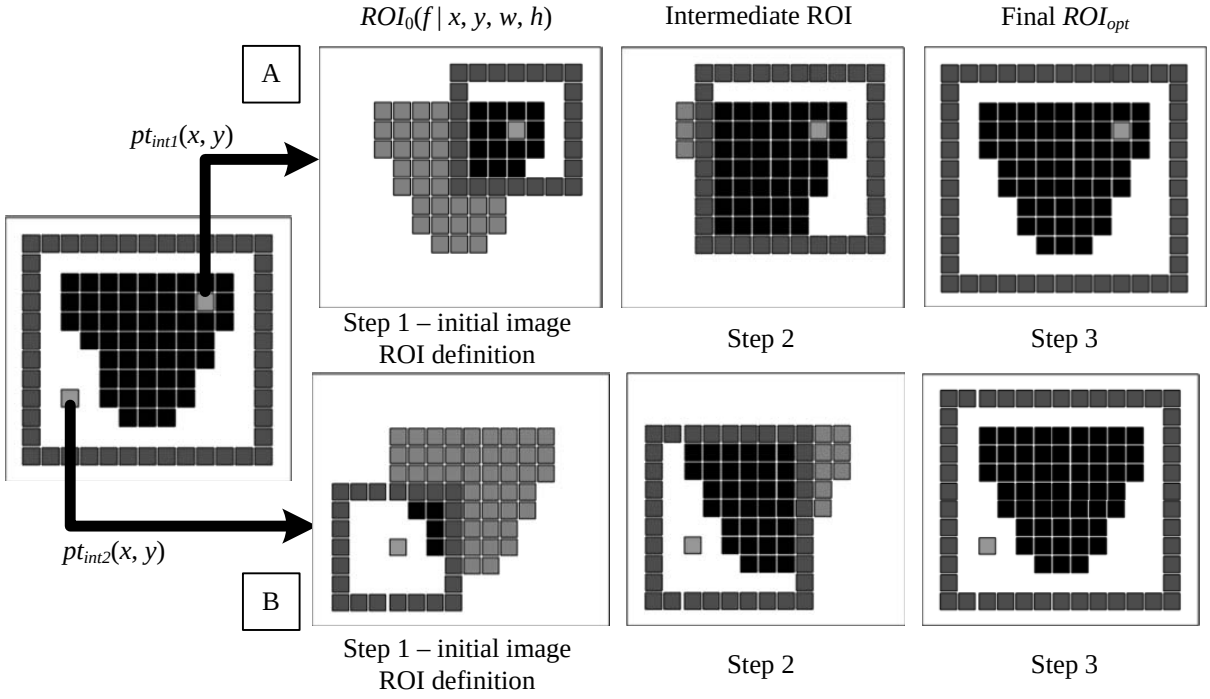


Figure 5.4.: Intermediate steps for defining an object's image ROI for two different cases of interest points.

In this thesis, the proposed solution for the image ROI definition problem is the development of a closed-loop control system that automatically adjusts the parameters of the current ROI based on segmentation evaluation. In the next paragraphs, such a closed-loop control system is presented, for the case of uniformly colored objects.

5.2.2. Control structure design

In Figure 5.5, the proposed cascade control system for image ROI parameters adjustment is presented. Although, the system is designed for the case of a region based segmented objects, the same concept can be applied for boundary based segmented objects.

The inner-loop of the cascade structure from Figure 5.5 is responsible for robust region segmentation of the current ROI. This inner-loop is represented by the robust region based color segmentation method from Chapter 4.1. The choice for this adaptive segmentation method comes from the fact that the values of optimal segmentation parameters are changing as the size of the image ROI is adjusted, as will be further explained. Based on

the segmentation result, the outer-loop automatically adjusts the parameters of the image ROI. In order to achieve this, the obtained segmentation is evaluated at each iteration step. The algorithm starts from an initial ROI, as explained below.

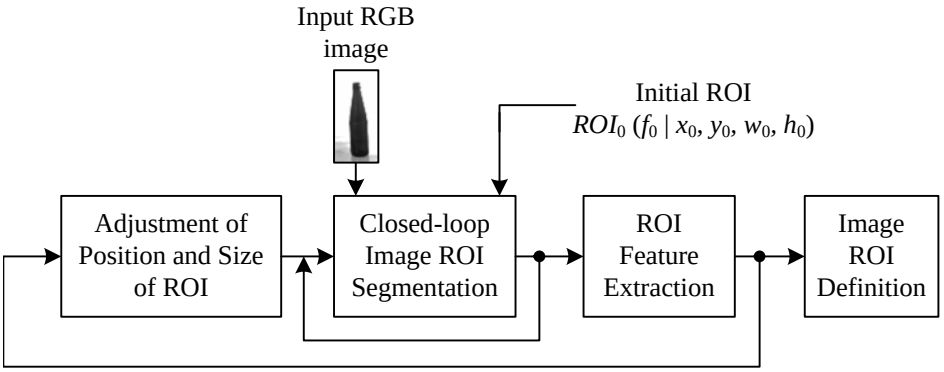


Figure 5.5.: Principle of cascade closed-loop image ROI definition.

Initial image ROI

The algorithm is initialized with the input image $f(x, y)$ and a predefined ROI imposed on the interest point $pt_{int}(x, y)$. On the initial ROI, the robust region segmentation operation is applied. This part of the algorithm plays a crucial role in the success of the method, since it represents the initial segmentation of the object in the initial ROI. In Figure 5.6(a), eight initial ROIs of objects to be manipulated, in the Activities of Daily Living (ADL) scenario of FRIEND, are presented. The robust region segmentation of the ROIs is presented in Figure 5.6(b). From the eight ROIs in the image, 1, 3, 4, 5 and 7 have their interest point defined on the object, whereas 2, 6 and 8 outside of it, in its vicinity. As can be seen from Figure 5.6(b), all the segmented ROIs contain as segmented object pixels a part of the object to be manipulated. This signifies a robustness of the method with respect to ill interest point definition. As it will be further explained, from the segmented regions, the final object ROIs will be calculated, respectively.

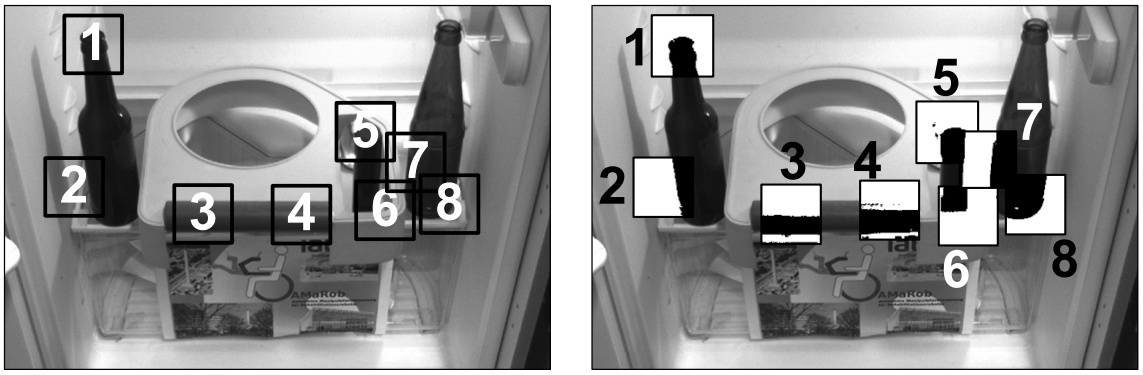
The initial ROI is defined with constant parameters:

$$ROI_0(f|x = x_0, y = y_0, w = w_0, h = h_0), \tag{5.5}$$

where the reference coordinate (x_0, y_0) of the ROI is calculated from the user defined interest point $pt_{int}(x, y)$ and the predefined values of the width w and height h of ROI_0:

$$\begin{cases} x_0 = pt_{int}(x) - (w_0/2), \\ y_0 = pt_{int}(y) - (h_0/2), \end{cases} \tag{5.6}$$

where $pt_{int}(x)$ and $pt_{int}(y)$ are the coordinates of the interest point on the x and y axes of the image plane, respectively.



(a) (b)

Figure 5.6.: Imaged scene from the FRIEND ADL scenario with different user-defined initial ROIs (a) and their respective segmentation results (b).

After initial segmentation, the edges of ROI_0 are set around the detected object according to the next two rules:

1 The position of the image ROI edges which intersect the object are left unchanged;

2 The position of the image ROI edges which do not intersect the object are set at a one pixel distance from it.

The purpose of modifying ROI_0 is to adjust the edges of the object's ROI close to its boundaries. From these new boundaries, the feedback optimization process of the ROI's parameters will start. Feedback optimitation is used in the outer-loop of the cascade structure from Figure 5.5. The procedure of automatic adjustment of the edges of an image ROI will be further explained, together with the choice of the *actuator – controlled variable pair* for the outer-loop.

Choice of the actuator – controlled variable pair

As explained in Chapter 2.1, the design of a closed-loop control system for image processing differs significantly from classical control applications, especially in the choice of actuator – feedback variables. In this section the characteristics of the outer-loop from Figure 5.5 will be detailed.

Having in mind that the goal of the proposed algorithm is to optimally bound the object of interest according to the image ROI segmentation result, a proper choice for the actuator – controlled variables is the 2D position of the edges of the ROI in the image plane. In the example from Figure 5.7, the ROI edges are moved in the directions where the object of interest is found, according to the segmentation result. On one hand, the positions of the top and right edges are changed towards the top and right image plane, respectively, since they intersect the segmented object. On the other hand, the bottom and left edge are translated towards the segmented object, that is to the top and right, respectively, since they do not intersect the object. An important note is that the

translation of the edges is given by the segmentation result performed only on the ROI.

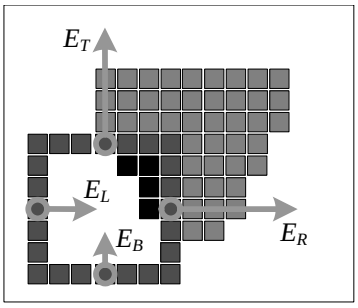


Figure 5.7.: 2D translation of ROI edges in the image plane.

In order to adjust the ROI's parameters vector from Equation 5.1, a mapping from the original ROI coefficients $[x, y, w, h]$ to the ROI edges from Figure 5.7 was performed:

$$ROI(f|x, y, w, h) \longrightarrow ROI(f|E_L, E_T, E_R, E_B). \tag{5.7}$$

where E_L, E_T, E_R and E_B corresponds to the left, top, right and bottom edges of the ROI, respectively.

The control of the position of an image ROI is achieved by manipulating its edges, that is, the position of $\{E_L, E_T, E_R, E_B\}$ in the 2D image plane. This position is changed with the actuator variable u_e, defined as an increment added to the current position of the ROI edges. Since a ROI is represented by four edges, an equal number of closed-loop control structures has to be implemented, that is, one feedback mechanism for each edge.

The controlled variable for the feedback loop is a measure of quality related to the positions of the ROI edges in the image plane. This quality measure is obtained by calculating the number of segmented object pixels touching the edges of the image ROI. Mathematically, this measure can be modeled as an estimate of probability of the number of object pixels placed on each edge of the segmented object, respectively:

$$\rho_i = \frac{number\ of\ segmented\ pixels\ on\ the\ i^{th}\ edge}{i^{th}\ edge\ length}. \tag{5.8}$$

where ρ_i represents the probability of the edge i to intersect segmented object pixels. Following the above reasoning, the optimal value of ROI_{opt} corresponds to $\rho_i = 0$, which is also the reference value for the proposed control system.

Feedback control design

The variable ρ_i is used in the proposed system as a switching element for readjusting the position and size of the object ROI. According to the value of ρ_i, the parameters of the ROI are changed using the actuator variable u_e. Having in mind the definition of the 2D image plane, given in Figure 5.1, u_e has a negative value for E_L and E_T and positive for

E_R and E_B. Given the actuator u_e, the relation between the positions of the ROI edges at two adjacent iteration steps is:

$$ROI_n(f|E_L^n, E_T^n, E_R^n, E_B^n) = ROI_{n-1}(f|(E_L^{n-1}-u_{eL}), (E_T^{n-1}-u_{eT}), (E_R^{n-1}+u_{eR}), (E_B^{n-1}+u_{eB})), \tag{5.9}$$

where n represents the ROI edges positions at the n^{th} iteration of the algorithm. u_{eL}, u_{eT}, u_{eR} and u_{eB} corresponds to the increment value added to the edges of the ROI, that is E_L, E_T, E_R and E_B, respectively.

Based of Equation 5.9, the value of u_e is calculated as:

$$u_{ei} = \begin{cases} K, & \text{if } \rho_i > 0, \\ 0, & \text{if } \rho_i = 0, \end{cases} \tag{5.10}$$

where $i \in \{E_L, E_T, E_R, E_B\}$. K represents an integer defined as the value of the actuator u_e, here chosen as $K = 1$.

The final structure of the closed-loop control system for bottom-up image ROI definition is displayed in Figure 5.8. As can be seen, the inner-loop is responsible for robust region based segmentation of the ROI, at each iteration step of the algorithm. The inner-loop takes as input, along with the RGB image, the initial ROI_0 calculated according to the defined interest point $pt_{int}(x, y)$. The outer-loop controls the positions of the edges of the ROI with respect to the number of segmented object pixels lying on the edges of the ROI. Since a ROI has four edges, four control loops are implemented, one for each edge. For the control structure from Figure 5.8, the reference value of the outer-loops corresponds to 0 segmented pixels on the ROI's edges, that is $\rho_i = 0$. The automatic ROI adjustment process is finished when the edges are positioned at a one pixel distance from the segmented object in the 2D image plane.

In Figure 5.9, the variation of the optimal hue angle u_h can be seen, for the case of three different uniformly colored objects. This variation was obtained from the robust region based segmentation method explained in Chapter 4.1. The value of u_h was calculated within the inner-loop of the control structure from Figure 5.8. As said before, the proposed adaptive segmentation method was used because the optimal value of segmentation parameters are changing with respect to the adjustment of the image ROI. In the example from Figure 5.9, after a couple of cycles, the hue angle reaches a stable value for all three objects, that is, the optimal hue angle needed for color segmentation was determined.

The outcome of this bottom-up image ROI definition method can be used directly as optimal region based segmentation in the ROVIS 2D feature-based object recognition module from Figure 3.6. This is because the algorithm is based on optimal segmentation for the definition of the image ROI.

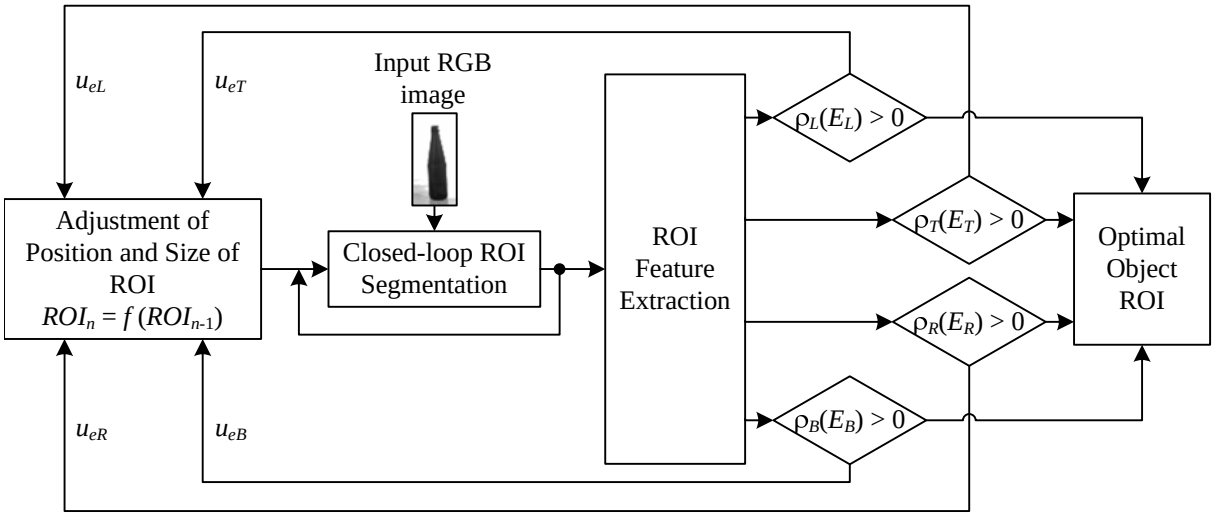


Figure 5.8.: Closed-loop control structure for robust object ROI definition.

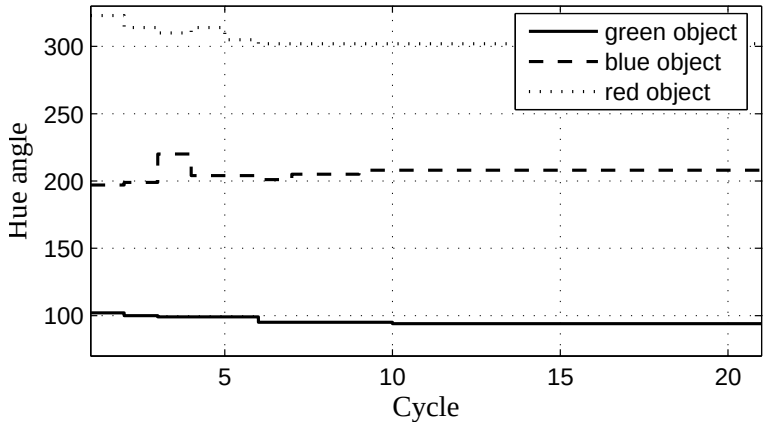


Figure 5.9.: Variation of the hue angle u_h for the case of three uniformly colored objects.

5.2.3. Performance evaluation

Experimental setup

The experimental setup for testing the proposed region based ROI definition algorithm is composed of four objects to be manipulated from the FRIEND ADL scenario: two bottles, a mealtray and a spoon. In order to test the algorithm's robustness with respect to variable illumination, the ROI definition method was applied on four images acquired under different illumination conditions. The test images can be as seen in Appendix D, Figure D.1(a-d). For each object to be manipulated, five interest points were defined, on and outside the boundaries of the objects. The method was applied on each interest point for each of the four scenes from Figure D.1(a-d), thus obtaining a number of 20 experiments per object. Figure 5.10 shows an example of different given interest points

for the case of a bottle (points 1 and 2 are placed on the object, whereas points 3, 4 and 5 outside of it).

