DISSERTATION

Robust Range Image Processing for Robot Manipulation

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der technischen Wissenschaften unter der Leitung von

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Markus Vincze
Institut für Automatisierungs- und Regelungstechnik (E376)

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informatinstechnik

von

Dipl.-Ing.(FH) Mario Richtsfeld
geb. am 16.05.1979
Matr. Nr.: 0326895
Grestnerstr. 26/3/1
A-3250 Wieselburg

Wien, im Dezember 2009

Mario Richtsfeld

Abstract

Today, robots are used for always new tasks in our daily life. There are robot lawn mowers, automatic vacuum cleaners, automatic guided vehicle systems or automatic walkers and gripping aids for people with disabilities. The task is to extend the capabilities of modern robots from simply following given procedures to achieve autonomous decisions. Thus, the use of sensors to scan the environment becomes essential. Environmental information has to be processed to avoid collisions or to achieve a robust detection of objects.

Particularly, laser sensors stand out for the detection of the environment due to their high degree of accuracy. However, industrial and domestic applications often provide single-view images. The challenge for the subsequent range image processing is to handle the shadowing effects of object- and self-occlusions.

This thesis deals with the efficient interpretation of range images for industrial as well as service robotics in an unstructured environment attempting to solve this single view challenge.

For the industrial task of autonomous robot stitching a robust edge tracking method of two overlapping carbon fibre mats has been researched. For this application real-time performance is needed to guarantee continuous edge tracking. The key challenges are robust edge detection and tracking in order to set the seams correctly and to control the robot motion.

The domestic field exhibits clear differences to industrial applications. In comparison to industrial applications where the robots act in specific robot cells with defined tasks, domestic robot applications exhibit the problem of unstructured environments. Three different cases for object grasping have been considered for this purpose:

Firstly, a method for robust detection of cylindrical objects in cluttered tabletop scenes was developed. A reliable feature detection is essential for the safe detection and gripping of cylindrical objects on the table.

Secondly, methods were developed to safely grasp arbitrary objects in cluttered tabletop scenes. Potential grasping points and poses are determined based on detected geometrical features, e.g., edges, planes, curvatures, and rotation axis of the segmented objects.

Finally, a segmentation method for 3D objects into useful sub-parts was developed. This work is based on the observation that human vision segments an object into different parts and analyses their spatial and functional relationships before a grasping and manipulation task is executed. A part-based description allows to detect the different parts and functional properties (e.g., handles and container) of an object. The developed approach represents a novel method to segment 3D point clouds as well as meshes.

All the developed methods of this thesis are evaluated in detail and show their efficiency and robustness.

Kurzfassung

Heute finden Roboter Verwendung für immer neue Aufgaben im alltäglichen Leben, als selbstfahrende Rasenmäher, Staubsauger, fahrerlose Transportsysteme oder als Geh- bzw. Greifhilfe für Behinderte. Die Fähigkeiten eines modernen Roboters müssen dabei so erweitert werden, dass dieser nicht nur vorgegebenen Abläufen folgt, sondern eigenständige Entscheidungen trifft. Um dies zu bewerkstelligen sind Sensoren zur Abtastung der Umgebung notwendig. Die Umgebungsdaten müssen ausgewertet werden, um Kollisionen zu vermeiden oder Objekte bzw. deren Eigenschaften robust zu erkennen.

Lasersensoren zeichnen sich durch eine hohe Genauigkeit bei der Umgebungserfassung aus. Jedoch sind die Objekte bei industriellen, als auch praktischen Anwendungen oft nur von einer Seite zugänglich. Dadurch enstehen bei der Bildaufnahme durch einen Laserscanner Abschattungen, welche für die Bildverarbeitung eine große Herausforderung darstellen.

Diese Dissertation beschäftigt sich daher mit der effizienten Auswertung von Tiefenbildinformationen vom Industrie- bis zum Servicerobotikbereich bei einer veränderlichen Umgebung mit einer eingeschränkten Umgebungserfassung.

Um das automatische Vernähen zweier überlagerter Kohlefasermatten zu ermöglichen wurden Methoden zum sicheren Verfolgen des Kantenverlaufes entwickelt. Dabei ist eine hohe Verarbeitungsgeschwindigkeit gefordert, die eine kontinuierliche Bewegung des Roboters entlang der Nahtbahn gewährleistet. Die Herausforderung dabei ist eine robuste Kantendetektion und -verfolgung, um eine korrekte Naht zu erreichen.

Der Bereich der Servicerobotik weist deutliche Unterschiede zum Industriebereich auf. Während bei Industrieanwendungen ein Roboter in einer definierten Umgebung arbeitet, muß ein Roboter im Servicerobotikbereich mit einer veränderlichen Umgebung zurechtkommen. Für diesen Zweck wurden drei Problemfälle zum Greifen von Objekten betrachtet:

Zu Beginn wurde eine Methode entwickelt, um bekannte zylindrische Gegenstände auf einem Tisch unter anderen Objekten zu erkennen und einzupassen. Eine verläßliche Merkmalsdetektion ist notwendig, damit ein sicheres Greifen der zylindrischen Objekte am Tisch ermöglicht wird.

Daraufhin wird versucht, die wissenschaftliche Lücke zum Greifen beliebiger Objekte zu schließen. Basierend auf den detektierten, geometrischen Merkmalen wie Kanten, Ebenen, Krümmungen und Rotationsachsen werden potentielle Greifpunkte als auch Positionierungen des Robotergreifers berechnet.

Zuletzt wird das Segmentieren beliebiger Objekte in “sinnvolle” Einzelteile behandelt. Diese Arbeit basiert auf der Tatsache, dass der Mensch unbewußt bevor er ein Objekt greift und handhabt dieses in Einzelteile zerlegt und deren Zusammensetzung analysiert. Diese Eigenschaft ermöglicht es dem Menschen zwischen den funktionalen Bereichen eines Objektes, wie zum Beispiel eines Henkels und dem Behälter, zu unterscheiden. Der in dieser Arbeit vorgestellte Ansatz stellt eine neue und schnelle Möglichkeit dar, um Punktwolken als auch Polygonnetze zu segmentieren.

Die in dieser Arbeit entwickelten Methoden wurden experimentell evaluiert und deren Effektivität und Robustheit gezeigt.

Acknowledgements

I would like to express my gratitude to Markus Vincze for supervising my project employment and for his support and advice during the thesis work. I am also indebted to Prof. Peter Kopacek for his review of this thesis as second supervisor. My gratitude additionally goes to Prof. Alexander Weinmann from the Vienna University of Technology who drummed up my interest for automation and control and supported the possibility to write my thesis. I am grateful to my office colleagues of the V4R group at ACIN who continuously cared for an enjoyable working environment. Special thanks to Michael Zillich for his inspiring ideas and many interesting discussions.

I further want to thank my mother for her support during all the years of my study and finally, I am grateful to all the people who supported me with valuable comments and discussions or with their patience and appreciation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"The human mind has first to construct forms,
independently, before we can find them in things."
– *Albert Einstein*

This dissertation addresses the problem of robust feature detection in range images to realise stable manipulation tasks. To obtain accurate 3D object information laser range scanners or stereo cameras are used to obtain 3D coordinates of points on object surfaces. In order to minimise the computational effort, accurate model representation is needed in order to allow a faster feature detection. Robust feature detection contains the extraction of lines, edges, and planar regions directly from range images. The goal is to present and experimentally evaluate some 3D vision based techniques and to demonstrate how these techniques can be employed during typical robotic tasks. The problems of robustness and flexibility in terms of feature detection are addressed. These detected features in combination with a task based a-priori knowledge enable safe and efficient robot handling and manipulation tasks.

To augment the possibilities and flexibility of robots in different applications, 3D vision systems are needed to obtain higher performance of automated processes. Hence efficient methods for 3D image processing are used.