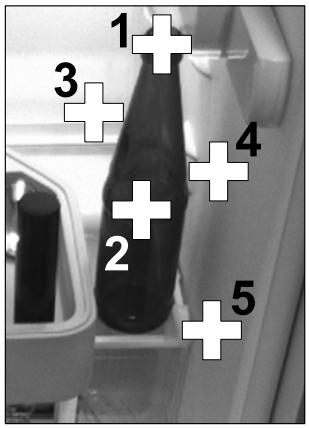


Figure 5.10.: Definition of interest points on an object to be manipulated.

In order to test the performance of the proposed approach, the closed-loop ROI definition method was compared with a traditional open-loop region growing [30] algorithm. The used region growing method considers as foreground object pixels the neighbors of the seed pixel (interest point) that have an intensity value in the interval range $[px - 10, px + 10]$, where px is the intensity value of the seed pixel. All pixels in the interval $[px-10, px+10]$ and connected to already segmented pixels are considered foreground. The final ROI is represented by the bounding box of the segmented object.

The calculated image ROIs from both methods were compared with the ideal ROI, manually obtained. For evaluation purposes, two performance metrics were defined:

- position and size error from the ideal ROI;
- number of missegmented pixels.

Position and size error from the ideal ROI

The position and size error from the ideal ROI represents an estimation of the 2D displacement of the calculated ROI in the image plane. In Figure 5.11, the errors of each ROI parameter can be seen, that is (x, y) coordinates, width and height. For the first 8 experiments the interest point was placed outside the object to be manipulated, whereas for the next 12 on it. As can be seen from the diagrams in Figure 5.11, the closed-loop algorithm is characterized by a constant tendency to zero error in all experiments, in comparison to the region growing one. If region growing provides relatively good results for interest points placed on the objects, for the points lying outside of it the results contain a large error.

The results from Figure 5.11 have been quantified as the following metric:

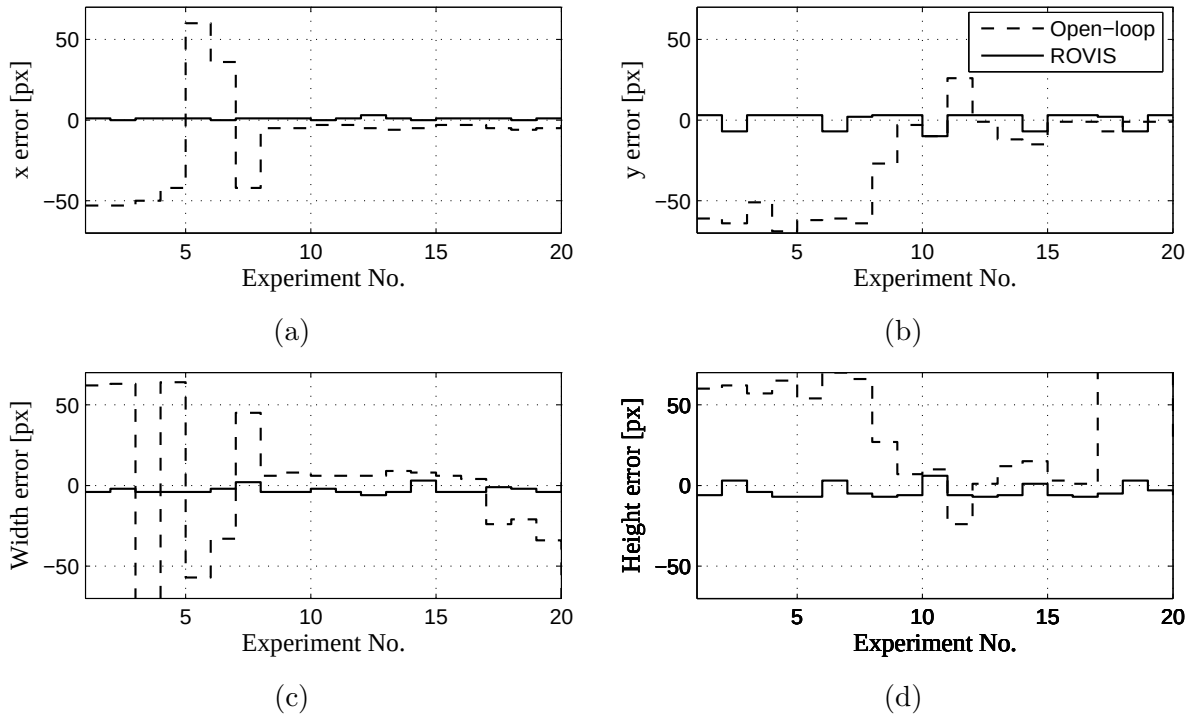


Figure 5.11.: Calculated image ROI position and size error from the ideal ROI.

$$d_p = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(x_{ri}-x_i)^2+(y_{ri}-y_i)^2+(w_{ri}-w_i)^2+(h_{ri}-h_i)^2}, \tag{5.11}$$

where n is the number of performed experiments and x_r, y_r, w_r and h_r are the manually determined reference values of the ideal image ROI coordinates, width and height, respectively. The obtained statistical results for the four considered objects to be manipulated are summarized in Table 5.1.

Number of missegmented pixels

A performance metric similar to the one above is the number of missegmented pixels in the image ROI. In this case, the calculated values are the number of segmented pixels in the obtained ROI and the ideal, reference, number of segmented pixels obtained from the reference ROI. As in the previous metric, the ideal value for the metric is zero. As can be seen from Figure 5.12, the ROVIS closed-loop results have a constant tendency to zero error in comparison to the region growing ones.

The quantification of results from Figure 5.12 is performed using Equation 5.12. The statistical results are shown in Table 5.1.

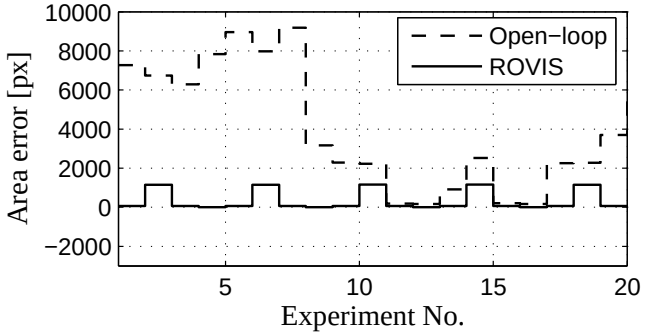


Figure 5.12.: Number of missegmented pixels. Dotted line – image ROI definition through region growing. Solid line – proposed closed-loop image ROI definition.

$$a_p = \frac{1}{n}\sum_{i=1}^{n}(Area_{ri} - Area_i), \tag{5.12}$$

where n is the number of performed experiments and $Area_r$ is the reference number of segmented object pixels. As can be seen from Table 5.1, the error values given by d_p and a_p for the ROVIS closed-loop approach are much smaller then the ones of the region growing algorithm, thus emphasizing the robustness of the proposed approach with respect to both variable illumination conditions and interest point definition.

Table 5.1.: Average error values of the calculated ROIs.

Object	Bottle 1		Bottle 2		Handle		Spoon	
Performance measure	d_p [px]	a_p [px]	d_p [px]	a_p [px]	d_p [px]	a_p [px]	d_p [px]	a_p [px]
Region growing	49	5895	62	9049	52	5149	22	1999
Closed-loop	3	320	6	447	7	316	8	364

5.3. Top-down image ROI definition through camera gaze orientation

The case of top-down image ROI definition takes into account available knowledge regarding the imaged scene. In the ROVIS architecture, this knowledge is represented by the contextual information regarding container objects, that is, they form relatively large rectangular shapes in the input image. The classification of objects in ROVIS has been explained in Chapter 3.2.1. Using this information and the robust boundary based segmentation algorithm from Chapter 4.2, the image ROI can be set on the container. In comparison to the previous case of bottom-up ROI definition, when only one object to be manipulated was bounded, in case of the top-down approach, the ROI will bound a container that can include more objects to be manipulated. It is important to mention that

the container detection method presented in this chapter is strictly used only for defining the image ROI. The 3D Position and Orientation (POSE) of the container, needed for manipulator path planning, is reconstructed using the SIFT based marker detection algorithm, as explained in Chapter 3.2.2.

The main problem with defining a ROI on container objects is that there are cases when the container is not present in the *Field of View* (FOV) of the camera. Also, it can happen that, although a part of the container is found in the imaged scene, a relatively large part is again out of the FOV. In order to cope with this problem, a gaze orientation system for the stereo camera has been proposed. The goal is to center the container object in the middle of the camera FOV.

The control of the camera's orientation belongs to the field of *active vision*, which deals with the methodologies of changing the parameters of a camera system (e.g. position, orientation, focus, zoom etc.) for facilitating the processing of visual data [93]. The adaptation of these parameters is performed in a visual feedback manner. The types of visual control structures used for this purpose are mainly classified into two distinctive approaches: *position-based* and *image-based*. Position-based active vision implies the 3D reconstruction of the imaged objects. This is a relative difficult problem due to the non-linearities of the transformations and the uncertainties of image processing systems. On the other hand, for the second case of image-based active vision, the usage of extracted image features for a direct control of camera parameters provides a way around the complexity of 3D reconstruction.

The algorithm presented in this chapter involves the automatic adaptation of the POSE of the camera system for localizing container objects. Once a container is detected and centered in the middle of the camera's FOV, the image ROI is set on it. The image-based active vision approach is used in designing the visual control system, whereas the boundary based segmentation method described in Chapter 4.2 is used for recognizing the containers.

5.3.1. Stereo camera head configuration in FRIEND

The global stereo camera used for environment understanding in the ROVIS system is mounted on a *Pan-Tilt Head* PTH unit placed on a special rack behind the user, above his head, as illustrated in Figure 3.15. The PTH is a servo-electric 2-DoF Schunk® robotic joint which provides to the stereo camera a coverage of a field of view corresponding to a maximum pan and tilt angles of $\pm 1080°$ (3 rotations) and $\pm 120°$, respectively. The resolution used for the encoders of the motors has a value of 4 [Arcsec/Inc] for the pan and 5 [Arcsec/Inc] for the tilt angle. Another representative characteristic of the chosen PTH is its angular velocity, which has a maximum value of 248 [°/sec] for pan and 356 [°/sec] for the tilt.

As explained in Chapter 3.2.2, the POSE of the stereo camera in Cartesian space is related to the "world" coordinate system W. The world coordinates is the reference coordinate

system of the FRIEND robot, located at the basis of the manipulator arm. The 3D virtual environment reconstructed in the World Model of FRIEND is related to this reference coordinate system. In Figure 5.13, the coordinate system of the stereo camera is represented with respect to W and a 3D reconstructed object in the environment. Since a stereo system is used, the camera has two coordinate frames attached, one for the left camera lens, C_L, and one for the right one, C_R. The PTH unit is described by the Pan and $Tilt$ coordinate frames, the first one providing rotation along the y axis, for the pan angle α, and the second one rotating around the x axis, for the tilt angle β. Knowing the transformations between the camera and the world and between the object and the camera, the calculated 3D pose of the imaged object can be related to W. This represents the calculation of the transformation from the robot's reference frame and the object, ${}^{W}_{Object}T$, needed for manipulator path planning and object grasping.

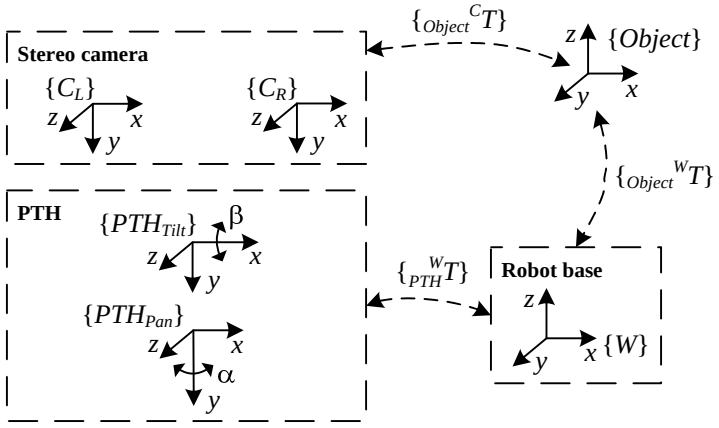


Figure 5.13.: Configuration of the stereo camera – PTH system and transformation of coordinates between the robot's world reference, camera lenses, PTH and a 3D reconstructed object.

One important component for 3D reconstruction is the recalibration process of the stereo camera when its orientation changes, that is, the recalculation of the projection matrices Q_L and Q_R, as in Equation 2.33. The projection matrices describe the relation between the camera and the reference coordinate system W. These matrices are composed of two sets of coefficients, the *intrinsic* parameters describing the manufacturing characteristics of the camera (e.g. focal length) and the *extrinsic* parameters, which describe the coordinate transformation from the camera to the reference coordinate system, ${}^{W}_{C_L}T$ and ${}^{W}_{C_R}T$. At the initialization of ROVIS, the projection matrices are calculated via camera calibration, as illustrated in Figure 3.6. The calibration procedure searches for a marker with a predefined POSE attached to the base of the FRIEND robot, that is the world coordinates W. Having in mind that the POSE of the marker is known a priori, the POSE of the stereo camera system can be calculated with respect to the reference coordinate system. At this initialization phase, the current pan α_{cal} and tilt β_{cal} angles of the PTH unit, also called calibration angles, are measured and stored in the World Model. Further, knowing

the calibration angles, the POSE of the PTH_{Pan} coordinate system, to which the stereo camera is rigidly attached, can be calculated:

$$ {}^{W}_{PTH_{Pan}}T = {}^{W}_{C_L}T \cdot {}^{C_L}_{PTH_{Tilt}}T(\beta_{cal}) \cdot {}^{PTH_{Tilt}}_{PTH_{Pan}}T(\alpha_{cal}), \tag{5.13} $$

where T represents coordinate transformation. As a convention, the PTH coordinate system is calculated through the left camera lens C_L.

The importance of the POSE of PTH_{Pan} is dependent on the recalculation of the projection matrices when the calibration marker is no longer in the FOV of the camera. In order to assure precise 3D reconstruction, the projection matrices are updated with every movement of the PTH unit, using the following relations:

$$ \begin{cases} {}^{W}_{C_L}T = {}^{W}_{PTH_{Pan}}T \cdot {}^{PTH_{Pan}}_{PTH_{Tilt}}T(\alpha) \cdot {}^{PTH_{Tilt}}_{C_L}T(\beta), \\ \\ {}^{W}_{C_R}T = {}^{W}_{PTH_{Pan}}T \cdot {}^{PTH_{Pan}}_{PTH_{Tilt}}T(\alpha) \cdot {}^{PTH_{Tilt}}_{C_R}T(\beta), \end{cases} \tag{5.14} $$

where α and β represent the current pan and tilt angles, respectively.

5.3.2. Image based visual feedback control

Visual control, or active vision, has been extensively investigated for the purpose of online adaptation of camera parameters [21, 49, 41]. In this thesis, visual control is used for centering the stereo camera's FOV on possible container objects present in the support scenarios of the FRIEND robot. In Figure 5.14, the block diagram of the active vision structure used to control the orientation of the camera system is shown.

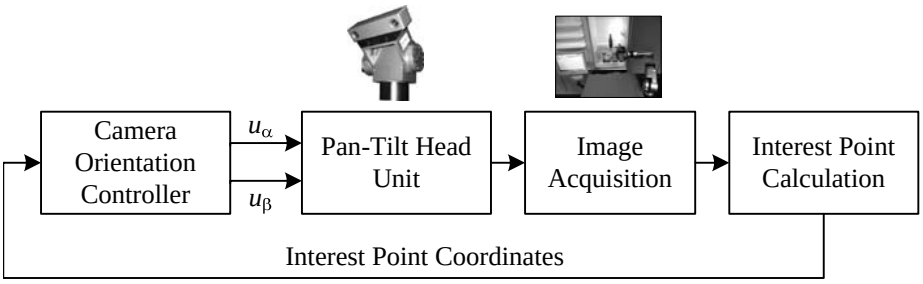


Figure 5.14.: Block diagram of the closed-loop camera gaze orientation system.

In the diagram from Figure 5.14, location of features on the 2D image plane are used in a feedback manner to adjust the orientation of the camera. Since image features are directly used in adapting the camera's orientation, there is no need for stereo image acquisition. Stereo images are normally used for reconstructing the full POSE of an object. From the calculated features, the actuator variables for the pan and tilt angles, $\{u_\alpha, u_\beta\}$, are to be determined. The features are used to compute a so-called interest point which is to be centered in the middle of the camera's FOV. In this case, the interest point is calculated

using image processing techniques and should not be confuzed with the interest point from Chapter 5.2, defined through user interaction. In the following, the interest point and the design of the visual controller will be explained.

Interest point definition through robust boundary based segmentation

One important part in the visual feedback loop from Figure 5.14 is the calculation of the interest point $pt_{int}(x, y)$ from which the control error is determined. Keeping in mind that the purpose of the presented algorithm is to fix the orientation of the camera on a container, proper image features have to be used for determining the location of the container in image.

The robust boundary based segmentation algorithm from Chapter 4.2 has been used for calculating the position of containers in the 2D image plane. The obtained rectangular shapes are combined with contextual knowledge regarding containers in order to determine the interest point $pt_{int}(x, y)$. The extra knowledge is represented by the fact that containers have a relatively large size in the image. $pt_{int}(x, y)$ is defined as the middle point of the largest detected rectangular object. Also, the size of the image ROI is set according to the size of the detected container.

In order to adapt the camera's orientation in a feedback manner, the container object has to be found in the FOV of the camera. This is needed for determining the interest point which represents the feedback variable of the visual controller. Before applying the visual control law, the gaze orientation system works in a so-called candidate search mode where a camera sweep is performed through the environment. The increment value added at each sweep step has a value of 5°. The search is used for detecting the best candidate for a container object. After the container object has been found, visual feedback control for camera gaze orientation is activated. The mentioned steps are illustrated in the flowchart from Figure 5.15.

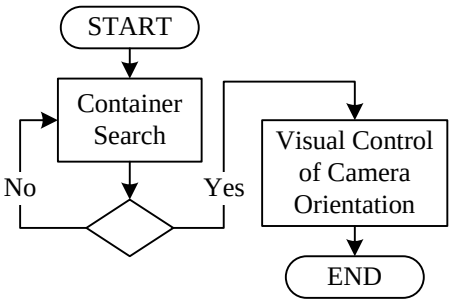


Figure 5.15.: Flowchart of the camera gaze orientation algorithm in ROVIS.

Modeling and controller tuning

In Figure 5.16, the control error that drives the pan α and tilt β angles of the PTH unit, calculated in 2D image plane, can be seen.

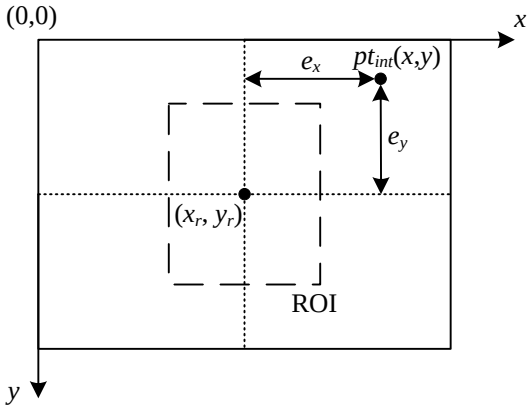


Figure 5.16.: Error in 2D image coordinate system between the calculated interest point $pt_{int}(x, y)$ and the reference image point (x_r, y_r).

The reference value, to which the interest point has to be driven, is the center of the camera, (x_r, y_r). As said before, $pt_{int}(x, y)$ is determined using the robust boundary based segmentation algorithm from Chapter 4.2. The parameters of the ROI are determined from the detected container, that is its 2D position, width and height. Based on the calculated container position, a visual feedback control method can be derived for adapting the viewing angles of the camera through the PTH system. The values of the actuator variables for the pan u_α and tilt u_β orientations are computed using a PI (Proportional-Integral) control law in velocity form:

$$\begin{bmatrix} u_\alpha(k) \\ u_\beta(k) \end{bmatrix} = \begin{bmatrix} u_\alpha(k-1) \\ u_\beta(k-1) \end{bmatrix} + \begin{bmatrix} K_{P\alpha} & 0 \\ 0 & K_{P\beta} \end{bmatrix} \cdot \begin{bmatrix} e_x(k) - e_x(k-1) \\ e_y(k) - e_y(k-1) \end{bmatrix} + \begin{bmatrix} K_{I\alpha} & 0 \\ 0 & K_{I\beta} \end{bmatrix} \cdot \begin{bmatrix} e_x(k) \\ e_y(k) \end{bmatrix}, \tag{5.15}$$

where k is the discrete time, $e_x = x_r - pt_{int}(x)$ and $e_y = y_r - pt_{int}(y)$ represent the control errors for the pan and tilt components, respectively. K_P and K_I are the proportional and integral gains, respectively. The relation between the two gains is given by $K_I = \frac{K_P}{T_I}$, where T_I is the integral time constant.

The final block diagram of the visual feedback control system for top-down image ROI definition, based on the control law from Equation 5.15, is illustrated in Figure 5.17.

At the center of the diagram from Figure 5.17 is the image processing chain used for calculating the interest point and subsequent the visual control error. For each component (pan and tilt, respectively), a feedback loop is derived based on the position of $pt_{int}(x, y)$ along the x axis of the image plane for the case of the pan α angle and, similarly, based on the position of $pt_{int}(x, y)$ along the y axis, for the case of the tilt β angle.

Since the control error is calculated in image pixels and the input to the pan and tilt modules represents radians, a conversion between the two measures had to be adopted. Knowing that the length of a pixel l_{px} for the used Bumblebee® camera has a value of

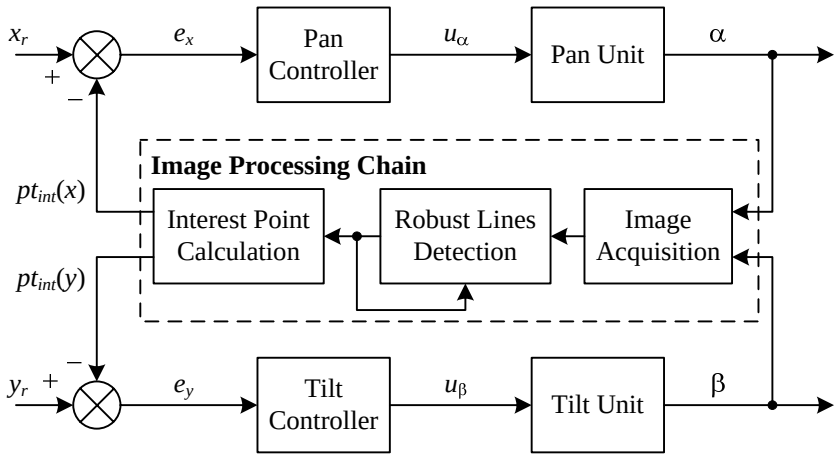


Figure 5.17.: Block diagram of proposed visual control structure for camera orientation and image ROI definition.

$l_{px} = 0.00465$mm and the camera's focal length is $f = 6$mm, the FOV covered by a pixel γ_{px} can be calculated as:

$$\gamma_{px} = 2 \cdot arctan\left(\frac{l_{px}}{2f}\right). \tag{5.16}$$

Using Equation 5.16, the covered FOV of a pixel has a value of $\gamma_{px} \approx 0.0444°$. By substitution, it turns out that a degree is represented in the FOV of the camera by a value of 22.52px. Keeping this mapping in mind, the control signals in pixels, u_x and u_y, can be easily converted into control signals in radians, u_α and u_β. These values are references to the pan and tilt modules, respectively.

The definition of the visual control law, is followed by the tuning of the visual controller. The goal of this procedure is to find the proper gains K_P and K_I that will assure good system performances, like stability, fast settling time, small steady-state error etc. [71]. In this thesis, the used tuning method is the so-called "trial-and-error" approach where different values of K_P and K_I are tried and evaluated. Finally, the values that provide the best performance results are chosen for on-line implementation in the robotic system. In Figure 5.18, different step responses of the proposed visual control system are presented. As can be seen, for a proper choice of proportional and integral gains ($K_P = 1$, $K_I = 0.6$) the system reaches a steady-state error and an adequate transient response.

5.3.3. Position based visual control in an intelligent environment

An *Intelligent Environment* is defined as a location (e.g. home, office, hospital etc.) equipped with different sensors and actuators linked together to form a system that can perform specific tasks. In the context of robot vision, an intelligent environment is defined as a location where different markers are placed in the scene in order to provide scene

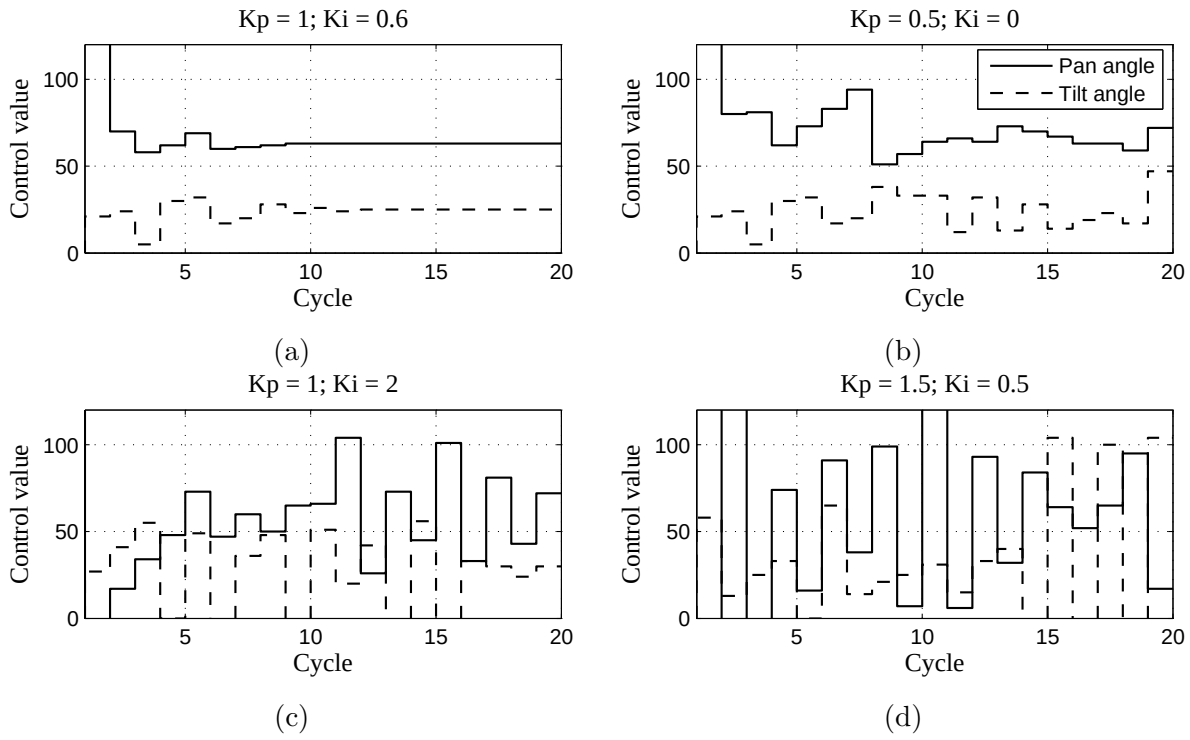


Figure 5.18.: Step responses of the proposed visual controller for different combinations of proportional K_P and integral K_I gains.

related information to a robot.

In the FRIEND system, natural markers are used to enrich the visual information of the robot. The algorithm used to detect the marker in the 2D image is based on the SIFT [57, 58, 59] feature detector, shortly explained in the introduction of Chapter 4. Details regarding the structure and implementation of the marker detection algorithm in ROVIS can be found in [10]. The principle of the method is to use a model image, the natural marker, to off-line train a classifier. Once the model image has been detected, its POSE can be reconstructed. Knowing the position of the model image and the 3D geometric structure of the container, the POSE of the imaged container can be reconstructed.

The centering of the container in the FOV of the camera is performed in a similar manner as in the case of the feature-based visual control structure presented above. Here, the feature-base control law is replaced with a position-base one. For the last one, the control error is calculated in 3D Cartesian space from the obtained POSE of the container and the orientation of the camera system.

Once the container object has been detected and centered in the FOV of the camera, the image ROI is calculated from the 3D virtual model of the container. This is done using the 3D to 2D mapping approach explained in Chapter 3.2. The resulting image region enclosing the container, in which objects to be manipulated are located, represents the image ROI.

6. Object recognition and 3D reconstruction in ROVIS

Classification and Position and Orientation (POSE) estimation of objects imaged under variable illumination conditions is a challenging problem in the computer vision community. In ROVIS, object recognition is dependent on the type of segmentation used, region or boundary based, and by how segmented images are evaluated. The difference between segmentation evaluation is especially visible in the choice of extracted features used for object classification and further 3D reconstruction. For example, if for region segmentation a set of object features are optimal for classification, for boundary segmentation they might not at all be the proper ones.

Having in mind these statements, the proper segmentation method used for recognizing an object to be manipulated in ROVIS is selected based on scenario context, as explained in Chapter 3. Namely, the robot knows what it expects to see (e.g. a bottle or a book). In this chapter, two types of object recognition methods used in ROVIS are detailed. Both approaches take into account the type of binary segmented image given as input. The output of classification represents the object's class and the 2D object feature points to be used for 3D reconstruction.

The structure of the object recognition and reconstruction chain used in ROVIS is presented in Figure 6.1. The overall goal of the components from the illustrated block diagram is to reliably extract the 3D POSE of objects to be manipulated so that a robust autonomous manipulator action is accomplished [73]. In ROVIS, container objects are localized using a SIFT based method, as explained in Chapter 3.2.1. 3D object reconstruction of objects to be manipulated is strongly dependent on the result of object recognition, that is, on the result of feature extraction and object classification. In the following, the modules from Figure 6.1 will be detailed, for both types of segmentations, respectively. Final 3D reconstruction is explained as an independent component which takes as input 2D object feature points and the determined object class.

6.1. Recognition of region segmented objects

The recognition of region segmented objects deals with analysis of visual data acquired by grouping pixels with similar characteristics. The objective of the image processing chain from Figure 6.1 is to parallel process left and right stereo images in order to extract feature points of objects to be manipulated and used them to reconstruct their POSE.

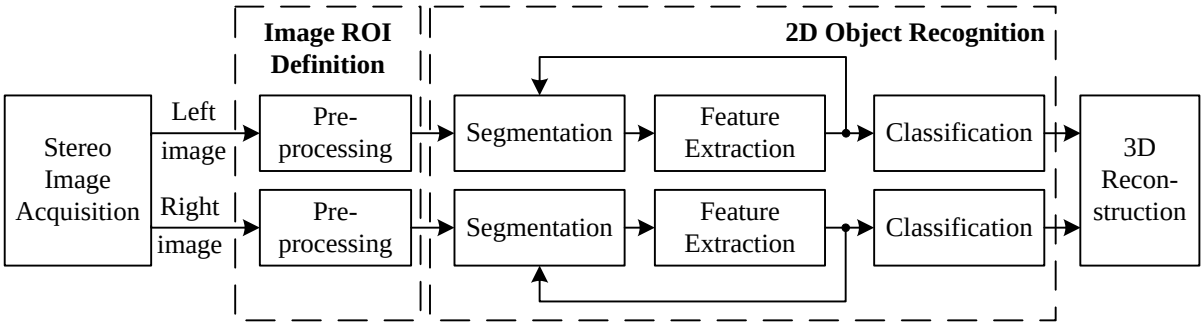


Figure 6.1.: Object recognition and 3D reconstruction chain in ROVIS.

For the considered case of region based segmented objects, the 2D feature points are two, represented by the object's top and bottom, or left and right margins, respectively:

$$\begin{cases} p_{Li} = (x_{Li}, y_{Li}), \\ p_{Ri} = (x_{Ri}, y_{Ri}), \qquad i = 1, 2. \end{cases} \tag{6.1}$$

where p_{Li} and p_{Ri} represent 2D region based segmented feature points in the left and right image, respectively. The ROVIS operations involved in the extraction of the 2D feature points from region based segmentation are detailed below.

6.1.1. Region based image processing operations

Image pre-processing

At this stage, the image ROI is defined with one of the robust methods presented in Chapter 5. As already mentioned, the choice of method is dependent on scenario context. The segmentation method applied on the image ROI is based on color or intensity data, depending on the amount of color information available in the image, as explained in Chapter 4.1. For obtaining color data, the HSI color model detailed in Chapter 2.3 is used. The acquired RGB images from the stereo camera are converted at the pre-processing stage into HSI images which are further fed to the robust segmentation module.

Robust region based segmentation

One important level of the object recognition chain from Figure 6.1 is the robust region based segmentation component described in Chapter 4.1. Based on the calculated image ROI, that is bounding only one object to be manipulated or a whole container, two types of input-output characteristics are obtained. The goal of the proposed closed-loop segmentation algorithm is to find the minimums of these characteristics, which correspond to good segmented objects. The minimums are calculated using an appropriate extremum seeking algorithm, as explained in Appendix A. In the following, for the sake of clarity, the examples show the variation of the hue angle with respect to the control variable

from Equation 4.6. The optimal saturation threshold is also calculated for each hue angle value, as detailed in Chapter 4.1. Based on the image ROI size, the two characteristics types are:

- *Input-output characteristic for a ROI bounding one object*: The optimal thresholding increment corresponds to one of the two minimums of the characteristic in Figure 6.2(a). In this case, one minimum represents the object of interest and the second minimum the background, or noise. The system is able to distinguish between the object and the background through a classification method presented later in this chapter. Since this ROI case corresponds to the bottom-up image ROI definition algorithm presented in Chapter 5.2, the segmented object is extracted directly from the result of ROI definition.
- *Input-output characteristic for multiple objects*: If multiple objects exists in the image ROI, as the case of the ROI definition algorithm from Chapter 5.3, the input-output characteristic will contain more minimas, as seen in the curve from Figure 6.2(b). Based on extremum seeking control, the minimas are determined. Each minima corresponds to an optimal segmented object, or background (noise). The objects in the obtained binary images are classified and used for 3D reconstruction.

In both ROI cases presented above, the success of segmentation is related on the proper choice of the operating ranges u_{low} and u_{high}, as detailed in Chapter 2.1. For characteristics as the one in Figure 6.2(b), a combination of different operating ranges and extremum seeking control are used to find the minimas.

Region based feature extraction

As discussed in Chapter 3.2, in ROVIS, features of objects to be manipulated are used for two purposes:

- classification of the obtained segmented shapes;
- 3D reconstruction of the recognized objects to be manipulated for manipulator arm path planning and object grasping.

In order to obtain the features of the segmented objects for classification and 3D reconstruction, their shapes are extracted from binary images using the contour extraction method from Chapter 2.5.