Humans show an extraordinary capability for several manipulation and detection tasks of objects [Kuhtz-Buschbeck et al., 1998]. E.g., grasping of objects above or beside, depending on possible collisions, and scaling of the grip aperture adequately, according to various sizes of the target object. This is one of the characteristics that most distinguishes humans from other species of living creatures in the world. Manipulation is one of the most useful and challenging tasks in any robot research system and represents a key component for many robotic applications in all kinds of areas such as industry, medicine, services, the automobile industry and space robotics. Robotics manipulation range from the design of robotic hands, object detection, segmentation, physics of contacts among objects, to studies on neurological and psychological foundations of manipulation.

In general, these applications need to be flexible enough in order to deal with unstructured and uncertain working environments. In the case of grasping tasks this uncertainty stems from several sources: pose estimation of the objects, relevant features of objects, and the diversity of tasks to perform with the objects. To overcome these difficulties, a combination of a wide range of partial solutions is used. Among them, relevant sensory information is needed.

The goal of this work is to advance well-known topics of 3D image processing to realise more robust methods for industrial applications. The considered areas are: edge detection, object detection and object segmentation. Fig. 1.1 illustrates "The Big Picture" of the presented thesis to realise a step towards full automation.

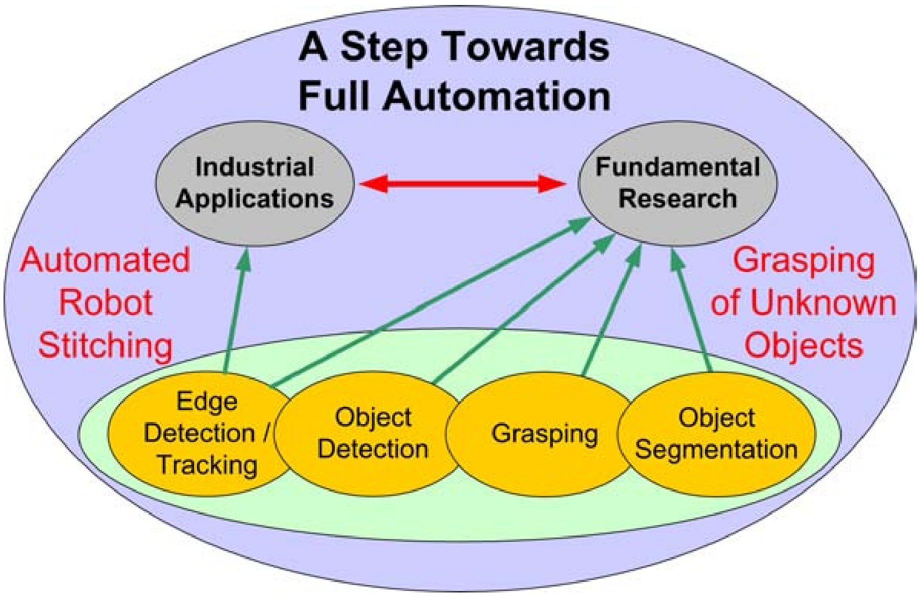


Figure 1.1: "The Big Picture" of the presented thesis.

There exists a number of robust edge detection and tracking methods for industrial applications, however these methods exhibit big problems for new tasks such as automated robot stitching of carbon fibre mats. The novelty is based on the combination of different existing and newly developed edge detection methods to carry out robust edge tracking in the presence of outliers and artefacts in noisy range data using an industrial robot.

The grasping of arbitrary, unknown objects on a table by a robot arm represents an unsolved problem also for fundamental research. The considered areas for this task are: object detection, grasping point and pose detection, and object segmentation to realise robust grasping of unknown objects for industrial deployment in future time.

This thesis introduces two research areas to realise fast, robust, and efficient range image processing for robot manipulation.

The first part of this work considers a novel method for fast and robust real-time edge detection for optical seam following. Several edge detection methods are combined in a voting scheme to increase the edge tracking robustness.

The second part presents three challenges for grasping:

1. Rapid detection of cylindrical objects:
   The first application in grasping considers a low-level contribution in rapid detection

of cylindrical objects on a table to realise a stable grasp, and manipulation with a 7 DOF robot arm based on a single view.

2. Grasping of unknown objects:
   The second part describes a robust approach to calculate possible grasping points and poses for unknown objects placed on a table, scanned from a single view. In particular, a novel method to accomplish this is presented. In order to deal with unknown objects, a grasping system must be able to use its sensors to detect and extract all necessary information about the object, since they are not yet available as a model. This information is then used to decide on the most convenient way to grasp the object.

3. Point cloud segmentation to obtain different object parts:
   The third part presents a novel point cloud and mesh segmentation method to extract an object into different parts. Some sub-parts of an object deliver potential grasping poses [Huebner et al., 2008]. Additionally, dividing a point cloud into simpler sub-parts has several benefits in modelling [Funkhouser et al., 2004], robotics [Huebner et al., 2008] or collision detection [Li et al., 2001].

To exploit the flexibility of robots 3D vision systems are required in both parts for high performance automated processes. Thus, 3D image processing is the important link between the robot vision and its autonomous behaviour.

In the following Section 1.1, a motivation for this research is given ending up in a presentation of the problem statement. The contributions of this thesis are described in Section 1.2 and finally, the organization of this dissertation is provided in Section 1.3.

1.1 Motivation and Problem Statement

Robotic appliances are increasingly becoming a bigger part in our everyday lives. In the near future, service robots will support people with several handicaps to improve the quality of their lives, which includes a high demand of robustness and flexibility. Today most of the work carried out in robotics is still concerned with finding solutions to fundamental problems like object detection, segmentation, grasping or handling. To solve these problems robust feature detection is used to obtain process relevant and task specific information.

3D data is extensively used for feature detection by using stereo systems or laser range scanners. Due to the high amount of data, efficient methods for 3D image processing in the growing field of robotics are needed to realise the necessary requirements. The motivation of this thesis is to take a step towards full automation and to rapidly detect features for industrial and robotic applications.

1.1.1 Feature Detection for Automated Fibre Mat Stitching

Textiles produced with carbon fibres are becoming increasingly important for the automobile and aeronautic industry. Robust and light-weight components are manufactured by infiltrating the carbon fibres with resin. The gluing of these carbon fibre reinforced polymer (CFRP) composites strains several parts in the final hardening step and thus a new technology is needed to improve the pre-forming process. Sewing instead of gluing strengthens the final properties and achieves better solidity and durability. First industrial applications such as the AIRBUS A380 rear pressure bulkhead have been realised by showing the potential of the stitching technology. In this case the stitching head is mounted on a portal and the sewing process is performed in 2D. To achieve a 3D application, an industrial robot equipped with a sewing head is used. One of the final goals is to achieve lot-size-one production with an industrial robot.

Today, teach-in programming, i.e. an operator code specifying points on the trajectory for flexible production, is no longer state-of-the-art. Sensors are used to interact with the robot control for a fully automatic process. During the stitching process possible obstacles are detected and the trajectory along the edge of the fibre mats is adjusted. Using such a visual approach, a higher reproducibility and precision of the seams is expected [Sim et al., 2002].

The objective is to develop a high-precision automated stitching system. So a laser range sensor is used in front of the sewing head to guide the robot along the carbon fibre edges as illustrated in Fig. 1.2. The key challenge is reliable, robust edge detection and tracking in order to set the seams correctly and to control the robot motion. Range scanners are one of the most efficient ways of measuring the carbon fibre mats.

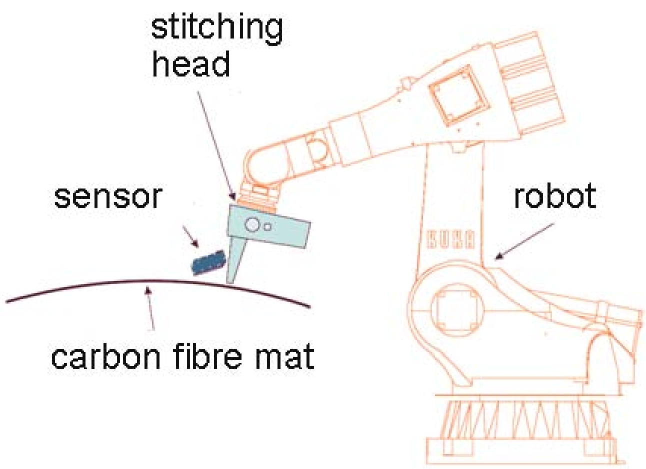


Figure 1.2: Overview of the edge detection and tracking approach for an automated fibre mat stitching system.