From the minimas in Figure 6.2, the segmentation results are a set of binary images B containing segmented objects to be manipulated along with noise viewed as black segmented pixels in the image:

$$B = \{b_0, b_1, ..., b_n\}, \tag{6.2}$$

where n represents the total number of binary images, related to the total number of minimas in the respective input-output characteristic. The binary images represent the

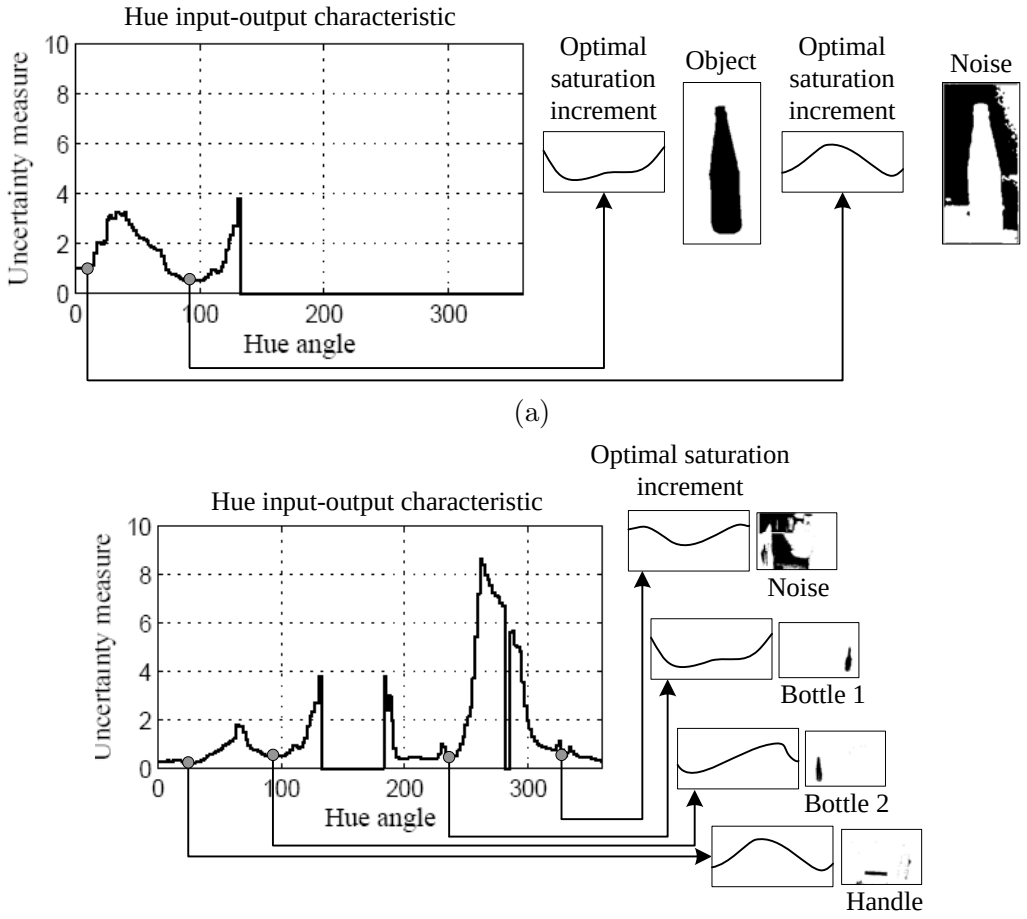


Figure 6.2.: The uncertainty measure I_m of segmented pixels vs. threshold increment u_h. One (a) and multiple (b) objects to be manipulated in the image ROI.

input to the feature extraction module. The task of this module is to extract the properties of the binary shapes found in the ROI of the segmented images. The binary shapes are modeled as polygonal approximations [30] of the objects contours:

$$C = \{c_0, c_1, ..., c_v\}, \tag{6.3}$$

where v represents the total number of detected contours. The number of shapes is strictly dependent on the number of objects to be manipulated in the image ROI. For each contour object features are calculated and stored in the features set as:

$$Y = \{y_i | i = 0, ..., d\}, \tag{6.4}$$

where y_i is a particular feature and d the total number of extracted features of an object, as follows:

- *Area* – the number of pixels in the extracted contour,
- *Bounding box* – the smallest rectangle containing the extracted contour,
- *Eccentricity* – division of the height of the bounding box of the contour to its width,
- *Width* – the width of the bounding box along each dimension,
- *Centroid* – the center of mass of the contour,
- *Central* and *invariant moments* of the extracted contour.
- *Color* $= u_{h\ opt}$ or *Intensity level* $= u_{i\ opt}$, depending on the used segmentation type.

Depending on the amount of color information in the image ROI, intensity or color segmentation is used to extract the objects. Based on the used method, either the color of an object is saved as a feature, for the case of color segmentation, or its intensity value for the case of intensity segmentation. Both features are given by the value of the optimal segmentation parameter described in Chapter 4.1, $u_{h\ opt}$ or $u_{i\ opt}$, respectively.

In order to save computation time, the set of contours C is filtered from noise using the area, perimeter and height features. It is known that, for objects to be manipulated in the FRIEND scenarios, these features can reside only on specific intervals. The new set of filtered contours is defined as:

$$C' = \{c'_0, c'_1, ..., c'_w\}, \quad C' \subseteq C, \tag{6.5}$$

where w is the number of filtered contours.

Region based object classification

The obtained features from the feature extraction module are further classified at the object classification stage. In ROVIS, object classification is based on invariant moments. The invariant moments from Equation 2.26 are calculated for each binary shape and combined in the Euclidean distance:

$$d_r = \sqrt{(I_{r1} - I_1)^2 + (I_{r2} - I_2)^2} \tag{6.6}$$

where I_i and I_{ri}, $i = 1, 2$, are, respectively, measured moments of shape and reference Hu moments of the object to be manipulated. The reference Hu moments, accessed from the system's World Model, are calculated from the so-called *ground truth image* which is obtained off-line by manual segmentation of the reference image until a "full" compact well shaped object region is obtained.

The extracted object data is used for object classification using the minimum distance classifier presented in Chapter 2.6. The classifier was trained using a number of sample images. The sample information was separated in three categories representing sets of training, validation and classification data.

6.1.2. Closed-loop improvement of object recognition

Due to different reflections and shadows during image acquisition it may happen that not all object pixels are segmented as foreground pixels even though the uniformly colored object is thresholded with the reference thresholding interval. Bearing this in mind and the definition of a good segmented image object as one which contains "full" and well shaped segmented object region, it turns out that the binary segmented image has to be improved to obtain the full, compact, object region. The goal of this improvement is to extract as precise as possible 2D object feature points used in 3D reconstruction. In ROVIS, for the case of region based segmented objects, the improvement is achieved using morphological dilation. The dilation operation increases the area of foreground pixels while covering the "holes" in the segmented regions. The dilation operator takes two inputs. One is the binary image to be dilated and the other is the so-called structuring element. The structuring element is nothing but a matrix consisting of 0's and 1's. The distribution of 1's determines the shape of the structuring element and the size of the pixel neighborhood that is considered during image dilation. The structuring element is shifted over the image and at each image pixel its elements are compared with the set of the underlying pixels according to some predefined operator. As a result, basically, a white background pixel turns to a black foreground pixel if there are black pixels in its neighborhood that are covered by the 1's of the structuring element. The effect of "filling" the segmented regions by dilation strongly depends on the shape and size of the structuring element as well as on the number of performed dilations.

The above presented object recognition chain is extended with an extra closed-loop algorithm for improving its performance. For this purpose, the feedback structure from Figure 6.3 is proposed, where the first closed-loop is responsible for robust image ROI segmentation and the second one for the improvement of the obtained segmented ROI.

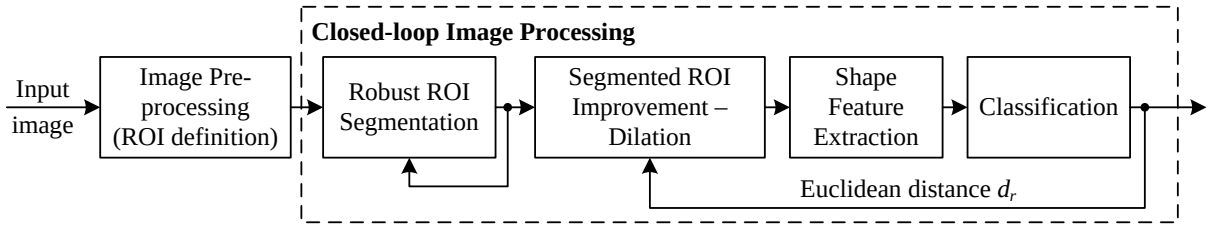


Figure 6.3.: Improved closed-loop region based object recognition in ROVIS.

Feedback control design

The purpose of the feedback loop from object classification is to "fill" the holes still present in the segmented object. In ROVIS, this is done by the included dilation closed-loop. The actuator (input) variable in this closed-loop is the height of the dilation structuring element. The controlled (output) variable is the shape of the segmented object of interest expressed by the Hu moments in Equation 2.26, i.e. by the Euclidean distance 6.6. Bearing

in mind that the Euclidean distance d_r measures the closeness of the segmented object Hu moments to their reference values, it turns out that the desired value of d_r is equal to zero. In order to investigate the input-output controllability of the dilation process, the input-output characteristic shown in Figure 6.4 is considered. As can be seen, it is possible to achieve the global minimum of d_r when changing the input variable across its operating range. In real-world applications, when dilating the segmented image corresponding to the image different from the reference one, the global minimum of d_r is not equal but it is very close to zero, as shown in Figure 6.4(b). Due to the input-output characteristic having global minimum, the control action based on the extremum searching algorithm, presented in Appendix A, is suggested, as shown in Figure 6.5.

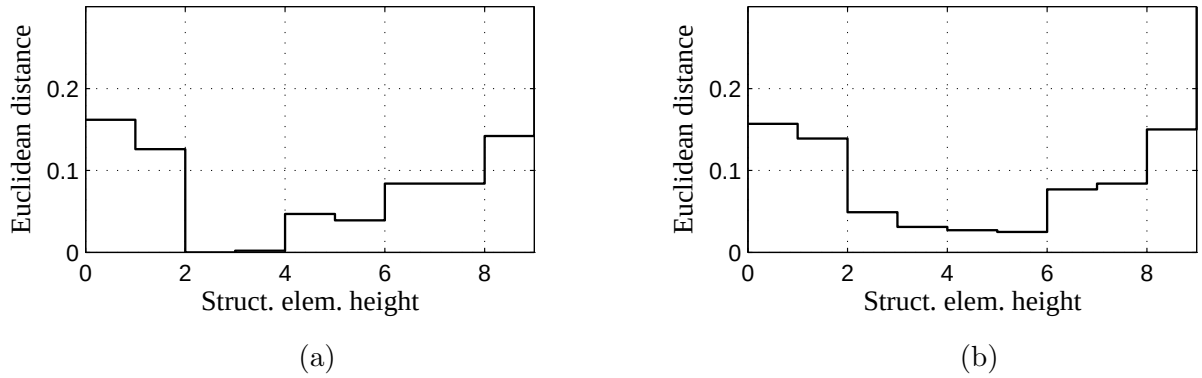


Figure 6.4.: Euclidean distance d_r vs. height of the dilation structuring element in the case of reference (a) and alternative (b) image.

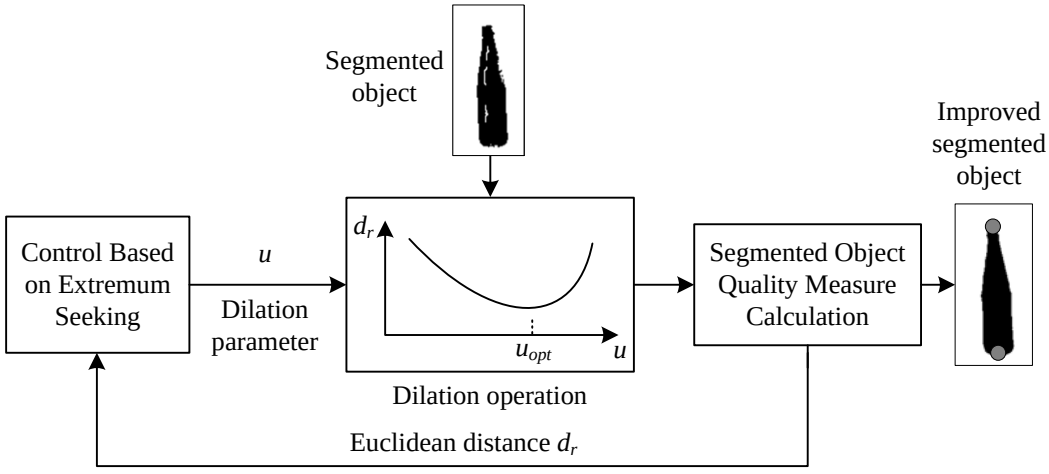


Figure 6.5.: Block diagram of feature extraction closed-loop for region based segmentation improvement.

From the improved binary images, the 2D feature points from Equation 6.1 can be extracted and further provided as input to the 3D reconstruction module, as explained later in this chapter. The gray circles from the output binary image in Figure 6.5 represent

the 2D feature points of a segmented bottle object.

6.1.3. Performance evaluation of 2D region based object recognition

In order to evaluate the effectiveness of the proposed closed-loop ROVIS object recognition method, its performance is compared with the performance of a traditional open-loop segmentation algorithm consisting of thresholding and dilation steps [30]. In contrast to the closed-loop method, which use feedback information on the processing results to adjust the processing parameters at particular processing level, the open-loop method uses constant reference parameters of both thresholding and dilation operation. These parameters are determined off-line, as discussed above, by manual thresholding and dilation of the reference image. On the other hand, the dilation closed-loop from Figure 6.3 is using the feedback information on object classification result to adjust the dilation parameter for improvement of binary image quality and feature extraction.

In order to evaluate the performances of the considered segmentation methods, the Euclidean distance in Equation 6.6 was used as performance criterion. A set of images of the FRIEND environment in the Activities of Daily Living (ADL) scenario were taken in different illumination conditions, ranging from 50lx to 900lx. Each captured image was segmented using the two tested segmentation methods. For each segmented image, the distance measure 6.6 was calculated after extracting Hu moments as relevant features of the segmented object region. The results are shown in Figure 6.6. As it can be seen, the Euclidian distance calculated from segmented images obtained by open-loop segmentation of bright images is almost equal to the desired zero value. This means that both considered segmentation methods give a good segmentation result for images captured in lighting conditions similar to the reference ones. This is an expected result even for the open-loop method since the used constant processing parameters are determined off-line by manual segmentation of the reference image. However, the performance of open-loop segmentation, in contrast to the ROVIS method, degrades significantly with the changing of the illumination conditions. However, as evident from Figure 6.6, even the proposed closed-loop method gave bad object segmentation results in images captured in very dark illumination condition. But, this bad result can be considered irrelevant for the robustness evaluation of the proposed method. Namely, applications of the rehabilitation robotic system FRIEND are considered to be indoor. For that reason the condition of dark illumination can be avoided since the system FRIEND operates always either in the very bright daily light conditions or in bright artificially light conditions.

6.2. Recognition of boundary segmented objects

Similar to the recognition of region segmented objects, the goal of recognizing boundary segmented shapes is to extract their 2D feature points by paralelly processing left and right stereo images, as illustrated in Figure 6.1. In case of boundary segmentation the

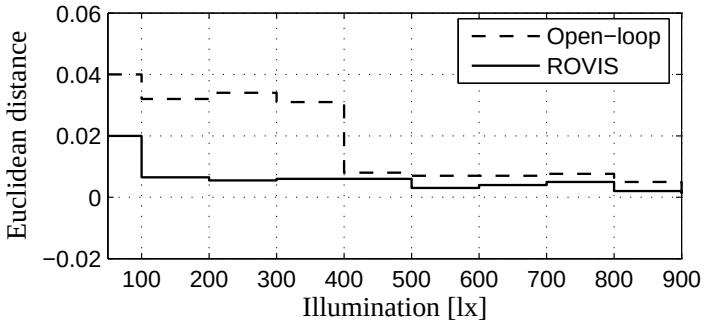


Figure 6.6.: Comparison between open-loop and ROVIS 2D region based object recognition methods.

number of extracted points are four in each image, defined as:

$$\begin{cases} p_{Li} = (x_{Li}, y_{Li}), \\ p_{Ri} = (x_{Ri}, y_{Ri}), \qquad i = 1, 2, 3, 4. \end{cases} \tag{6.7}$$

where p_{Li} and p_{Ri} represent 2D boundary based segmented object feature points in the left and right image, respectively. In this segmentation case, the four corners of a book. The boundary based object recognition methods used in the image processing chain from Figure 6.1 are detailed below.

6.2.1. Boundary based image processing operations

ROI definition

The first step in the object recognition chain from Figure 6.1 is image pre-processing, mainly represented by the definition of the image ROI. This process is performed through the top-down ROI definition method presented in Chapter 5.3. The container object, in this case the library desk, is centered in the camera's Field of View (FOV) and its boundaries detected, thus setting the image ROI on it.

Robust boundary segmentation and feature extraction

The detection of object boundaries is implemented using the robust boundary segmentation algorithm from Chapter 4.2. The objective of the method is to get precise locations of 2D object feature points needed by 3D reconstruction. These points are given as intersections of parallel and perpendicular lines that would form book objects. Based on the presented feedback optimization method, the optimal values of canny and hough transform are determined. The output of the algorithm is represented by the candidate solutions vector $N_{\#}$ which contains combinations of parallel and perpendicular lines, as explained in Chapter 4.2. These candidates have to be classified as real book objects or

noise by the classification method descried in the following.

As in the previous case of region segmented objects, different feature of the boundary contours are extracted for later use at the 3D object reconstruction stage. The features represent 2D object properties like its *area*, *bounding box*, *eccentricity*, *width*, *centroid*, *central* and *invariant moments*. These feature are extracted based on the object's contour calculated using the contour approximation method from Chapter 2.5.

Boundary based object classification

Because of image noise and texture, not all the candidate solutions in vector $N_{\#}$ represent real objects, that is books. The purpose of the classification procedure described here is to distinguish between spurious candidate solutions, called *negatives*, and real objects, named *positives*. This has been achieved with the help of the Minimum Distance Classifier, described in Chapter 2.6, and different extracted boundary object features, as follows:

- Relative object area $A_r = \frac{A_{obj}}{A_f}$,
- Eccentricity (object's width-to-height ratio) R_{wh},
- Relative number of object pixels $R_{px} = \frac{N_f}{N_b}$,

where A_{obj} represents the object area bounded with extracted lines and A_f the area of the whole image ROI. N_f and N_b correspond to the number of foreground and background pixels covered by the Hough lines in the binary segmented image, respectively. The above mentioned features were combined in two Euclidean distance measures forming the two positive and negative object classes:

$$D_{pos} = \sqrt{(A_r - a_1)^2 + (R_{wh} - b_1)^2 + (R_{px} - c_1)^2}, \tag{6.8}$$

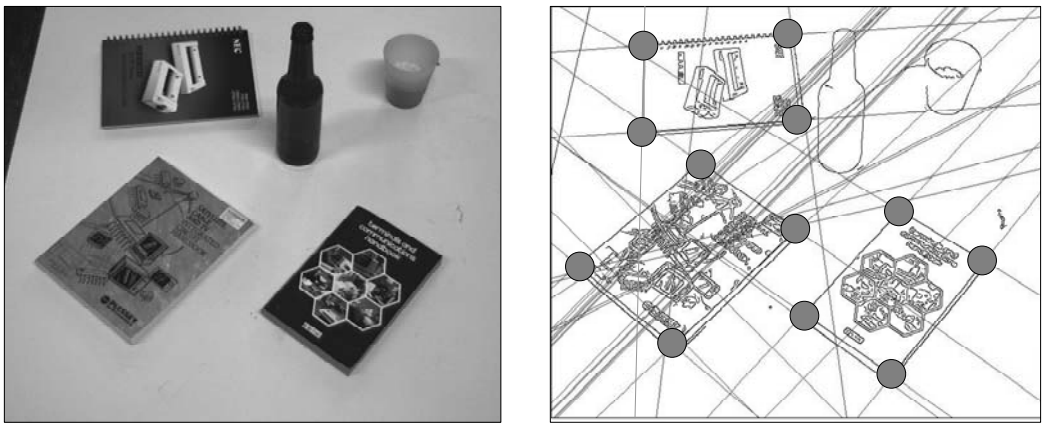
$$D_{neg} = \sqrt{(A_r - a_2)^2 + (R_{wh} - b_2)^2 + (R_{px} - c_2)^2}, \tag{6.9}$$

where the coefficients in Equations 6.8 and 6.9 have the values:

$$\begin{cases} a1 = 0.03;\ b1 = 0.9;\ c1 = 1.1; \\ a2 = 0.01;\ b2 = 0.7;\ c2 = 2.5. \end{cases} \tag{6.10}$$

The coefficients in Equation 6.10 have been heuristically determined using a number of 50 positive training samples and 50 negative ones. An object is considered positive and classified as a book if $D_{pos} > D_{neg}$. The intersections of the Hough lines of a detected object give the four object feature points 2D image. In Figure 6.7 the extraction of object feature points from recognized books in the FRIEND Library scenario can be seen.

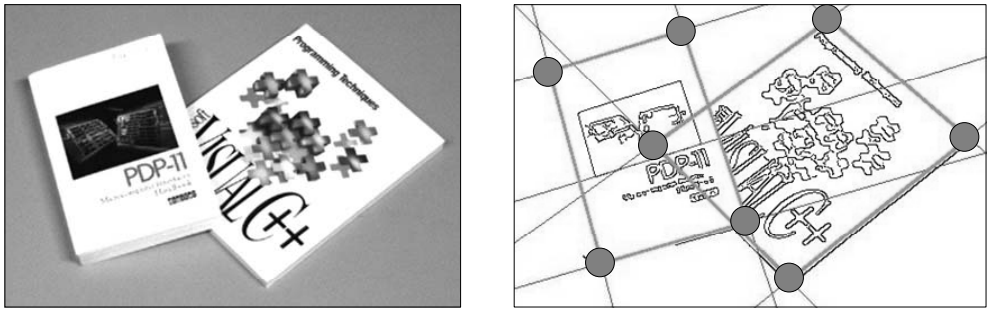
In real world environments, object grasping and manipulation based on visual information has to cope with object occlusion. One advantage of the Hough transform is that it can cope with partial occlusion [37], that is, candidate solutions are found although objects



(a) (b)

Figure 6.7.: Extraction of object feature points (a) Input image. (b) Recognized objects and their feature points.

overlap each other by a certain degree. In ROVIS, a boundary segmented object, partially occluded, is considered a positive if it is occluded by less than 30% of its own area. In Figure 6.8, the result of recognizing two books, one being partially occluded, can be seen.



(a) (b)

Figure 6.8.: Object recognition with partial occlusion. (a) Input image. (b) Recognized objects and their feature points.

6.2.2. Performance evaluation of 2D boundary based object recognition

As in previous performance evaluations conducted in this thesis, the comparison of the proposed boundary based object recognition algorithm is made according to its open-loop counterpart. In open-loop, the values of the parameters of boundary recognition are constant. The constant values are determined off-line from a reference image acquired in optimal illumination conditions. For the canny edge detector, the value $T_H = 101$ has been determined by manually obtaining the optimal segmentation result from the reference image. The threshold values of the hough transform accumulator has been similarly calculated with the value $T_{HG} = 62$.

The testing database consisted of a number of 60 sample images acquired in different illumination conditions ranging from 6lx to 2500lx. Examples of sample input images can be seen in Appendix D, Figure D.2.

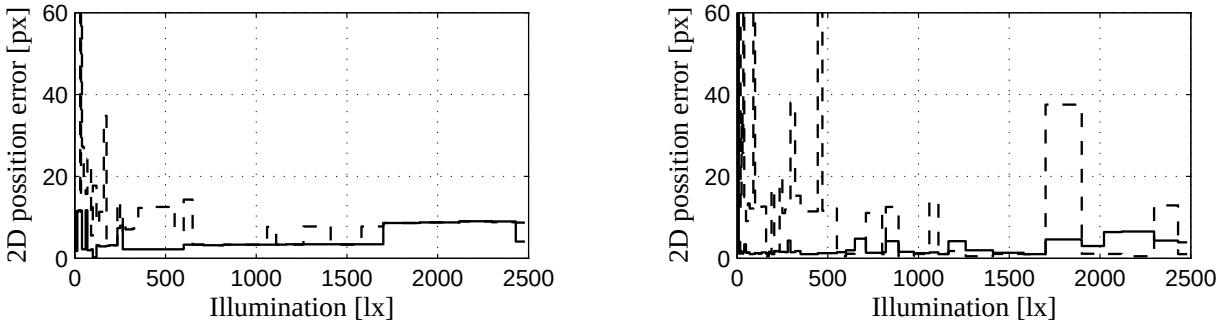
The position of the detected objects was compared with the real position of the objects, measured with respect to a marker placed in the middle of the library desk. Since the real world measurements are made in millimeters and the ones from 2D object recognition in pixels, a conversion between the two had to be set. Keeping in mind that the sample images represent the same scene in different illumination conditions, a mapping between pixel distances with respect to millimeters was adopted. The chosen metric maps one camera pixel to a value of 1.1mm.

The error between the real position of the object with respect to the one calculated via image processing was set using the following metric:

$$d_b = \frac{1}{4} \cdot \sum_{i=1}^{4} \sqrt{(x_{ri} - x_i)^2 + (y_{ri} - y_i)^2}, \tag{6.11}$$

where (x_{ri}, y_{ri}) represent the real world object coordinates, transformed in pixels, and (x_i, y_i) the object coordinates calculated using the open-loop and the proposed ROVIS boundary based object recognition algorithm, respectively. The points (x_i, y_i), with $i = 1 \ldots 4$, represent the four detected 2D feature points of the object, that is the four book corners needed for 3D reconstruction. The extracted points are measured in clockwise direction.

The diagrams in Figure 6.9 represent the error between real and calculated 2D object feature points over different illumination conditions. Ideally, the 2D position error should be zero. As can be seen from the diagrams, the ROVIS error is smaller than the open-loop one.



(a) (b)

Figure 6.9.: Evaluation of 2D boundary based object recognition. 2D position error, given by Equation 6.11, between calculated and real positions of objects in the image plane. (a,b) Two object cases.

6.3. 3D object reconstruction

The 3D reconstruction module of the image processing chain from Figure 6.1 deals with the mapping of real world recognized objects into a virtual environment suitable for planning the motion of the robotic arm [72]. In the overall architecture of FRIEND the *Mapped Virtual Reality* (MVR) system is used for such a task. Along with the recognized objects, a model of the robotic system is also present in the virtual reality world. As explained in Chapter 3.2.2, the calculated 3D positions of objects are placed in MVR according to the considered "world" coordinate system W, represented in FRIEND by the basis of the robotic arm. MVR includes an algorithm for calculating distances between objects. Using this algorithm the optimal kinematic configuration of the manipulator joints can be obtained and the motion of the arm planned. The MVR system is also used as a safety feature during the motion of the manipulator when the path of the arm is checked in real-time for collisions. In order to obtain such real-time computation, the objects are represented in the virtual space with three basic forms: sphere, cuboid and cylinder. A more complex object can be obtained by combinations of the three basic forms [72].

The characteristics of the 3D objects in MVR, that is spheres, cuboids and cylinders, are calculated from the extracted 2D object features. The 2D feature points used to reconstruct an object, defined in Equations 6.1 and 6.7, are obtained after classifying the object's primitive shape (e.g. bottle, glass, meal-tray, book etc.). The primitive shapes of the objects are stored in the World Model of the system. As an example, the feature points needed for reconstructing a bottle are its top and bottom points. Similar, for a book object, its four detected corners. In order to use the 2D feature points for 3D modeling, their 3D position has to be reconstructed. This reconstruction is performed using feature points from the left and right stereo images and the constraint of epipolar geometry [34]. The complete ROVIS procedure for 3D reconstruction of a point is detailed below.

Using reconstructed 3D feature points and 2D object features obtained at feature extraction level (e.g. object height and width, are, centroid, moments etc.), the 3D model and POSE of an object can be calculated and saved in the MVR [106]. In 3D Cartesian space, the position of an object is given by its attached reference frame, defined in homogeneous coordinates as:

$$O = (x, y, z, 1), \tag{6.12}$$

where x, y and z represent the object's 3D position O along the three axes of the Cartesian space. The object's reference frame O is calculated using the reconstructed 3D feature points. For a particular 3D point P, the reconstruction procedure is as follows. The relationship between a 3D P and its perspective 2D image projections (p_L, p_R) is given as:

$$\begin{cases} p_L = Q_L \cdot P, \\ p_R = Q_R \cdot P, \end{cases} \tag{6.13}$$

where where p_L and p_R represent 2D image feature point coordinates in left and right stereo images, respectively. Q_L and Q_R are the left and right camera projection matrices determined during ROVIS initialization at the Camera Calibration stage, as seen in Figure 3.6. Q_L and Q_R are defined in Equation 2.33 as the product of the intrinsic and the extrinsic camera parameters.

The 3D orientation of an object is given by the Euler angles which express the orientation of the attached object reference frame along the x, y and z axes:

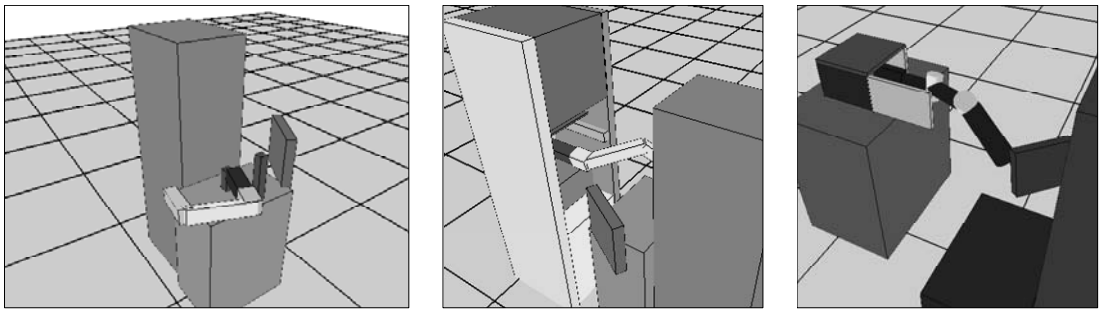
$$\begin{cases} \Phi = arctan\left(\frac{w_{31}}{w_{32}}\right), \\ \Theta = arccos(w_{33}), \\ \Psi = -arctan\left(\frac{w_{13}}{w_{23}}\right), \end{cases} \tag{6.14}$$

where Φ, Θ and Ψ give the object's orientation along the x, y and z axes, respectively. The coefficients w_{ij} from Equation 6.14 represent values of the object's *rotation matrix* [34] formed using the scalar component of each unit vector along the Cartesian axes:

$$Rot = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}. \tag{6.15}$$

In Equation 6.15, each column represent unit vectors along x, y and z, respectively. After the model of an object is calculated and saved, the manipulative skills can plan the movement of the manipulator and also determine the optimal grasping point of the reconstructed object.

In Figure 6.10, three examples of typical scenes from the FRIEND's ADL scenario can be seen. Figure 6.10(a) represents the MVR model of the FRIEND system and two reconstructed objects placed on its tray. In Figure 6.10(b,c) the interaction of FRIEND with two reconstructed container objects, a fridge and a microwave can be seen, respectively



(a) (b) (c)

Figure 6.10.: Reconstructions of objects of interest in the ADL support scenario of FRIEND. (a) Objects to be manipulated placed on the system's tray in front of the robot. (b,c) Manipulation of objects placed in different containers.

6.4. Performance evaluation of final 3D object reconstruction

The ultimate goal of the ROVIS system is reliable 3D reconstruction of objects. This reconstruction should assure correct 3D modeling of the FRIEND environment for the purpose of collision free path planning [73]. Therefore, the evaluation of the ROVIS effectiveness is done through the comparison of manually measured and automatically calculated 3D features of different objects to be manipulated, like bottles, mealtrays or books. Two different methods are used for the automatic calculation of the 3D features: 3D reconstruction based on the ROVIS system and 3D reconstruction based on a traditional open-loop system, with no included feedbacks at image processing level. In contrast to ROVIS, which uses feedback information to adjust image processing parameters, the open-loop method uses constant parameters. These parameters are determined offline by manually applying the image processing operations to the object image captured in reference illumination condition.

3D reconstruction in FRIEND ADL scenario

A scene from the FRIEND working scenario "serve a drink", shown in Figure D.1, was imaged in different illumination conditions ranging from 15lx to 570lx. This range of illumination corresponds to a variation of the light intensity from a dark room lighted with candles (15lx) to the lighting level of an office according to the European law UNI EN 12464 (500lx). Each captured image was processed using the two tested methods. The object feature points were extracted from each resulting segmented image and subsequently the 3D object coordinates were calculated and compared to the real measured 3D locations in order to calculate coordinates errors X_e, Y_e and Z_e. Also the width of the mealtray handle and the heights of the bottles were estimated based on extracted right and left end feature points, that is based on extracted top neck and bottom feature points, and compared to the real mealtray handle width, as error W_e, and, real bottle heights, as the error H_e, respectively. The comparison results are shown in Figures 6.11 and 6.12. The statistical measures of achieved error in experiments performed in different illumination conditions are given in Table 6.1.

Table 6.1.: Statistical results of open-loop vs. the ROVIS object recognition and 3D reconstruction approach for the ADL scenario.

	Open-loop				ROVIS			
	X_e [m]	Y_e [m]	Z_e [m]	W_e or H_e [m]	X_e [m]	Y_e [m]	Z_e [m]	W_e or H_e [m]
Max error	0.1397	0.0391	0.2357	0.1130	0.0049	0.0086	0.0029	0.0341
Mean	0.0331	0.0083	0.0121	0.0359	0.0024	0.0051	0.0017	0.0051
Std. dev.	0.0146	0.0071	0.0001	0.0282	0.0016	0.0021	0.0001	0.0044

As can be seen, the 3D object features calculated using the proposed vision architecture only differs slightly from the real coordinates over the whole considered illumination range, thus demonstrating the robustness of ROVIS. However, the 3D object features calculated

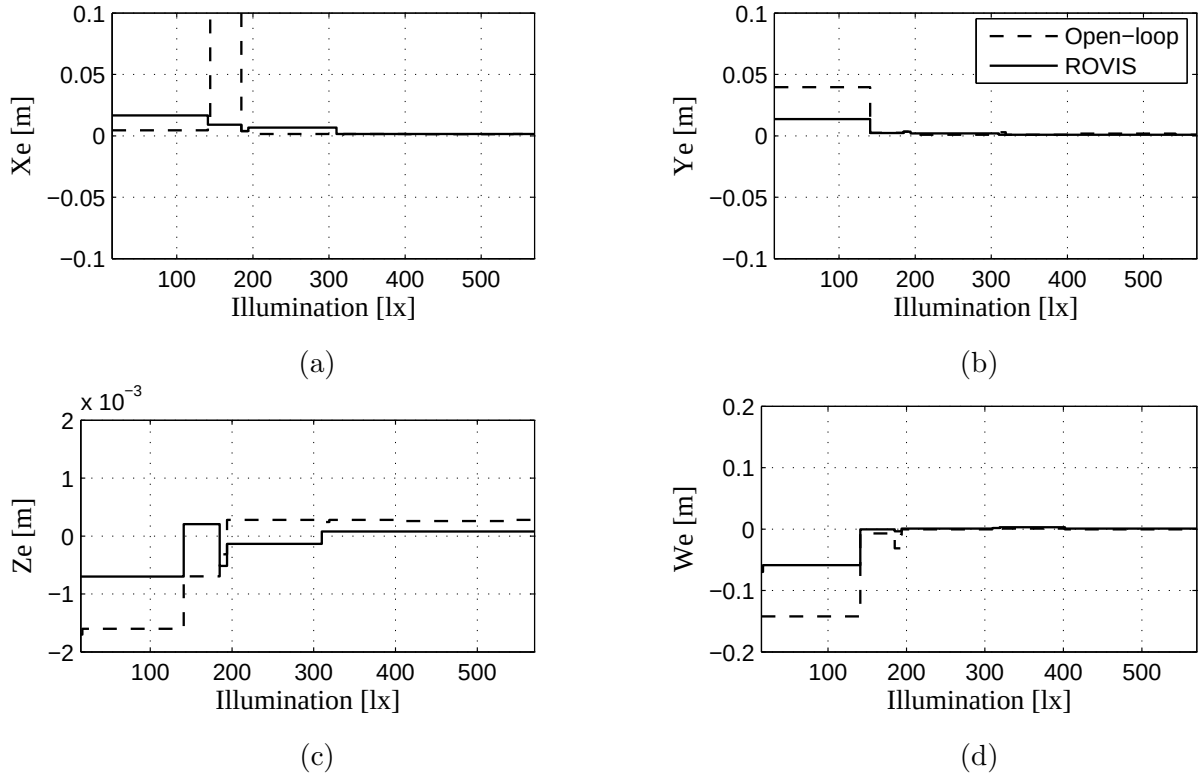


Figure 6.11.: Evaluation of 3D reconstruction. Difference between the real 3D object features and 3D features calculated from images resulting from the proposed ROVIS architecture and from the traditional open-loop processing chain. Case of mealtray object.

from the images resulting from the open-loop method, which uses constant parameters, significantly differs from the real coordinates for a number of illumination conditions which differ from reference illumination of 200lx. This indicates the importance of using feedback information on current results to adapt the image processing parameters to different environmental conditions. The image processing results are influenced not only by the intensity of illumination, but also by the position of the illuminant. This phenomenon can be observed in the sharp peeks from the diagrams from Figure 6.12. When the position of the illuminant was changed, the error of open-loop image processing results increased.