The challenge is to find a reliable and robust method to detect the edge between two overlapped carbon fibre layers, because this data is the main perceptive input for robot control. The difficulties to overcome are the black and reflective carbon fibres, an edge height between $0.3mm$ and $1.1mm$, outliers, and obstacles. To obtain reliable results a two out of three voting scheme for edge detection is applied.

1.1.2 Feature Detection for Object Grasping

"People have always been fascinated by the exquisite precision and flexibility of the human hand. When hand meets object, we confront the overlapping worlds of sensorimotor and cognitive functions" [Castiello, 2005]. Although partial solutions for certain cases exist, there is still no general valid solution to grasp an object. In the near future, service robots will support people with several handicaps to improve the quality of their lives. One of the required key technologies is to set up the grasping ability of the robot. This includes autonomous object detection of a wide variety of various objects scanned from one single view in a cluttered setting. The determination of the object pose enables a grasp motion to fulfil the task of providing objects from any position on a table to the user. That also includes the segmentation of the objects on the table as illustrated in Fig. 1.3.

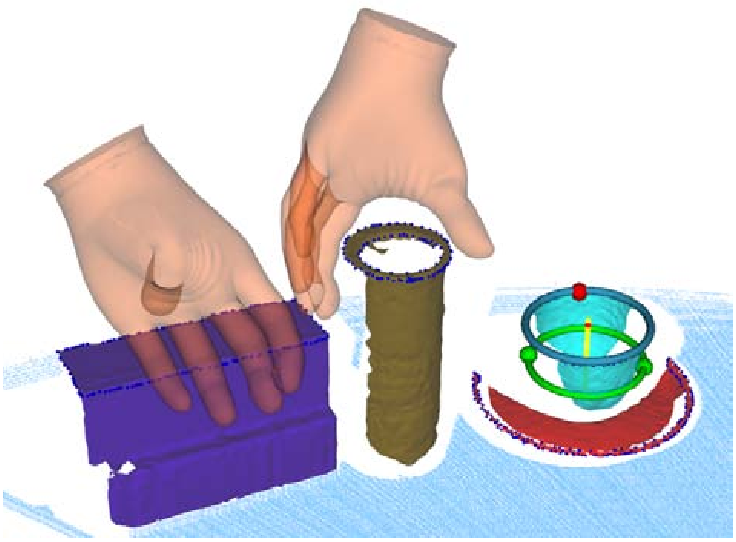


Figure 1.3: Detection of grasping points and hand poses. The green points display the computed grasping points for rotationally symmetric objects. The red points show an alternative grasp along the top rim. The illustrated hand poses show a possible grasp for the remaining graspable objects.

Human vision segments an object into sub-parts and analyses their spatial and functional relationships [Shipley and Kellman, 2001]. A part-based description allows to detect the different reciprocal functional properties of an object.

The task of grasping an unknown object, is often considered trivial from our human point of view. Humans use prior knowledge to perform simple tasks like opening a door or grasping a mug from a table. The solution is that a very complicated task must be designed as a series of simpler tasks. The system has to detect the object, prepare a grasping motion and then grasp, handle, and manipulate the object.

An additional challenge is that the acquired range images are sparse, due to the laser and camera shadows and from a single view, the rear side of an object is not visible due to self occlusions and the front side may be occluded by other objects.

1.2 Contributions

Robust feature detection is an important part in range image processing and scene understanding. Depending on the complexity of the model structure, different methods are possible. This thesis handles the problem of efficient model feature detection for several tasks under the constraint of achieving fast and robust results.

The first contribution of this work presents an efficient two out of three voting scheme for robust 2D edge detection and tracking in an industrial environment.

To work towards a solution for the challenge of grasping unknown objects, three contributions will be presented: a rapid cylinder fitting method, a robust grasping point and pose detection based on 2.5D range images, and finally model part segmentation based on 3D point clouds.

1.2.1 Edge Detection

Edge detection and tracking is a well analysed section in the field of range image processing. However, the main problem is that especially industrial applications need a robust and real-time edge detection method to handle the problem of noisy scan lines under changing light, surface, and material conditions to close the gap between laboratory research and flexible industrial production.

The approach of an automated CAD (Computer Aided Design) based path planning and a sensor guided edge detection to realise a path adjustment can be applied for a sewing process that realises lot-size-one production.

Contributions in this field have been made by introducing a real-time approach where existing and new developed edge detection methods are combined in a voting scheme to increase the edge tracking robustness [Richtsfeld et al., 2007a]. The novelty is based on different edge detection methods and their combination to carry out robust edge tracking in the presence of outliers and artefacts in noisy range data using an industrial robot. The individually developed methods show a very high reliability [Biegelbauer et al., 2007].

The experiments show that a two out of three voting over three methods achieves a better detection result than the individual methods. Thus the approach of a voting scheme for edge detection and localisation is suitable for the use in related industrial applications under difficult conditions. Two different laser stripe sensors have been tested, whereby the developed edge detection methods and the voting scheme highlighted their flexibility and robustness. Chapter 3 describes the approach in detail and demonstrates the robustness and computational effectiveness [Biegelbauer et al., 2007, Richtsfeld et al., 2007a, Richtsfeld et al., 2007b, Richtsfeld et al., 2007c].

1.2.2 Rapid Cylinder Detection

Least-square fitting is a well known and popular method to fit cylinders into range data, but this technique works well only with a good enough estimation of the starting pose, noiseless range data and enough data points around the barrel. Section 2.5 summarises least-square fitting and model-based cylinder detection.

In the fields of industrial and home robotics, the requirements of complete 3D data, noiselessness, and obstacle-free situations are often not provided. The contribution of this work is a fast and robust method optimised for fitting cylinders in sparse and noisy range data under difficult and changing light conditions recorded from a single view.

Observing the curvature of the scanned objects is needed to segment and detect the cylindrical object. The method presents a cylinder fit based on random samples to handle the typical outlier problem.

The first step of the approach is to discover the table surface by a RANSAC-based [Fischler and Bolles, 1981] plane fit and to analyse the curvature of the objects with the normal surface vectors. As a second step, these curvature points are used to segment all objects on the table and to calculate the radius with a RANSAC-based 3D circle fit. The normal surface vectors are used to calculate the cylinder axis and the pose of the detected cylinder. Chapter 4 describes the approach in detail and demonstrates the robustness and computational effectiveness compared with the standard least-square fitting of cylindrical objects [Richtsfeld et al., 2008].

1.2.3 Grasping Point and Grasping Pose Detection

The grasping task was studied from a psychological, biological, and engineering focus but still remains unresolved. Although partial solutions for certain cases exist, there is still no general valid solution. Section 2.7 gives a short summary of the wide field of grasping and highlights the difficulties in establishing a stable grasp.

In home robotics the grasping task is usually limited to a small number of known and simple object shapes. The contribution of this work is a fast and robust method to identify potential grasping points and grasping poses based on the top surfaces of the unknown object shapes on a table. Therefore only one single view from the laser range scanner is available where 2.5D point clouds are obtained. The acquired range images are sparse due to single scanning with laser and camera shadows.

As mentioned above, the algorithm detects the table surface with a RANSAC-based plane fit. The algorithm then segments the objects on the table. Different segmentation methods have been tested, based on region-growing and mesh generation. To avoid a possible over-segmentation of rotationally symmetric objects, an additional merging step is implemented. Then the algorithm analyses the curvature of the objects with the normal surface vectors, which are calculated by a principal component analysis (PCA) either on the point clouds or automatically during the mesh generation step. The top surface of the objects is also detected with a RANSAC-based plane fit taking the normal surface vectors into account. Based on the information received, potential grasping points and poses are detected by analysing the top rim or shape of the objects, whereby parallel planes or surface patches are detected and used to obtain a potential grasp from a single view. Chapter 5 describes the approach and the developed method in detail and demonstrates the robustness and computational effectiveness by grasping a variety of objects [Richtsfeld and Vincze, 2008a, Richtsfeld and Vincze, 2008b, Richtsfeld and Vincze, 2008c, Richtsfeld and Zillich, 2008, Richtsfeld and Vincze, 2009b].

1.2.4 Object Part Segmentation

Dividing a point cloud into simpler sub-parts has several advantages for modelling [Funkhouser et al., 2004], robotics [Huebner et al., 2008] or collision detection [Li et al., 2001]. The presented work includes a new segmentation algorithm, based on radial reflection. Although the examples in Chapter 6 are related to applications in the area of computer graphics and robotics, the majority of the algorithm developed here can be applied with only trivial modifications to more complex segmentation problems. The segmentation pipeline consists of a convex hull-based segmentation algorithm to identify the potential core part and all sub-parts of the object.

The contribution is an advanced method based on the work of [Katz et al., 2005], which directly segments a point cloud and the algorithm is also able to segment a mesh. Chapter 6 describes the method in detail and demonstrates the robustness and computational effectiveness compared with standard segmentation methods. Additionally, the cut discrepancy and the rand index of the proposed method have been evaluated with the help of 10 object classes [Richtsfeld and Vincze, 2009a]. In comparison with other segmentation methods the proposed algorithm based on radial reflection shows best time performance.

1.3 Thesis Overview

This thesis is organised as follows: Starting with the state-of-the-art technology in Chapter 2, a theoretical background is given of: edge detection, curvature analysis, and segmentation methods. An overview of geometric fitting, mathematical methods for least-squares minimisation and mesh processing is also presented. Finally, this chapter introduces grasping from a biological, psychological, and technical view.

Chapter 3 presents an application-driven research project. This is necessary to understand the motivation of this thesis and the research potential in the future, of which feature detection in range images is needed. Hence an edge detection method based on a two out of three voting over three methods is presented. The experiments show that a voting over three methods achieves a better detection result than the individual methods.

Chapter 4 introduces a simple and quite fast method for a rapid detection of known cylindrical objects in range images. This low-level 3D image processing is evaluated in detail with a performance analysis.

Chapter 5 describes a new and efficient method for the calculation of grasping points and poses for unknown objects based on range data. The proposed grasping method is evaluated in detail with a grasp analysis.

Chapter 6 presents a 3D point cloud and mesh segmentation method. The efficiency of this method is presented with a lot of different shape examples.

All methods are evaluated in terms of robustness and computational performance.

Finally this work is concluded with a discussion on the reliability of rapid object detection methods in Chapter 7, which ends up with future research topics, e.g. a concept of a fully automated learning, registration and detection system.

Chapter 2

State of the Art

"A scientific theory should be as simple as possible, but not simpler."
– *Albert Einstein*

This chapter looks at the research literature on 3D part and feature detection. The model-based detection of features requires a 3D object description, and the reason why Section 2.1 summarises geometric primitives, generalised cylinders and their applications in 3D vision.

Section 2.2 presents different edge detection methods on single scan lines. Based on these methods, Section 2.3 reviews the literature on curvature analysis focused on 3D point clouds.

Section 2.4 gives an overview of point cloud segmentation methods. Different methods to carry out automatic 3D object segmentation into meaningful parts have been published in the last few years. 3D model segmentation algorithms can be categorised into two main classes. The first class is developed for applications like reverse engineering of CAD models [Attene et al., 2006b]. The second class attempts to segment natural objects into meaningful parts and are mainly region-based or edge-based range image segmentation methods.

Section 2.5 introduces least-squares fitting, by describing the common mathematical methods. The fitting of the geometric models in point clouds is an important issue. A fast and robust detection process is especially required for industrial applications.

Section 2.6 gives a short overview of mesh processing and the most important basics are illustrated there. A point cloud can be also represented by a mesh. A mesh or grid is a combination of vertices, edges and faces that define the shape of an object in 3D modelling. Normally the faces consist of triangles or other simple convex polygons because this simplifies rendering and more general concave shapes could be composed. Thus, mesh processing is a difficult and significant task in 3D image processing. Different methods, however can only operate on a mesh.

Concluding this chapter, Section 2.7 reviews the literature on grasping from a biological, psychological, and technical view.

2.1 Parametric Model Description

A large number of basic parametric models can be found by reviewing the research literature used in 3D computer vision, particulary on especially point clouds. A point cloud is a set of points in a 3D coordinate system. These points are defined by x, y, and z coordinates. These models range from simple descriptions to highly complex representations. The number of shape parameters varies depending on the complexity of the model shape. Simple model shapes like spheres, cylinders, cones, toroids, and cubes with a few number of parameters for shape description are geometric primitives. More complex shapes like generalised cylinders [Zerroug, 1999], superquadrics [Barr, 1981], and geons [Biederman, 1987] are models whose complexity in shape is limited as well as the number of parameters. In contrast high parametric models like hyperquadrics [Haverinen and Röning, 2000], implicit polynomials [Keren et al., 1994], and spherical harmonic surfaces [Brechbühler et al., 1992] can represent any arbitrary shape with an open number of parameters.

In order to avoid an overstrain of the reader with methods and characteristics of parametric model description in detail, the author gives only a short overview in the fields of geometric primitives and generalised cylinders.

2.1.1 Geometric Primitives

Geometric primitives are used in 3D vision for object recovery, recognition and detection. Fig. 2.1 presents the models by the increasing number of parameters. Tab. 2.1 illustrates the number of parameters for a full 3D description, with $\mathbf{r}$, $\mathbf{R}$, $\mathbf{h}$, $\mathbf{s}$, $\mathbf{\Psi}$ as shape parameters, $\mathbf{p} = (x, y, z)^\top$ as position point, and $\mathbf{a} = (a_x, a_y, a_z)^\top$ as orientation vector. Instead of using the orientation vector $\mathbf{a}$, which contains redundant orientation information for rotationally symmetric objects the angles α and β describe the orientation of the object by defining a rotation around the x- and y-axis of the world coordinate system. A

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

Figure 2.1: Geometric primitives: **(a)** *sphere,* **(b)** *cylinder,* **(c)** *cone,* **(d)** *torus,* **(e)** *cube.*

cube is not a rotationally symmetric object, so the orientation vector for a unique pose determination must be defined with three angles: α, β, γ. The orientation vector goes through the centre p_c and is perpendicular to the top surface. With the lateral length s the cube is fully described in 3D space. The description of range data has been intensively researched by fitting geometric primitives [Feddema and Little, 1997, Lukacs et al., 1998, Marshall et al., 2001, Taylor and Kleeman, 2003]. Rotationally symmetric primitives can

	Shape	Pose	# Parameters
Sphere	r	$\mathbf{p}_c$	4
Cylinder	r, h	$\mathbf{p}_a, \alpha, \beta$	7
Cone	h, Ψ	$\mathbf{p}_a, \alpha, \beta$	7
Torus	r, R	$\mathbf{p}_c, \alpha, \beta$	7
Cube	s	$\mathbf{p}_c, \alpha, \beta, \gamma$	7

Table 2.1: Summary of the geometric primitives shape and pose parameter.

be described with an implicit closed form, whereas with a cube no closed form description is possible [Taylor and Kleeman, 2002]. Instead a cube can only be modelled by more complex model descriptions such as a superquadric, hyperquadric or spherical harmonic surface.

The following sections give an overview in model description of geometric primitives with their parameters and distance functions. This description is needed to obtain the objective function for the least-squares fitting, which will be an important part in a later case. Initially a general description of a plane is presented, to give a fully comprehensive overview.