3D reconstruction in FRIEND Library scenario

The evaluation of final 3D reconstruction for the case of the Library scenario was performed in a similar manner as on the ADL case presented above, taking into consideration the relevant object feature points of books and the illumination interval [15lx, 1200lx]. Having into account the 3D shape of a book, that of a cuboid object, the object reference frame taken into consideration for 3D reconstruction is the middle point of the book, that is, the intersection of the book's major with minor axis. Once this point is successfully

calculated, the 3D reconstruction of the book can be made using its extracted width and height. Since the books are found on a flat table, parallel to the ground, only the orientation along the Z axis, that is the Ψ angle from Equation 6.14, was measured.

The final results for reconstructing books object 3D feature points can be seen in Figure 6.13, for the case of two books. As it can be seen from the diagrams and the statistical results from Table 6.2, in comparison to open-loop processing, the obtained ROVIS POSEs of books are precise enough for reliable object manipulation. As for the case of the ADL scenario, the calculated values of the objects poses are under a tolerance error which makes them reliable for visual guided grasping. The sharp transitions in Figure 6.13, for the case of the open-loop approach, are due to the nonlinearity of image processing. Using constant processing parameters, only for a small change in illumination, the position error of a book can increase considerably. This phenomenon is also illustrated in Figure 4.16, where the extraction of 2D feature points from open-loop boundary segmentation has large error when illumination changes.

Table 6.2.: Statistical results of open-loop vs. the ROVIS object recognition and 3D reconstruction approach for the case of the Library scenario.

	Open-loop				ROVIS			
	X_e [m]	Y_e [m]	Z_e [m]	Ψ_e [°]	X_e [m]	Y_e [m]	Z_e [m]	Ψ_e [°]
Max error	0.6294	0.2513	0.7881	34.2198	0.0127	0.0134	0.0069	6.3774
Mean	0.1291	0.1202	0.2493	14.9300	0.0101	0.0054	0.0038	4.6119
Standard deviation	0.0360	0.0229	0.0437	8.1207	0.0043	0.0045	0.0011	2.1551

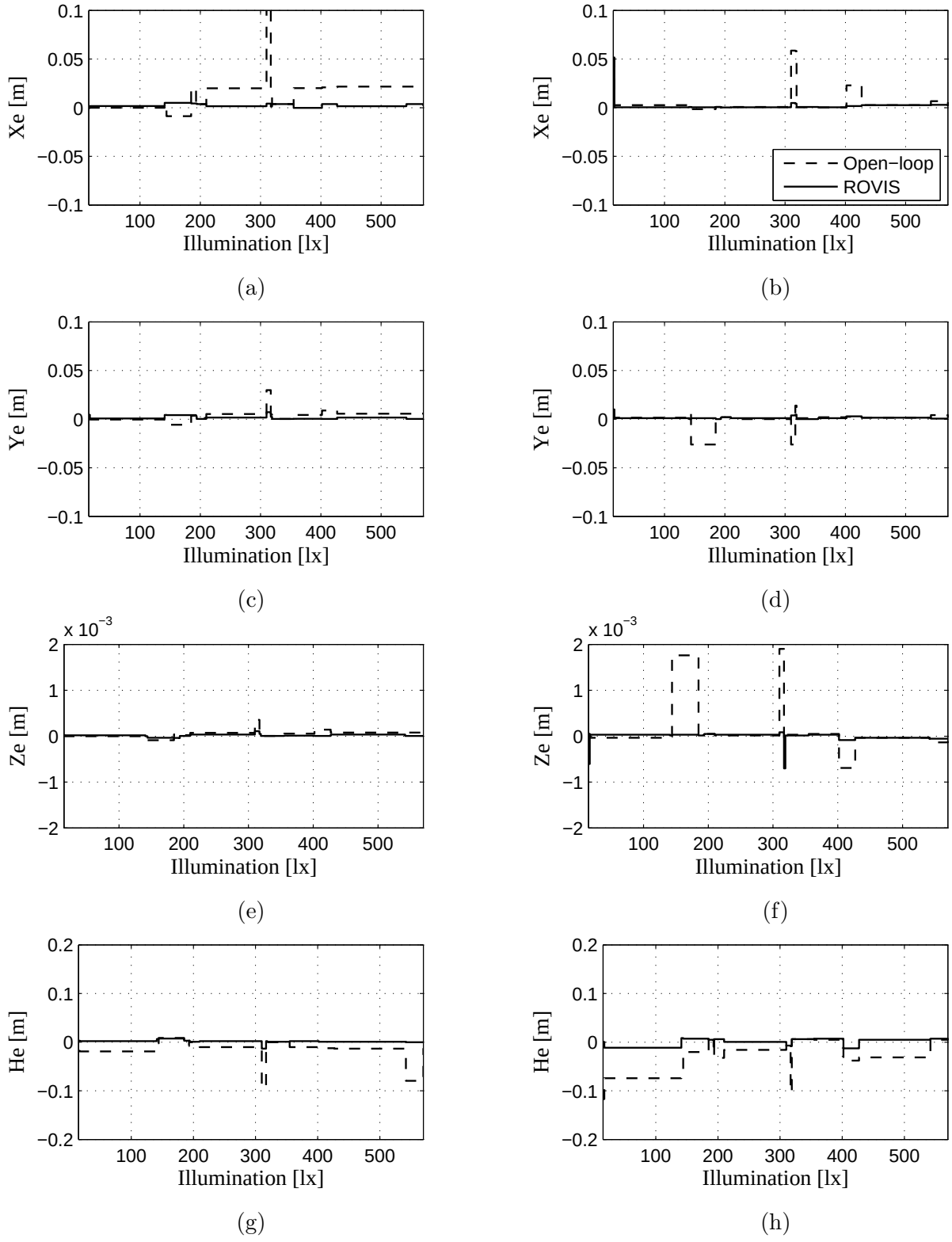


Figure 6.12.: Evaluation of 3D reconstruction. (a,c,e,g) Blue bottle object. (b,d,f,h) Green bottle object.

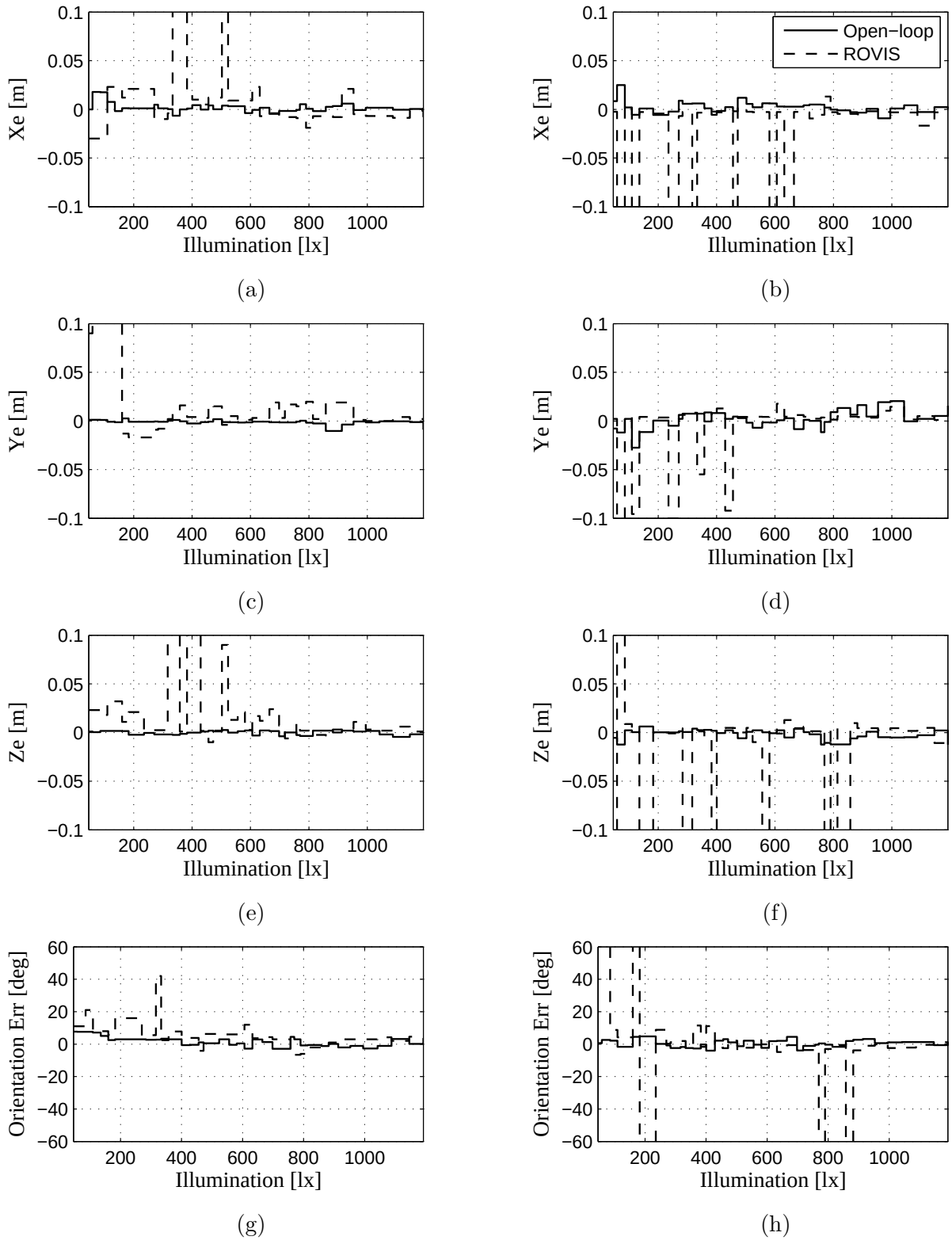


Figure 6.13.: Evaluation of 3D reconstruction. (a,c,e,g) Book one. (b,d,f,h) Book two.

7. Conclusions and outlook

In this thesis the novel vision system ROVIS for service robotics has been presented. The purpose of the robot vision architecture is to precisely reconstruct objects to be manipulated by a dexterous robotic arm. The precision of object detection plays a crucial role in the success of object manipulation. The core concepts of ROVIS are represented by the automatic calculation of an image ROI, where vision algorithms are applied, and inclusion of feedback mechanisms within image processing operations, as well as between various components of ROVIS. The goal of this inclusion has as purpose the improvement of the overall robustness of the robot vision system.

A still open problem in the robot vision community is robustness of vision algorithms against external influences, like variable illumination conditions and cluttered scenes. In ROVIS, this was achieved through the implementation of various feedback mechanisms at image processing levels. The objective of feedback is to automatically adjust the parameters of image processing methods in order to calculate their optimal working values according to current working conditions, that is current illumination. The principle of feedback was applied in Chapter 4 for the development of two robust image segmentation methods, region and boundary based, used as basic blocks for image ROI definition methods in Chapter 5 and the object recognition and 3D reconstruction chain from Chapter 6. The proposed closed-loop algorithms have been tested against their traditional open-loop counterparts. At the end of Chapter 6, the effectiveness of ROVIS has been demonstrated by an overall system test performed as a comparison between ROVIS and traditional open-loop 3D reconstruction.

The objects found in typical service robotics scenes, where ROVIS is used, were classified into object classes representing containers and objects to be manipulated. This classification plays an important role in the overall structure of the system since contextual knowledge is used in different amounts at different stages of processing. The image ROI definition concept in ROVIS is based on how humans visualize scenes, namely concentrating their attention on several interest regions in the environment. Combining this knowledge with the defined object classes, in Chapter 5 two approaches for closed-loop image ROI definition have been presented. The two ROI definition cases are developed around the so-called "bottom-up – top-down" framework. The first case, bottom-up, treats the definition of a ROI starting from an ill-defined user interest point in the input image with the objective of bounding the desired object of interest only. The second case, top-down, uses contextual knowledge regarding the imaged scene and the present objects to build an image ROI on containers found in the environment. On the calculated image ROI, the robust object recognition and 3D reconstruction chain from Chapter 6 is applied.

Depending on the class of the object that has to be grasped and handled by the manipulator, two approaches for recognizing and reconstructing objects to be manipulated have been set, that is for region and boundary segmented objects, respectively.

The ROVIS system presented in this thesis, along with the proposed robust image processing methods developed within, is intended to work as a basis platform for service robotic vision. From the hardware point of view, improvement of 3D object reconstruction can be achieved by incorporating in ROVIS range sensing, acquired using a 3D-ToF (Time-of-Flight) camera. The processing results from such a camera, which avoids the stereo correspondence problem, can be fusioned with data available from the global stereo camera, thus obtaining a better virtual picture of the robot's environment. Since the goal of ROVIS is reliable object reconstruction for the purpose of manipulator motion planning and object grasping, the inclusion of a so-called "eye-in-hand" camera mounted on the end effector of the manipulator arm can improve vision tasks, such as local object detection. The coordination between the global stereo and the "eye-in-hand" camera can be implemented in a visual servoing manner which may dynamically, on-line, readjust the motion of the arm with respect to changes in the imaged scene.

The ROVIS platform stands also as a basis for implementing a cognitive vision system for service robots. As discussed in the introduction, in recent years biologically motivated computer vision has become popular among vision and robotics scientists. ROVIS can be extended beyond its current capabilities through the inclusion of such structures. The advantages that could be gained are represented by a better adaptation of the vision system to new working environments and capabilities to learn new scenes from their context.

Further investigation in control for image processing applications represents also an extension of the work from this thesis, where feedback mechanisms have been successfully designed and tested at different levels of image processing.

Bibliography – own publications

[1] Saravana K. Natarajan, Dennis Mronga, and Sorin Mihai Grigorescu. Robust detection and 3D reconstruction of boundary detected objects. In *Methods and Applications in Automation (to be published)*. Shaker Verlag, GmbH, 2010.

[2] Sorin Mihai Grigorescu, Oliver Prenzel, and Axel Graeser. A new robust vision system for service robotics. In *Proc. of the 12th Int. Conf. on Optimization of Electrical and Electronic Equipments - OPTIM 2010 (to be published)*, Brasov, Romania, May 2010.

[3] Roko Tschakarow, Sorin Mihai Grigorescu, and Axel Graeser. FRIEND – a dependable semiautonomous rehabilitation robot. In *Proc. of the 2010 Int. Conf. ISR/ROBOTIK (to be published)*, Munich, Germany, June 2010.

[4] Torsten Heyer, Sorin Mihai Grigorescu, and Axel Graeser. Camera calibration for reliable object manipulation in care-providing system FRIEND. In *Proc. of the 2010 Int. Conf. ISR/ROBOTIK (to be published)*, Munich, Germany, June 2010.

[5] Sorin Mihai Grigorescu, Danijela Ristic-Durrant, and Axel Graeser. ROVIS: Robust machine vision for service robotic system FRIEND. In *Proc. of the 2009 Int. Conf. on Intelligent RObots and Systems*, St. Louis, USA, October 2009.

[6] Sorin Mihai Grigorescu, Danijela Ristic-Durrant, Sai Krishna Vuppala, and Axel Graeser. Closed-loop control in image processing for improvement of object recognition. In *Proc. of the 17th IFAC World Congress*, Seoul, Korea, July 2008.

[7] Sorin Mihai Grigorescu and Danijela Ristic-Durrant. Robust extraction of object features in the system FRIEND II. In *Methods and Applications in Automation*. Shaker Verlag, GmbH, 2008.

[8] Sorin Mihai Grigorescu and Axel Graeser. Robust machine vision framework for localization of unknown objects. In *Proc. of the 11th Int. Conf. on Optimization of Electrical and Electronic Equipments - OPTIM 2008*, Brasov, Romania, May 2008.

[9] Sai Krishna Vuppala, Sorin Mihai Grigorescu, Danijela Ristic-Durrant, and Axel Graeser. Robust color object recognition for a service robotic task in the system FRIEND II. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

Bibliography – references

[10] Faraj Alhwarin, Chao Wang, Danijela Ristic-Durrant, and Axel Graeser. Improved SIFT-features matching for object recognition. In *Proc. of the BCS Int. Academic Conference 2008 – Visions of Computer Science*, pages 179–190, 2008.

[11] Kartik B. Ariyur and Miroslav Krstic. *Real-Time Optimization by Extremum-Seeking Control*. John Wiley and Sons Ltd., New York, 2003.

[12] Tamim Asfour, Pedram Azad, Nikolaus Vahrenkamp, Alexander Regenstein, Kristian Bierbaum, Kai Welke, Joachim Schroeder, and Ruediger Dillmann. Toward humanoid manipulation in human-centred environments. *Robotics and Autonomous Systems*, 56(1):54–65, 2008.

[13] Michael Beetz, Tom Arbuckle, Armin B. Cremers, and Markus Mann. Transparent, flexible, and resource-adaptive image processing for autonomous service robots. In *Proc. of the 13th European Conf. on Artificial Intelligence ECAI 98*, pages 632–636. John Wiley and Sons Ltd., 1998.

[14] Zeungnam Bien, Myung-Jin Chung, Pyung-Hun Chang, Dong-Soo Kwon, Dae-Jin Kim, Jeong-Su Han, Jae-Hean Kim, Do-Hyung Kim, Hyung-Soon Park, Sang-Hoon Kang, Kyoobin Lee, and Soo-Chul Lim. Integration of a rehabilitation robotic system (KARES II) with human-friendly man-machine interaction units. *Autonomous Robots*, 16(2):165–191, November 2004.

[15] Cynthia Breazeal. Social interactions in HRI: The robot view. *IEEE Transactions on Systems, Man and Cybernetics–Part C: Applications and Reviews*, 34(2):181–186, May 2004.