Plane

In general, a plane is not part of the geometric primitives. However, its role is very important for primitives and more complex models. Normally a plane is described with three points p_0, p_1, p_2, with none of them being coincident and not all of them being collinear. It is possible to directly derive any one of the several representations of a plane from three points [Schneider and Eberly, 2003]. In the following the implicit, the parametric, and the explicit plane representations are described.

- The implicit equation of a plane through three points satisfies:

$$\begin{vmatrix} x - p_{0,x} & y - p_{0,y} & z - p_{0,z} \\ p_{1,x} - p_{0,x} & p_{1,y} - p_{0,y} & p_{1,z} - p_{0,z} \\ p_{2,x} - p_{0,x} & p_{2,y} - p_{0,y} & p_{2,z} - p_{0,z} \end{vmatrix} = 0. \tag{2.1}$$

  The final equation is:

$$ax + by + cz + d = 0. \tag{2.2}$$

- The parametric equation of a plane through three points satisfies:

$$p(s,t) = p_0 + s(p_1 - p_0) + t(p_2 - p_0). \tag{2.3}$$

- The explicit representation requires one point on the surface, the normal plane vector, and a third parameter d, which represents the perpendicular distance to the origin, to be known. From three points the normal surface vector can be calculated as cross product of the vectors between two pairs of points. Arbitrarily choosing p_0 as the point on the surface, and the normal vector $\vec{n}$ as $(p_1 - p_0) \times (p_2 - p_0)$:

$$p_0 \cdot \vec{n} + d = 0. \tag{2.4}$$

Sphere

A sphere represents the simplest geometric model and is described by a centre p_c and a radius r, so that the implicit equation for the sphere is [Schneider and Eberly, 2003]:

$$f(p) = \|p - p_c\|^2 = r^2. \tag{2.5}$$

The intersection of a linear component and a sphere, defined by an origin O and a direction vector $\hat{d}$ can be computed by substituting the equation for a linear component:

$$\|p - p_c\|^2 = r^2, \tag{2.6}$$

$$\|O + t\hat{d} - p_c\|^2 = r^2, \tag{2.7}$$

$$\|(O - p_c) + t\hat{d}\|^2 = r^2. \tag{2.8}$$

The second-order equation is of the form:

$$at^2 + bt + c = 0 \tag{2.9}$$

and can be solved by using the quadratic formula[1]:

$$t = -\hat{d} \cdot (O - p_c) \pm \sqrt{(\hat{d} \cdot (O - p_c))^2 - ((O - p_c) \cdot (O - p_c) - r^2)}. \tag{2.10}$$

Cylinder

Another important geometric primitive in our daily life is the cylinder. A cylinder has a centre point p_c, unit-length axis $\hat{d}$, radius r, and height h. The end disks of the cylinder are located at $p_c \pm (h/2)\hat{d}$. Assuming $\hat{u}$ and $\hat{v}$ is one unit-length vectors so that $\hat{u}, \hat{v}, \hat{d}$ is a right-handed set of orthonormal vectors [Schneider and Eberly, 2003]:

$$\hat{d} = \hat{u} \times \hat{v}. \tag{2.11}$$

The points on the cylinder barrel are parameterised by:

$$O(\theta, t) = p_c + (r \cdot \cos\theta)\hat{u} + (r \cdot sin\theta)\hat{v} + t\hat{d}, \theta \in [0, 2\pi], |t| \leq h/2. \tag{2.12}$$

The end disks are parameterised by:

$$O(\theta, \rho) = p_c + (\rho \cdot \cos\theta)\hat{u} + (\rho \cdot sin\theta)\hat{v} \pm (h/2)\hat{d}, \theta \in [0, 2\pi], \rho \leq [0, r]. \tag{2.13}$$

The projections of a cylinder (onto a line or plane) are determined by the cylinder wall and not the end disks and the quadratic representation of a cylinder wall is defined as:

$$(O - p_c)^\top = (\mathbf{I} - \hat{d}\hat{d}^\top)(O - p_c) = r^2 \tag{2.14}$$

and the boundness of the cylinder is specified by:

$$|\hat{d} \cdot (O - p_c)| \leq h/2. \tag{2.15}$$

[1]Note that a quadratic equation of the form $ax^2 + bx + c = 0$ has the solutions $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$.

Cone

The cone is the next geometric primitive to be introduced. The cone has a vertex V, axis direction vector $\hat{a}$, and an angle θ between the axis and the outer edge [Schneider and Eberly, 2003]. In most applications, the cone is acute, and so: $\theta \leq [0, \pi/2]$. Assuming that, in fact, the cone is acute, so $cos\theta > 0$. The algebraic condition is:

$$\hat{a} \cdot \left(\frac{X - V}{\|X - V\|} \right) = \cos\theta. \tag{2.16}$$

The quadratic representation is:

$$(\hat{a} \cdot (X - V))^2 = (\cos^2\theta)\|X - V\|^2. \tag{2.17}$$

Torus

A torus as geometric primitive is a quadric surface (4^{th} degree), most commonly known as a "doughnut". A torus may be defined by rotating a circle about an axis lying in the plane of the circle [Schneider and Eberly, 2003]. There are different alternative representations possible and two different implicit definitions:

$$\begin{aligned} &x^4 + y^4 + z^4 + x^2y^2 + 2x^2z^2 + 2y^2z^2 - 2(r_0^2 + r_1^2)x^2 \\ &+ 2(r_0^2 - r_1^2)y^2 - 2(r_0^2 + r_1^2)z^2 + (r_0^2 - r_1^2)^2 = 0 \end{aligned} \tag{2.18}$$

and

$$(r_0 - \sqrt{x^2 + y^2})^2 + z^2 = r_1^2 \tag{2.19}$$

and as a parametric definition:

$$\begin{aligned} x &= (r_0 + r_1 \cos v)\cos u, \\ y &= (r_0 + r_1 \cos v)\sin u, \\ z &= r_1 \sin u, \end{aligned} \tag{2.20}$$

where r_0 is the radius from the centre of the torus to the centre of the tube of the torus and r_1 is the radius of the tube itself. In general, the major radius is bigger than the minor radius ($r_0 > r_1$); this corresponds to a ring torus; the others are horn torus $r_0 = r_1$ and the self-intersection spindle torus $r_0 < r_1$[2].

Cube

Lastly, the geometric primitive the cube is presented. A cube has a special role among geometric primitives, because it is not a rotationally symmetric object. As description of the whole surface of a cube in 3D space, a centre point p_c, and the orientation of defined axis, is required. Afterwards, all plane parameters can be derived. Thus a least-squares

[2] Weissenstein, 1999, http://mathworld.wolfram.com/Torus.html

fit of a cube in point clouds is not as simple as with the other primitives. In this case each plane has to be fitted using an orthogonal distance from a point p to a line or respectively to a plane. Note that a cube can be more easily described with a more complex model where a closed distance function is used, e.g. superquadrics.

2.1.2 Generalised Cylinders

A fundamental problem in 3D computer vision is the recovery of 3D shapes from image data. One way to simplify the presentation are generalised cylinders (GCs), which combine volume and surface information quite concisely, as illustrated in Fig. 2.2(a). Fig. 2.2(b) shows that generalised cylinders can be used to model more complex objects. Generalised cylinders are the first dedicated part-level models in computer vision [Zerroug, 1999]. A generalised cylinder is composed of an axis, a base cross-section, and a sweeping rule which describes, in closed-form, how the cross-section evolves along the axis of the cylinder. If the cross-section is perpendicular to the axis, generalised cylinders are referred to as Right Generalised Cylinders (RGCs) [Dior et al., 1997, Williams et al., 1997].

The definition of the axis a as a function of the arc length s in a fixed coordinate system (x, y, z) is:

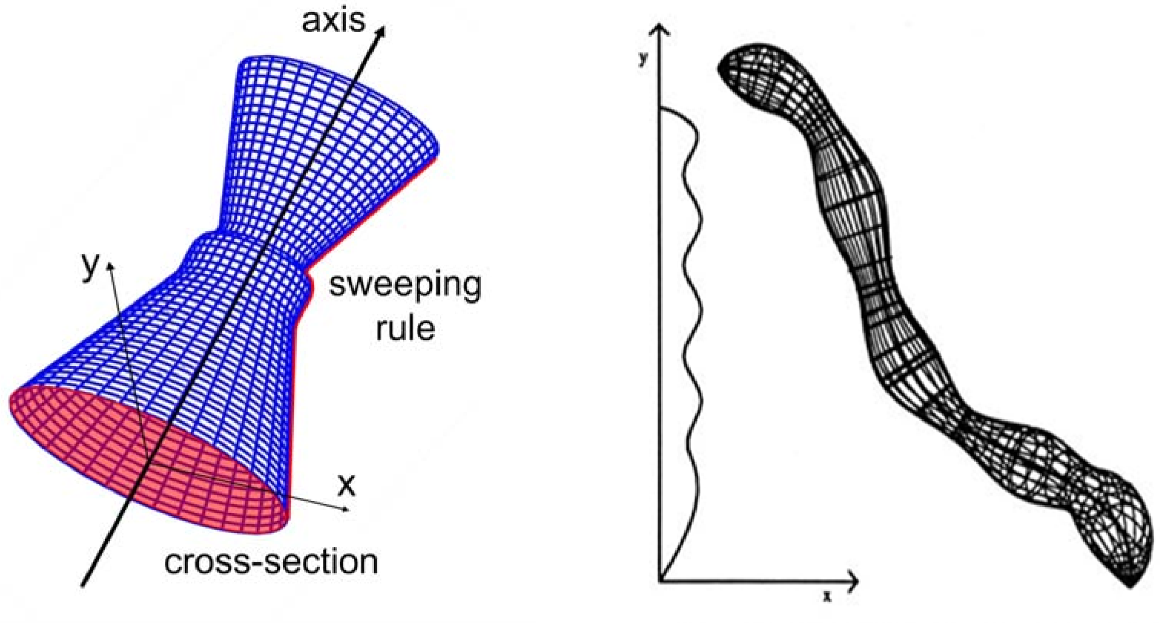
$$a(s) = (x(s), y(s), z(s)) \,. \tag{2.21}$$

The sweeping rule is more conveniently expressed in a local coordinate system. It can be defined by a cross-section boundary, which is parameterised by a parameter r:

$$sweeping\ rule = (x(r, s), y(r, s)). \tag{2.22}$$

To limit the complexity and to simplify the recovery of generalised cylinders from range images, constraints are often used. Straight axes and a constant sweeping rule are the main constraints added to generalised cylinders. Generalised cylinders are particularly attractive for the modelling of elongated shapes, because the axis of generalised cylinders often provides an intuitive method to conceptualise the design of a shape [Pillow et al., 1994]. The recovery of generalised cylinders from intensity images has been studied by many researchers and the expressiveness of generalised cylinder representation makes it well suited for visual representation [Grisoni and Marchal, 2003, Rao and Nevatia, 1988, Mohan and Nevatia, 1989, Zerrough and Nevatia, 1999]. [Pan et al., 2005] presented a parametric reconstruction of generalised cylinders. The reconstruction is achieved by some general assumptions on GCs and can, therefore, be applied to a broader subclass of GCs.

The recovery of generalised cylinders from intensity images is a complex issue, because it must rely on complicated rules to group low-level image features (edges, corners, and surface normals) to finally assemble them into generalised cylinders. The problems are based on the complicated parameterisation of generalised cylinders and the lack of a fitting function to provide a direct evaluation criterion on how well the model generalised cylinder fits the image data [Jaklic et al., 2000]. Only the recovery of a restricted subset of generalised cylinders such as straight homogeneous generalised cylinders [Ponce et al., 1989] has been presented in literature so far.



(a) (b)

Figure 2.2: Generalised cylinders: **(a)** *A generalised cylinder is defined as a volume formed by sweeping a cross-section along an axis.* **(b)** *A generalised cylinder with an arbitrary shaped axis and the displayed sweeping function [Coquillart, 1987].*

2.2 Edge Detection

Edge detection is a popular and well-investigated issue in computer vision. Fig. 2.3 illustrates the main types of edges. One of the oldest edge detectors is the Roberts opera-

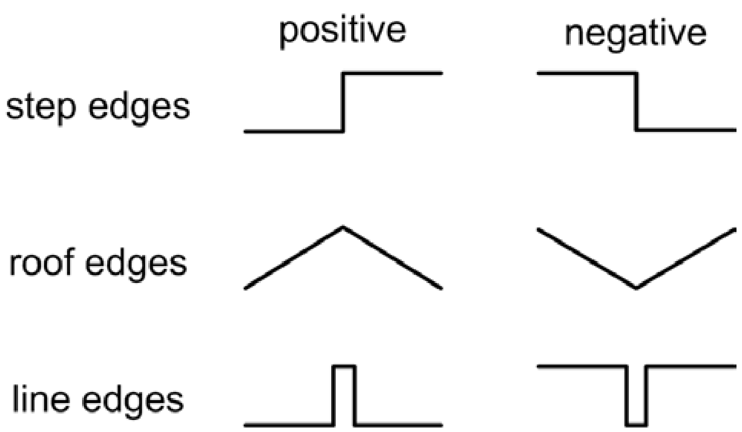


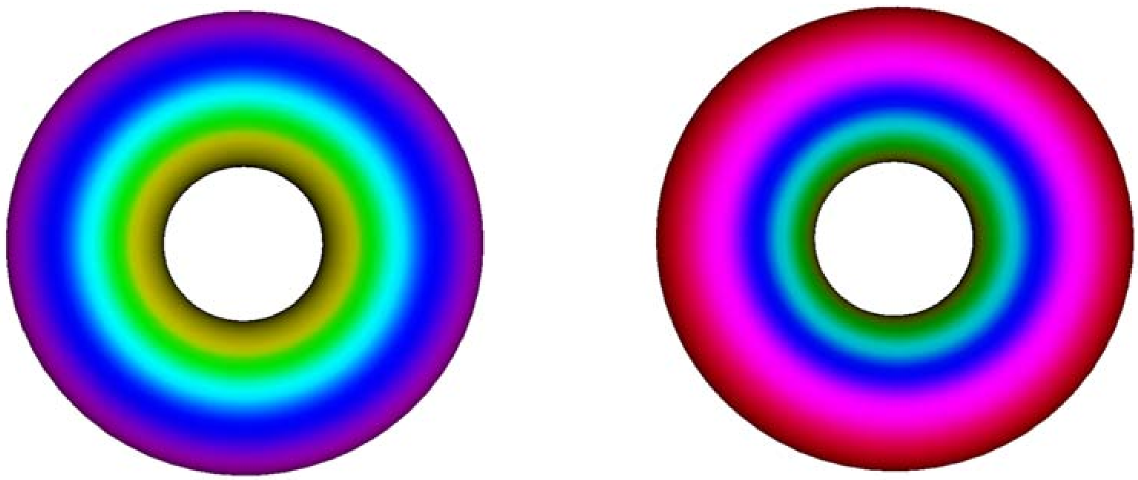
Figure 2.3: Types of edges: step edges, roof edges, and line edges with positive and negative direction.

tor [Roberts, 1965] which calculates the magnitude of the gradient using the approximated first derivative. This operator is very fast but sensitive to noise. Smoothing the raw data with a Gaussian filter and calculating the zero-crossings of the second derivatives improves the robustness as shown with the Marr-Hildreth edge detector [Marr and Hildreth, 1980]. The most popular operator is the Canny edge detector [Canny, 1986], well suited for noisy step edges. That is a multi-scale Gaussian-smoothed approach finding the locally strongest gradient using the second derivatives and is, therefore, computationally time-consuming.

A totally different approach that is well-suited for range images is the scan line approximation [Jiang and Bunke, 1999]. The raw data points are approximated by a set of bivariate polynomial functions, where the discontinuity of the fitted functions indicate the edge position. An improved scan line approach, better handling outliers, is proposed by Katsoulas et al. [Katsoulas and Werber, 2004] using an additional statistical merging step. Based on these techniques, Section 3.1 introduces a real-time edge detection method by voting the results of different detection methods to achieve more robust results.