[16] John F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[17] Yves Caron, Pascal Makris, and Nicole Vincent. Use of power law models in detecting region of interest. *Journal of the Pattern Recognition Society*, 40(9):2521–2529, September 2007.

[18] Yuanhao Chen, Long Zhu, Alan Yuille, and Hongjiang Zhang. Unsupervised learning of probabilistic object models (POMs) for object classification, segmentation, and recognition using knowledge propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1747–1761, October 2009.

[19] Roland T. Chin and Charles R. Dyer. Model-based recognition in robot vision. *ACM Computer Survey*, 18:67–108, 1986.

[20] Roberto Cipolla and Nick Hollinghurst. Visually guided grasping in unstructured

environments. *Robotics and Autonomous Systems*, 19:19–337, 1997.

[21] Peter I. Corke. *Visual control of robots: high-performance visual servoing*. John Wiley, 1996.

[22] Angel P. del Pobil, Mario Prats, Rafael Ramos-Garijo, Pedro J. Sanz, and Enric Cervera. The UJI librarian robot: An autonomous service application. In *Proc. of the 2005 IEEE Int. Conf. on Robotics and Automation. Video Proceedings*, Barcelona, Spain, 2005.

[23] Munjal Desai and Holly A. Yanco. Blending human and robot inputs for sliding scale autonomy. In *Proc. of the 14th IEEE Int. Workshop on Robot and Human Interaction Communication*, August 2005.

[24] Zacharay Doods, Martin Jaegersand, Greg Hager, and Kentaro Toyoma. A hierarchical vision architecture for robotic manipulation tasks. In Henrik I. Christensen, editor, *ICVS*, volume 1542 of *Lecture Notes in Computer Science*. Springer, 1999.

[25] Claire Dune, Christophe Leroux, and Eric Marchand. Intuitive human interaction with an arm robot for severely handicapped people - a one click approach. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

[26] Marc Ebner. *Color Constancy*. John Wiley and Sons Ltd., West Sussex, England, 2007.

[27] Astrid Franz, Ingwer C. Carlsen, and Steffen Renisch. An adaptive irregular grid approach using SIFT features for elastic medical image registration. In *Bildverarbeitung fuer die Medizin 2006*, pages 201–205. Springer Berlin Heidelberg, May 2006.

[28] Simone Frintrop. *VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search*. PhD thesis, Fraunhofer Institute for Autonomous Intelligent Systems (AIS), Berlin Heidelberg, 2006.

[29] Simone Frintrop, Erich Rome, Andreas Nuechter, and Hartmut Surmann. A bimodal laser-based attention system. *Computer Vision and Image Understanding*, 100(1):124–151, 2005.

[30] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice-Hall, New Jersey, 2007.

[31] Axel Graeser and Danijela Ristic-Durrant. Feedback structures as a key requirement for robustness: Case studies in image processing. In Alfons Schuster, editor, *Robust Intelligent Systems*. Springer-Verlag London Ltd, 2008.

[32] Matthias Hans and Birgit Graf. Robotic home assistant Care-O-bot II. In *Advances in Human-Robot Interaction*, volume 14, pages 371–384. Springer Berlin Heidelberg, July 2004.

[33] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proc. of the 4th Alvey Vision Conf.*, pages 147–151, 1988.

[34] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[35] Hans-Peter Hoffmann. SysML-based systems engineering using a model-driven development approach. *White Paper, Telelogic*, 2008.

[36] Lothar Hotz, Bernd Neumann, and Kasim Terzic. High-level expectations for low-level image processing. In *KI 2008: Advances in Artificial Intelligence*. Springer-Verlag Berlin Heidelberg, 2008.

[37] Paul V.C. Hough. Method and means for recognizing complex patterns. US Patent 3969654, 1962.

[38] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, pages 179–187, 1962.

[39] Seth Hutchinson, Gregory D. Hager, and Peter I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5), October 1996.

[40] Shengbing Jiang and Ratnesh Kumar. Decentralized control of discrete event systems with specializations to local control and concurrent systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 30:653–660, 2000.

[41] Erik Valdemar Cuevas Jimenez, Daniel Zaldivar Navarro, and Raul Rojas. *Intelligent Active Vision Systems for Robots*. Cuvillier Verlag, Goettingen, 2007.

[42] Ulrich Kaufmann, Gerd Mayer, Gerhard Kraetzschmar, and Guenther Palm. Visual robot detection in robocup using neural networks. In *RoboCup 2004: Robot Soccer World Cup VIII*, pages 262–273. Springer Berlin Heidelberg, March 2005.

[43] Vojislav Kecman. *Support Vector Machines, Neural Networks and Fuzzy Logic Models: Learning and Soft Computing*. MIT Press, Cambridge, MA, 2001.

[44] Dae-Jin Kim, Ryan Lovelett, and Aman Behal. An empirical study with simulated ADL tasks using a vision-guided assistive robot arm. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, pages 504–509, Kyoto, Japan, June 2009.

[45] Rex Klopfenstein Jr. Data smoothing using a least squares fit C++ class. *Instrumentation, Systems, and Automation (ISA) Transactions*, 37:3–19, 1998.

[46] Danica Kragic and Marten Bjoerkman. Strategies for object manipulation using foveal and peripheral vision. In *Proc. of the 4th IEEE Int. Conf. on Computer Vision Systems ICVS*, New York, USA, 2006.

[47] Danica Kragic, Marten Bjoerkman, Henrik I. Christensen, and Jan-Olof Eklundh. Vision for robotic object manipulation in domestic settings. *Robotics and Autonomous Systems*, 52(1):85–100, 2005.

[48] Danica Kragic and Henrik I. Christensen. A framework for visual servoing. In *Computer Vision Systems*, pages 345–354. Springer-Verlag Berlin Heidelberg, January 2003.

[49] David J. Kriegman, Gregory D. Hager, and Stephen A. Morse. *The Confluence of*

Vision and Control. Springer Verlag London, 1998.

[50] Miroslav Krstic and Hsin-Hsiung Wang. Stability of extremum seeking feedback for general nonlinear dynamic systems. *Automatica*, 2000(36):596–601, 2000.

[51] Oliver Lang. *Bildbasierte Roboterregelung mit einer am Greifer montierten Zoomkamera (in German)*. PhD thesis, Bremen University, Institute of Automation, Bremen, Germany, August 2000.

[52] Christophe Leroux, Isabelle Laffont, Nicolas Biard, Sophie Schmutz, Jean Francois Desert, Gerard Chalubert, and Yvan Measson. Robot grasping of unknown objects, description and validation of the function with quadriplegic people. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

[53] Ramon Leyva, Corinne Alonso, Isabelle Queinnec, Angel Cid-Pastor, Denis Lagrange, and Martinez-Salamero. MPPT of photovoltaic systems using extremum-seeking control. *IEEE Transactions on Aerospace and Electronic Systems*, 42(1):249–258, January 2006.

[54] Yaoyu Li, Mario A. Rotea, George T.-C. Chiu, Luc G. Mongeau, and In-Su Paek. Extremum seeking control of a tunable thermoacoustic cooler. *IEEE Transactions on Control Systems Technology*, 13(4):527–536, July 2005.

[55] Zhe Li, Sven Wachsmuth, Jannik Fritsch, and Gerhard Sagerer. Manipulative action recognition for human-robot interaction. In *Vision Systems: Segmentation and Pattern Recognition*, pages 131–148. I-Tech Education and Publishing, June 2007.

[56] Freek Liefhebber and Joris Sijs. Vision-based control of the Manus using SIFT. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

[57] David G. Lowe. Object recognition from local scale-invariant features. In *Int. Conf. on Computer Vision*, pages 1150–1157, Corfu, Greece, September 1999.

[58] David G. Lowe. Local feature view clustering for 3D object recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 682–688, Kauai, Hawaii, December 2001.

[59] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[60] Thorsten Lueth, Darko Ojdanic, Ola Friman, Oliver Prenzel, and Axel Graeser. Low level control in a semi-autonomous rehabilitation robotic system via a brain-computer interface. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

[61] Christian Martens. *Teilautonome Aufgabenbearbeitung bei Rehabilitationsrobotern mit Manipulator (in German)*. PhD thesis, Bremen University, Institute of Automation, Bremen, Germany, December 2004.

[62] Christian Martens. Task oriented programming of service-robots on the basis of

process-structures. In *Proc. of the 26th Colloquium of Automation*, pages 45–56, Salzhausen, Germany, November 2005.

[63] Christian Martens, Oliver Prenzel, Johannes Feuser, and Axel Graeser. MASSiVE: Multilayer architecture for semiautonomous service-robots with verified task execution. In *Proc. of the 10th Int. Conf. on Optimization of Electrical and Electronic Equipments - OPTIM 2006*, Brasov, Romania, May 2006.

[64] Christian Martens, Nils Ruchel, Oliver Lang, Oleg Ivlev, and Axel Graeser. A FRIEND for assisting handicapped people. *IEEE Robotics and Automation Magazine*, pages 57–65, March 2001.

[65] Randy Miller. Practical UML: A hands-on introduction for developers. *White Paper, Borland Developer Network*, April 2003.

[66] Majid Mirmehdi, Phil L. Palmer, Josef Kittler, and Homam Dabis. Feedback control strategies for object recognition. *IEEE Transactions on Image Processing*, 8(8):1084–1101, 1999.

[67] Melanie Mitchell. *An introduction to genetic algorithms.* MIT Press, Cambridge, MA, USA, 1996.

[68] Dimitris A. Mitzias and Basil G. Mertzios. A neural multiclassifier system for object recognition in robotic vision applications. *Journal of Imaging Measurement Systems*, 36(3):315–330, October 2004.

[69] Danny R. Moates and Gary M. Schumacher. *An Introduction to Cognitive Psychology.* Wadsworth Publishing Company, Inc., Belmont, California, 1980.

[70] Dinesh Nair, Lothar Wenzel, Alex Barp, and Afreen Siddiqi. Control strategies and image processing. In *Proc. of the 7th Int. Symposium on Signal Processing and its Applications*, pages 557–560, 2003.

[71] Katsuhiko Ogata. *Modern Control Engineering.* Prentice-Hall, London, UK, 2002.

[72] Darko Ojdanic. *Using Cartesian Space for Manipulator Motion Planning – Application in Service Robotics.* PhD thesis, Bremen University, Institute of Automation, Bremen, Germany, March 2009.

[73] Darko Ojdanic and Axel Graeser. Improving the trajectory quality of a 7 DoF manipulator. In *Proc. of the Robotik Conf.*, Munich, Germany, 2008.

[74] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, January 1979.

[75] Lucas Paletta, Erich Rome, and Hilary Buxton. Attention architectures for machine vision and mobile robots. In *Neurobiology of Attention*, pages 642–648. Elsevier, Inc., 2005.

[76] Jing Peng and Bir Bahnu. Closed-loop object recognition using reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2):139–154, February 1998.

[77] Wolfgang Ponweiser, Markus Vincze, and Michael Zillich. A software framework to

integrate vision and reasoning components for cognitive vision systems. *Robotics and Autonomous Systems*, 52(1):101–114, July 2005.

[78] Oliver Prenzel. Semi-autonomous object anchoring for service-robots. In *Methods and Applications in Automation*, pages 57–68. Shaker Verlag, GmbH, 2005.

[79] Oliver Prenzel. *Process Model for the Development of Semi-Autonomous Service Robots*. PhD thesis, Bremen University, Institute of Automation, Bremen, Germany, 2009.

[80] Oliver Prenzel, Christian Martens, Marco Cyriacks, Chao Wang, and Axel Graeser. System controlled user interaction within the service robotic control architecture MASSiVE. *Robotica, Special Issue*, 25(2), March 2007.

[81] Claudio M. Privitera and Lawrence W. Stark. Algorithms for defining visual region-of-interest: Comparison with eye fixations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):970–982, September 2000.

[82] R. Rakesh, P. Chaudhuri, and C.A. Murthy. Thresholding in edge detection: A statistical approach. *IEEE Transactions on Image Processing*, 13(7), July 2004.

[83] Danijela Ristic. *Feedback Structures in Image Processing*. PhD thesis, Bremen University, Institute of Automation, Bremen, Germany, April 2007.

[84] Danijela Ristic and Axel Graeser. Performance measure as feedback variable in image processing. *EURASIP Journal an Applied Signal Processing*, 2006(12), 2006.

[85] Danijela Ristic, Ivan Volosyak, and Axel Graeser. Feedback control in image processing. *atp international-automation technology in practice*, 2005(1):61–70, 2005.

[86] Danijela Ristic, Sai Krishna Vuppala, and Axel Graeser. Feedback control for improvement of image processing: An application of recognition of characters on metallic surfaces. In *Proc. of the 4th IEEE Int. Conf. on Computer Vision Systems*, New York, USA, January 2006.

[87] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Close-range scene segmentation and reconstruction of 3d point cloud maps for mobile manipulation in domestic environments. In *Proc. of the 2009 Int. Conf. on Intelligent RObots and Systems*, St. Louis, USA, October 2009.

[88] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56:927–941, 2008.

[89] Rolf D. Schraft, Evert Helms, Matthias Hans, and Stefan Thiemermann. Man-machine-interaction and co-Operation for mobile and assisting robots. In *Proc. of the Engineering of Intelligent Systems Conf. EIS 2003*, Madeira, Portugal, 2004.

[90] Sebastian Schuon, Christian Theobalt, James Davis, and Sebastian Thrun. High-quality scanning using time-of-flight depth superresolution. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1–7, Anchorage, USA, June 2008.

[91] Stephen Se, David Lowe, and Jim Little. Vision-based mobile robot localization and

mapping using scale-invariant features. In *Proc. of the 2001 IEEE Int. Conf. on Robotics and Automation, ICRA 2001*, pages 2051–2058, Seoul, Korea, May 2001.

[92] Michael Seelinger, John-David Yoder, Eric T. Baumgartner, and Steven B. Skaar. High-precision visual control of mobile manipulators. *IEEE Transactions on Robotics and Automation*, 18(6), December 2002.

[93] Rajeev Sharma. Role of active vision in optimizing visual feedback for robot control. In *The Confluence of Vision and Control*. Springer-Verlag London, 1998.

[94] Mohan Sridharan and Peter Stone. Color learning and illumination invariance on mobile robots: A survey. *Robotics and Autonomous Systems*, 57(6):629–644, June 2009.

[95] Bjarne Stroustrup. *The C++ Programming Language: Special Edition*. Addison Wesley, 1997.

[96] Motoki Takagi, Yoshiyuki Takahashi, Shinichiro Yamamoto, Hiroyuki Koyama, and Takashi Komeda. Vision based interface and control of assistive mobile robot system. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

[97] Geoffrey Taylor and Lindsay Kleeman. *Visual Perception and Robotic Manipulation*. Springer-Verlag, Heidelberg, 2006.

[98] James Trefil, editor. *Encyclopedia of Science and Technology*. McGraw-Hill, 2001.

[99] Katherine M. Tsui and Holly A. Yanco. Human-in-the-loop control of an assistive robot arm. In *Proc. of the Workshop on Manipulation for Human Environments, Robotics: Science and Systems Conf.*, August 2006.

[100] Katherine M. Tsui and Holly A. Yanco. Simplifying wheelchair mounted robotic arm control with a visual interface. In *AAAI Spring Symposium on Multidisciplinary Collaboration for Socially Assistive Robots*, March 2007.

[101] Diana Valbuena, Marco Cyriacks, Ola Friman, Ivan Volosyak, and Axel Graeser. Brain-computer interface for high-level control of rehabilitation robotic systems. In *Proc. of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR 2007*, Noordwijk, Netherlands, June 2007.

[102] Steve Vinoski and Michi Henning. *Advanced CORBA Programming with C++*. Addison Wesley, 2004.

[103] Ivan Volosyak. Farbenbasierte objektdetektion in der service-robotik (in german). In *Proc. of the 24th Colloquium of Automation*, pages 62–72, Salzhausen, Germany, November 2003. Shaker Verlag, GmbH.

[104] Ivan Volosyak, Oleg Ivlev, and Axel Graeser. Rehabilitation robot FRIEND II - the general concept and current implementation. In *Proc. of the IEEE 9th Int. Conf. on Rehabilitation Robotics ICORR 2005*, pages 540–544, Chicago, USA, 2005.

[105] Ivan Volosyak, Olena Kouzmitcheva, Danijela Ristic, and Axel Graeser. Improvement of visual perceptual capabilities by feedback structures for robotic system

FRIEND. *IEEE Transactions on Systems, Man and Cybernetics: Part C*, 35(1):66–74, 2005.

[106] Sai Krishna Vuppala and Axel Graeser. An approach for tracking the 3d object pose using two object points. In *Proc. of the Int. Conf. on Vision Systems ICVS*, pages 261–270, Santorini, Greece, 2008.

[107] Wei Wang, Houxiang Zhang, Wenpeng Yu, and Jianwei Zhang. Docking manipulator for a reconfigurable mobile robot system. In *Proc. of the 2009 Int. Conf. on Intelligent RObots and Systems*, St. Louis, USA, October 2009.

[108] Michael Wirth, Matteo Fraschini, Martin Masek, and Michel Bruynooghe. Performance evaluation in image processing. *EURASIP Journal on Applied Signal Processing*, 2006(12), 2006.

[109] Holly A. Yanco. Evaluating the performance of assistive robotic systems. In *Proc. of the Workshop on Performance Metrics for Intelligent Systems*, August 2002.

Bibliography – web resources

[110] http://trilobite.electrolux.com/.

[111] http://world.honda.com/asimo/.

[112] http://www.aquaproducts.com/.

[113] http://www.earth.google.com/, Google Earth product family.

[114] http://www.ifr.org/, IFR International Federation of Robotics.

[115] http://www.mhi.co.jp/kobe/wakamaru/english/index.html.

[116] http://www.wikipedia.org.

[117] http://www.omg.org/, Unified Modeling Language specification, version 1.5. *OMG document formal*, 2003.

[118] http://www.ptgrey.com/, Bumblebee digital stereo vision camera. Point Grey Research Co., February 2005.