2.3 Curvature Analysis

Estimating intrinsic geometric properties of a surface from range images is an important part of numerous algorithms in machine vision. Generally, two different approaches are classified to represent an object: a volumetric and a boundary-based method. A volumetric description utilises global characteristics of a 3D object (e.g. principal axis, inertia matrix or tensor-based moment functions). Boundary-based methods describe an object with properties of the boundary and their relationships. Segmentation, object recognition and detection tasks are mainly based on the differential invariant properties by estimating the Gaussian curvature K and the mean curvature H [Trucco and Verri, 1998] as illustrated in Fig. 2.4. The two principal curvatures at a point p of a surface S measure how the surface



(a) (b)

Figure 2.4: Curvature representation: **(a)** *Gaussian curvature.* **(b)** *Mean curvature. Example code from the freely available VTK library, http://public.kitware.com/vtk.*

bends by different amounts in different directions at that point. A normal plane at a point p on a differentiable surface S also contains a unique direction tangent to the surface and cuts the surface into a plane curve, $p \in S$. This curve contains different curvatures for different normal planes at p and κ_1 and κ_2 are the maximum and minimum values of this curvature.

Hereby the Gaussian curvature K is defined as:

$$K = \kappa_1 \kappa_2. \tag{2.23}$$

Another presentation is given by:

$$K(p) = det(S(p)), \tag{2.24}$$

whereby S is the shape operator.

The mean curvature H is defined as:

$$H = \frac{\kappa_1 + \kappa_2}{2}. \tag{2.25}$$

A more general presentation is:

$$H = \frac{1}{n} \sum_{i=1}^{n} \kappa_i. \tag{2.26}$$

[Surazhsky et al., 2003] analysed different computation schemes for local estimation of intrinsic curvature of the geometric properties. The algorithms and their modifications were tested on triangular meshes and compared with the Gaussian and mean curvatures of the non uniform rational B-spline surfaces (NURBs). It shows that the best algorithm for the Gaussian curvature estimation is the Gauss-Bonnet scheme and for the mean curvature it is the paraboloid fit method.

2.4 Range Image Segmentation

Segmentation is a crucial task to separate range images into useful parts [Hoover et al., 1995]. It is obvious from the results that there exists no perfect segmentation algorithm. Each algorithm has its own benefits depending on the task. On the whole, a good segmentation result is given if regions can be approximated from a given set of surface functions. To realise this step, two main different approaches exist: region-based and edge-based range image segmentation. Other approaches are: segmentation of range data into planar surfaces, curved surfaces, and surfaces of revolution. Region-based segmentation methods search for homogenous regions based on surface approximation and edge-based methods aim to segment boundaries between homogenous image regions. One drawback is that edge-based range segmentation algorithms often produce gaps between boundaries so region-based methods are more frequently used in practice. Additionally, the discontinuities of curved surfaces are smooth and hard to locate, and therefore edge-based range segmentation algorithms tend to under-segment range images. [Hoover et al., 1995, Hoover et al., 1996] presented a method to evaluate the results of different range image segmentation algorithms. They developed a tool to objectively compare a machine-generated segmentation against the specified ground truth.

2.4.1 Region-Based Range Image Segmentation

Region-based range image segmentation algorithms can be categorised into two main groups: parametric model-based range segmentation algorithms and region-growing algorithms. Parametric segmentation algorithms make use of parametric surface models and group data points to find out which points are part of the assumed models [Bab-Hadiashar and Gheissari, 2006]. Region-growing algorithms detect groups of neighbouring points based on homogeneous criteria. These criteria are formulated so that the segmented regions correspond to the surfaces of an object. Normally this step results in an over-segmentation. This failure can be adjusted with an additional merging step [Djebali et al., 2002]. In some cases the result is extended by employing an additional region-growing strategy. Different methods based on iterative or random methods can be used to obtain the initial regions.

2.4.2 Edge-Based Range Image Segmentation

These algorithms are based on edge detection and the labelling of edges to extract edges from a range image in order to find discontinuities of regions. Different edge types and methods are described in Section 2.2 and Section 2.3. After the edge detection step, edges with defined common properties are clustered together. [Fan et al., 1989] presented a typical method based on the detection of discontinuities using zero-crossing and curvature values. Here the range image is segmented at discontinuities to obtain an initial segmentation.

2.4.3 Object-Based Range Image Segmentation

Different methods for automatic 3D object segmentation into meaningful parts have been published in the last few years. 3D Model Segmentation algorithms can be categorised into two main classes. The first class is developed for applications like reverse engineering of CAD models [Attene et al., 2006b]. The second class aims to segment natural objects into meaningful parts. Most work on mesh segmentation, which is described in detail in Section 2.6 is based on iterative clustering. This method generates a mesh and segments all disconnected fragments with a face (triangle) connectivity filter. [Shlafman et al., 2002] segmented models into meaningful pieces using k-means clustering. Based on this idea, [Katz and Tal, 2003] developed a fuzzy clustering method and used minimal boundary cuts to achieve smoother boundaries between clusters. Unsupervised clustering techniques like mean shift can also be applied to mesh segmentation [Shamir et al., 2004]. [Cornea et al., 2005] published a method using skeletons to generate a hierarchical mesh decomposition. Meanwhile, [Katz et al., 2005] published a mesh segmentation algorithm based on pose-invariant models and the extraction of the core part and feature points. The method is able to produce consistent results, but works only on meshes. A computation intensive method is used to find feature points to limit the complexity and the number of parts.

2.5 Model-Fitting

The main fitting methods in literature can be divided into two general techniques: the least-squares fitting [Lawson and Hanson, 1974] and the clustering [Besl and Jain, 1985] method. The least-squares methods are focused on finding the sets of parameters to minimise the distance between the data points and the curve or surface; the clustering methods are based on mapping data points to the parameter space, such as the Hough transform [Hough, 1962] or the accumulation methods. The least-squares method is the preferred technique of fitting geometric shapes to 3D point clouds, because the clustering methods are time-consuming and computationally expensive and thus mainly used to fit lines and curves to point data in 2D space [Duda and Hart, 1972]. Least-squares fitting is reviewed in detail in this section for this reason.

2.5.1 Least-Squares Methods

Based on a distance measure, the least-squares technique calculates the least-squares sum of the range data points to the geometric model. Several metrics exist: the algebraic distance, the euclidean distance, and the taubin approximation within the objective function of the minimisation process. [Faber and Fischer, 2001b] illustrated that the best results regarding quality and accuracy is obtained by using the euclidean distance. In contrast the taubin approximation [Taubin, 1991, Taubin, 1993] or the algebraic distance [Fitzgibbon and Fisher, 1995] lead to a bias in the fitting results. Hence this section reviews only the euclidian distance function.

Euclidean Distance

Euclidean distances are invariant to transformations in euclidean space and do not exhibit the high curvature bias. The euclidean distance d_E between a point p_i and a surface is the distance between p_i and the surface point p_E whose tangent is orthogonal to the line joining p_i and p_E.

For primitive surfaces like planes, spheres, cylinders, cones and toroids, a closed form expression exists for the euclidean distance from a point p_i to the zero set. However, the expression of the euclidean distance to other curves or surfaces like superquadric or geons is more complicated and there exists no closed form, therefore an iterative procedure to estimate the euclidean distance must be carried out [Faber and Fischer, 2001a].

2.5.2 Model-Based Cylinder Detection

This section reviews the state-of-the-art model-based object detection using cylinders as members of geometric primitives and focuses on object detection dealing with 3D point clouds acquired from laser range data.