A. Extremum seeking control

The basis of extremum seeking control was set at the beginning of the 1920s [11], with further valuable contributions added in the 1960s. This control method found its way in a variety of control applications governed by highly nonlinear plants [53, 54]. Recently, stability analysis of this control method was investigated in [50].

In Figure A.1(a) the block diagram of the extremum seeking control method is presented.

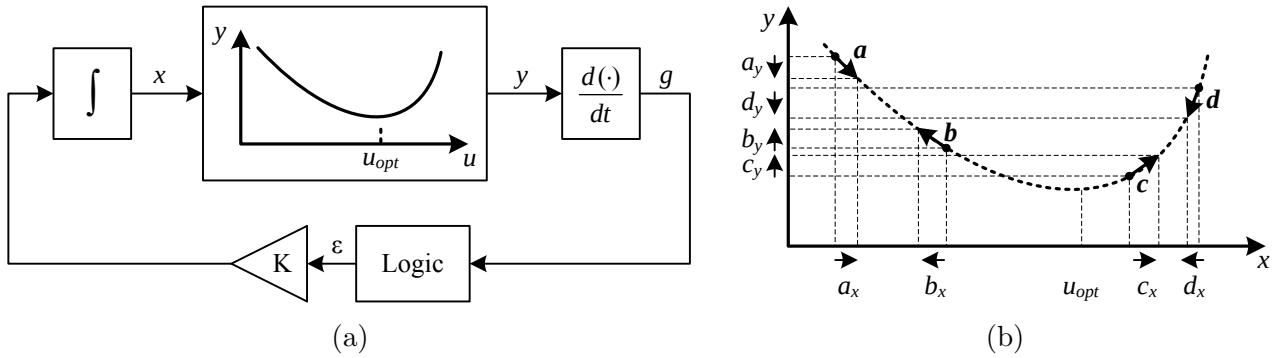


Figure A.1.: Principle of extremum seeking control. (a) Block diagram. (b) Example cases of the extremum seeking mechanism.

The equations describing the functioning of the method are basically an integrator:

$$\frac{dx}{dt} = K\epsilon, \tag{A.1}$$

where $\epsilon = \pm 1$ and K is a constant; a differentiator:

$$g = \frac{dy}{dt} \tag{A.2}$$

and a logic circuitry subsystem which implements the function:

$$L = \begin{cases} \text{change the sign of } \epsilon \text{ if } g > 0, \\ \text{keep the sign of } \epsilon \text{ if } g < 0. \end{cases} \tag{A.3}$$

The mechanism behind extremum seeking control is depicted in Figure A.1(b), where four cases are distinguished:

- Case $\boldsymbol{a}$:
$$\left.\frac{dx}{dt}\right|_{t^-} > 0 \textbf{ AND } \left.\frac{dy}{dt}\right|_{t^-} < 0, \tag{A.4}$$
where the horizontal component is increasing and the vertical one is decreasing, that is the vector $\boldsymbol{a}$ describes the movement of the working point towards the optimal point from its left side. In this case the controller must keep the sign of the horizontal variation, i.e. $\left.\frac{dx}{dt}\right|_{t^+} = K$.
- Case $\boldsymbol{b}$:
$$\left.\frac{dx}{dt}\right|_{t^-} < 0 \textbf{ AND } \left.\frac{dy}{dt}\right|_{t^-} > 0, \tag{A.5}$$
where vector $\boldsymbol{b}$ moves away from the optimal point, since the horizontal component is decreasing and the vertical one is increasing. The logic circuitry must change the sign of the of the horizontal variation, i.e. $\left.\frac{dx}{dt}\right|_{t^+} = K$.
- Case $\boldsymbol{c}$:
$$\left.\frac{dx}{dt}\right|_{t^-} > 0 \textbf{ AND } \left.\frac{dy}{dt}\right|_{t^-} > 0, \tag{A.6}$$
where both components of vector $\boldsymbol{c}$ move away from the optimal point. In this case the controller must change the sign of the horizontal variation, i.e. $\left.\frac{dx}{dt}\right|_{t^+} = -K$.
- Case $\boldsymbol{d}$:
$$\left.\frac{dx}{dt}\right|_{t^-} < 0 \textbf{ AND } \left.\frac{dy}{dt}\right|_{t^-} < 0, \tag{A.7}$$
where both the horizontal and vertical components of vector $\boldsymbol{d}$ are decreasing, that is they are moving towards the optimal point. Here, the logic circuitry must keep the sign of the horizontal variation, i.e. $\left.\frac{dx}{dt}\right|_{t^+} = -K$.

Having in mind that $\frac{dy}{dx} = \frac{dy}{dt} / \frac{dx}{dt}$, Equations A.4 to A.7 can be reduced to:

$$\left.\frac{dx}{dt}\right|_{t^+} = K \text{ if } \left.\frac{dy}{dt}\right|_{t^-} < 0, \tag{A.8}$$

$$\left.\frac{dx}{dt}\right|_{t^+} = -K \text{ if } \left.\frac{dy}{dt}\right|_{t^-} > 0. \tag{A.9}$$

Furthermore, Equations A.8 and A.9 can be written as:

$$\frac{dx}{dt} = -K \cdot \text{sign}\left(\frac{dy}{dx}\right). \tag{A.10}$$

The extremum seeking algorithm measures the sign of $\frac{dy}{dt}$, whereas the resulting dynamics are governed by $\frac{dy}{dx}$. The extremum $\frac{dy}{dx} = 0$ of the curve in Figure A.1(b) corresponds to the equilibrium point $\frac{dx}{dt} = 0$.

B. Universal Modeling Language

The birth of UML dates back to 1997 when it was standardized by the *Object Management Group* (OMG) consortium [117]. At the center of UML are nine types of modeling diagrams from which five are used in this thesis:

- *Use Case Diagrams*: shows the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases;
- *Class Diagrams*: describes the structure of a system by showing the system's classes, their attributes, and the relationships among the classes;
- *Sequence Diagrams*: shows how objects communicate with each other in terms of a sequence of messages; also indicates the lifespans of objects relative to those messages;
- *Statechart Diagrams*: standardized notation to describe many systems, from computer programs to business processes;
- *Activity Diagrams*: represents the business and operational step-by-step workflows of components in a system; an activity diagram shows the overall flow of control.

Over the past years UML has been extended with several other tools for coping with new problems suited for the UML standard. One such extension is the *System Modeling Language* (SysML), a tool used by systems engineers to specify and structure systems [35].

C. Genetic algorithms optimization

Genetic algorithms, or GA, is a search technique used in computing to find exact or approximate solutions to optimization and search problems [116]. GA are categorized as global search heuristics methods, belonging to the class of *evolutionary algorithms* that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover.

The basic structure of an evolutionary algorithm is shown in Figure C.1. At first, an *Initial Population* of possible solutions is selected at random from the search space. The values of the objective functions for each of these *individuals* or *genotypes* of the population are calculated and, based on this, a fitness value is assigned to each individual. A selection process filters out the individuals with bad fitness values and allows the fitter members to enter the next stage of the algorithm, the reproduction stage. In this phase, *offspring* is generated by varying or combining the genotypes of the selected individuals and integrated into the population. The procedure is repeated iteratively, until the termination criterion is met. Common termination criterias are:

- a solution has been found that satisfies a minimum fitness value;
- a fixed number of generations has been reached;
- manual inspection.

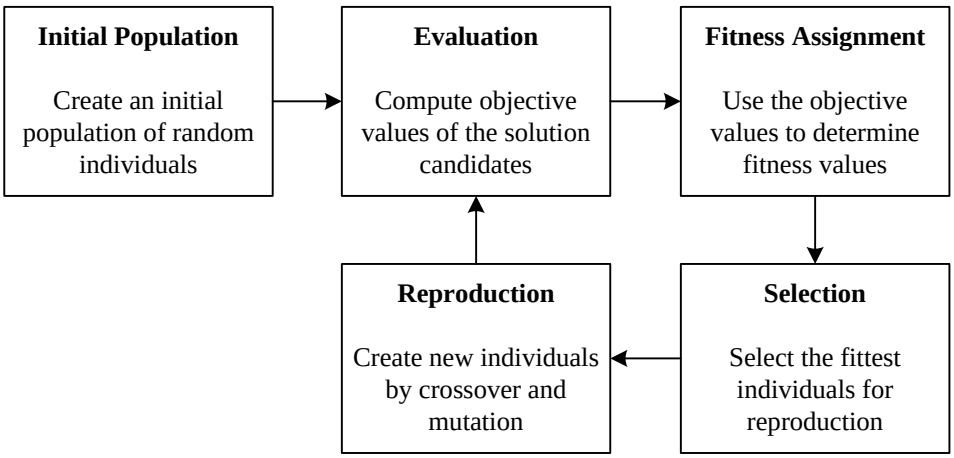
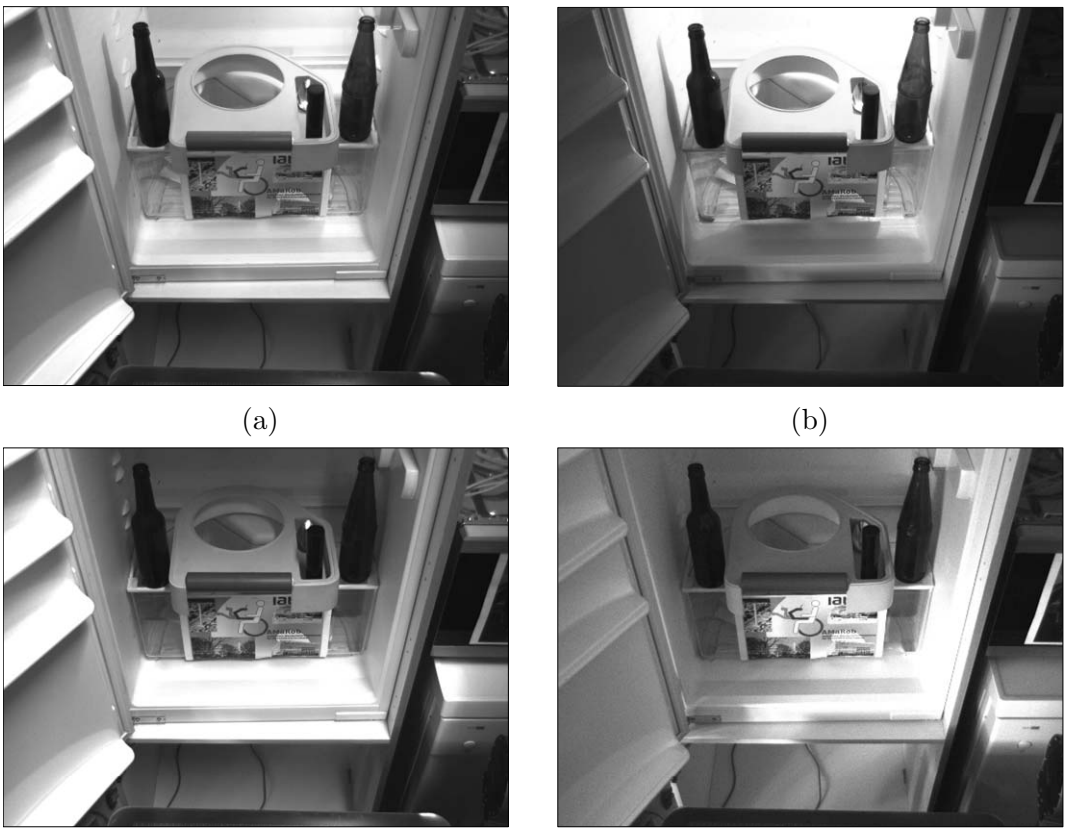


Figure C.1.: Basic structure of evolutionary algorithms.

Genetic algorithms represent a subclass of evolutionary algorithms, where the elements of the search space are encoded as binary strings or arrays of other elementary types. For this reason, they are also referred to as *chromosomes*. In the reproduction phase of the

GA, offspring is generated by the means of *crossover* and *mutation*. Thereby, crossover is used to generate the child chromosomes. When performing single-point crossover, both parental chromosomes are split at a randomly selected *crossover point*. Subsequently, one or two child genotypes are generated by swapping the upper and lower parts of the parental chromosomes. Mutation is used to preserve the diversity of a population by introducing a small chance of changing an individual. If the individuals are encoded as binary strings, this can be achieved by randomly toggling one of the bits.

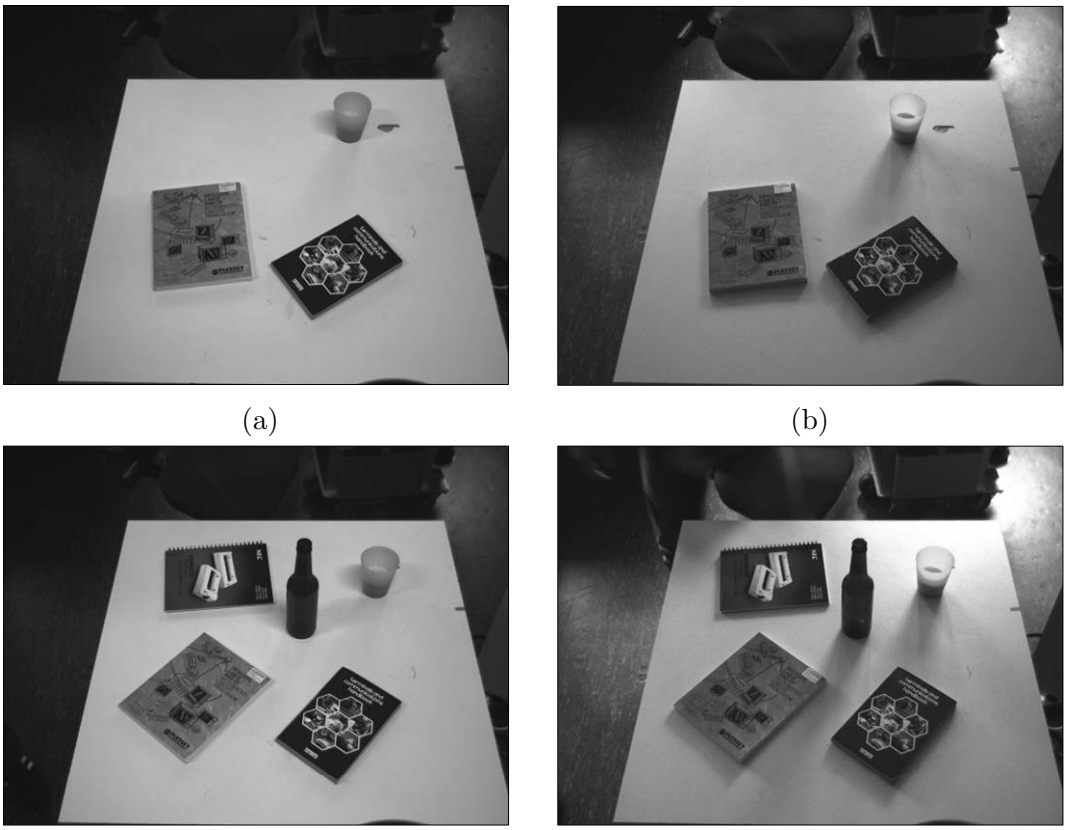
D. Sample images from FRIEND support scenarios



(a) (b)

(c) (d)

Figure D.1.: Images of the same FRIEND ADL scenario scene acquired in different illumination confitions. (a) 590 lx. (b) 328 lx. (c) 225 lx. (d) 137 lx.



(a) (b)

(c) (d)

Figure D.2.: Images from the same FRIEND Library scenario scene acquired in different illumination confitions. (a) 481 lx. (b) 218 lx. (c) 455 lx. (d) 179 lx.

E. List of abbreviations

2D	Two dimensional
3D	Three dimensional
ADL	Activities of Daily Living
ASIC	Application-Specific Integrated Circuit
BCI	Brain-Computer Interface
CCD	Charged Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
CORBA	Common Object Request Broker Architecture
DEC	Discrete Event Controller
DoF	Degrees of Freedom
DoG	Difference of Gaussian
DSP	Digital Signal Processor
FRIEND	Functional Robot with dexterous arm and user-frIENdly interface for Disabled people
FOV	Field of View
FPGA	Field-Programmable Gate Array
FPS	Frames Per Second
GUI	Graphical User Interface
HFOV	Horizontal Field of View
HMI	Human-Machine Interface
HSI	Hue-Saturation-Intensity color space
MASSiVE	Multi-Layer Architecture for Semi-Autonomous Service-Robots with Verified Task Execution
MVR	Mapped Virtual Reality
OMG	Object Management Group
P	Proportional controller
PI	Proportional-Integral controller
POSE	Position and Orientation
PTH	Pan-Tilt Head
RGB	Red-Green-Blue color space
ROI	Region Of Interest in image
ROVIS	RObust machine VIsion for Service robotics
SIFT	Scale-Invariant Feature Transform
ToF	Time of Flight
UML	Universal Modeling Language

F. List of symbols

$A(i,j)$	hough transform accumulator cell
$\mathbf{B}$	binary images vector
$\mathbf{C}$	extracted objects contours vector
C_l	object color class
d_r	Euclidean distance
E	region of interest edge
$f(x,y)$	digital image
H_{ev}	amount of color information
$I_{1...7}$	invariant moments
I_m	uncertainty measure
l	hough transform edge
p_8	estimate of the probability of a segmented pixel surrounded with 8 segmented pixels in its 8-pixel neighborhood
$pt_{int}(x,y)$	image interest point
$ROI(f\|x,y,w,h)$	image region of interest
T_L	low canny threshold
T_H	high canny threshold
T_{HG}	hough transform accumulator threshold
$[T_{min}, T_{max}]$	object thresholding interval
T_{opt}	optimal threshold
$t(x,y)$	thresholded binary image
u	actuator variable
u_{opt}	optimal value of actuator variable
(x,y)	pixel coordinates in an image
(x,y,z)	object coordinates in cartesian 3D space
$\mathbf{Y}$	extracted object features vector
y	controlled variable
α	pan angle
β	tilt angle
$\Delta\nu$	difference between hough transform angles
η_{pq}	object central moments of order $(p+q)$
μ_{pq}	object moments of order $(p+q)$
ρ_i	probability of the ROI edge E_i to intersect object pixels