In literature, many curvature-based approaches have been introduced to detect cylindrical objects in range images. [Yokoya and Levine, 1989] achieved an improved detection result by using a hybrid approach combining the mean H and the Gaussian curvature K segmentation with a step and roof edge detection. The work of [Hameiri and Shimshoni, 2003] is based on principal curvature histogrammes for cylinder fragment detection. Another approach of analysing the Gaussian image and the convexity of surface patches was proposed by [Taylor and Kleeman, 2003]. In contrast to curvature-based approaches, [Marshall et al., 2001] follows an approach by fitting least-squares models to segment a scene for object detection. [Attene et al., 2006a] approximates the object with cylinders based on a mesh representation. The idea of the robust axis determination of rotationally symmetric parts [Yacoub and Menard, 1997] is used for an improved and advanced determination of the cylinder pose. Chapter 4 introduces a sample-based approach, such as the RANSAC algorithm [Fischler and Bolles, 1981] for a cylinder estimation, which satisfies the required accuracy, short processing time and also operating on sparse data.

2.6 Mesh Processing

"I hate meshes. I cannot believe how hard this is. Geometry is hard."
– *David Baraff, Senior Research Scientist, Pixar Animation Studios*

This section exposes the basics of surface approximation based on point clouds with 3D meshes and the way how to process such meshes with different operators. Since mesh processing is a growing field in 3D computer vision and this section reviews the state-of-the-art methods used today. A triangulated mesh is a discrete structure that can be used to approximate a surface in Euclidian space $\mathbb{R}^k$.

2.6.1 Mesh Generation

Mesh generation is an important practice of generating shapes from data points captured from real objects by laser range scanners. A triangulated mesh is a discrete structure, which can be used to approximate a surface which contains triangles or other simple convex polygons in euclidian space. It consists of three types of geometric components: *vertices*, *edges*, and *faces*. Vertices are points which describe the corners or intersections of geometric shapes. Edges are lines whose end points are vertices and faces are convex polygons. A finite collection of vertices, edges, and faces is called: *polygonal mesh* or *polymesh* with the following conditions [Schneider and Eberly, 2003]:

- Each vertex must be shared by at least one edge.
- Each edge must be shared by at least one face.

- If two faces intersect, the vertex or edge of intersection must be a component in the mesh.

If all faces are triangles, the model is called a triangle mesh or trimesh as illustrated in Fig. 2.5. If the mesh has no multiple connected sub-meshes, the mesh is said to be

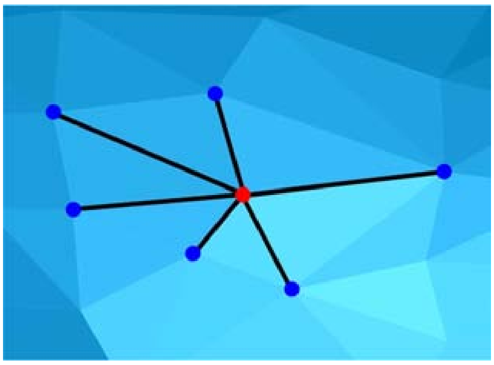


Figure 2.5: A triangle mesh: The vertex of the intersection (coloured red) and all surrounding faces.

connected. A *connected mesh* is said to be a *manifold mesh* if each edge in the mesh is shared by at most two faces. Also, a connected mesh is said to be a *closed mesh* if it is manifold with each edge shared by exactly two faces and is non-self-intersecting, otherwise it is an *open mesh*.

In general, it will be differentiated between *structured* and *unstructured* meshes. In a structured mesh all elements have the topology of a regular grid; Unstructured meshes are often computed by using quadtrees or a Delaunay triangulation of point clouds. The following sections describe the basic knowledge needed for mesh processing to understand the procedure which will be further adjusted in this work.

Voronoi Diagram

A Voronoi diagram or Dirichlet tessellation is the decomposition of a metric space determined by distances to a specified discrete set of points in the space as illustrated in Fig. 2.6(a). Voronoi diagrams find widespread applications in areas such as nearest neighbour clustering, facility location, path planning or crystallography.

Let $P = (p_1, p_2, ..., p_n)$ be a set of points in the two-dimensional Euclidean plane called *sites*. The plane is then partitioned by assigning every point in the plane to its nearest site. All those points assigned to p_i from the Voronoi region $V(p_i)$[3]:

$$V(p_i) = \{x : |p_i - x| \leq |p_j - x| \, \forall j \neq i\}. \tag{2.27}$$

The condition in this case is that this site has to be closed. If P contains only three points: p_1, p_2, and p_3, the diagram contains the bisectors B_{12}, B_{23}, and B_{31}. In this case, the perpendicular bisectors of the three sides of a triangle all pass through one point, the

[3]The Voronoi region is not a polygon by our definition of polygon, because it might be unbounded.

circum centre, the centre of the unique circle that passes through the triangle's vertices. However, the circumcentre of a triangle is not always inside the triangle.

If $H(p_i, p_j)$ is a closed half plane with boundary B_{ij} and containing the point p_i, then $H(p_i, p_j)$ can be considered as all the points that are closer to p_i as to p_j. Hence the Voronoi region $V(p_i)$ can be described as:

$$V(p_i) = \bigcap_{i \neq j} H(p_i, p_j), \tag{2.28}$$

where the notation implies that the intersection is to be taken over all i and j provided that $i \neq j$. Note that the Voronoi regions are convex and, if the regions are bounded, they are convex polygons. The edges are called Voronoi edges and the vertices Voronoi vertices, at which a point on the interior of a Voronoi edge has two nearest sites, and a Voronoi vertex has at least three nearest sites [O'Rourke, 1998].

One of the main characteristics of a Voronoi diagram is the correspondence of the dual graph of a Voronoi diagram to the Delaunay triangulation for the same set of points P as illustrated in Fig. 2.6(c). It can also be defined as a triangulation of the sites with the property that for each triangle the circumcircle $C(v)$ (v is the centre of the circle $C(v)$) does not contain any other sites.

Other properties are:

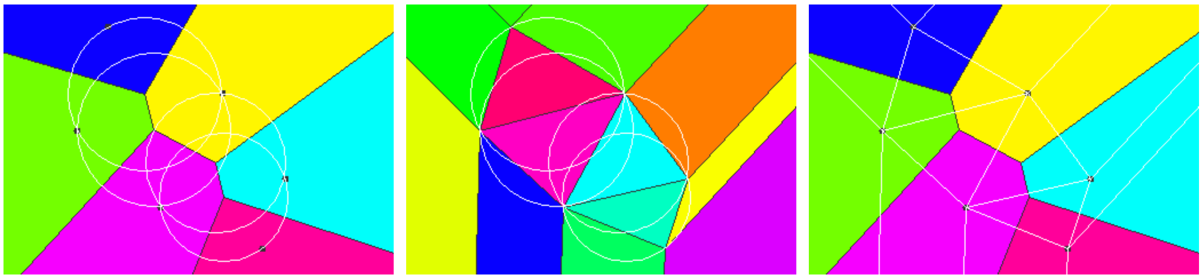
- Each Voronoi region $V(p_i)$ is convex.
- If p_j is the nearest neighbour to p_i, then $(p_i,\ p_j)$ is an edge of the Delaunay triangulation $D(P)$.
- If there is a circle through p_i and p_j that contains no other sites, then $(p_i,\ p_j)$ is an edge of $D(P)$.

The Voronoi cells can also be defined by measuring distances to objects that are not points. In this case the Voronoi diagram with these cells is also called the medial axis. The medial axis is used in segmentation, recognition and other computational applications, and a simplified version of the Voronoi diagram of line segments is the straight skeleton [O'Rourke, 1998].

Delaunay Triangulation

The Voronoi diagram $V(P)$ and the Delaunay triangulation $D(P)$ are dual structures as illustrated in Fig. 2.6(b). Each contains the same information, but represents it in another form. As defined by [O'Rourke, 1998] the properties of a Delaunay triangulation are:

- $D(P)$ is the straight-line dual of $V(P)$.
- $D(P)$ is a triangulation, if no four points of P are cocircular. Every face is a triangle. This is Delaunay's theorem. The faces of $D(P)$ are called *Delaunay triangles*.
- Each face (triangle) of $D(P)$ corresponds to a vertex of $V(P)$.



(a) (b) (c)

Figure 2.6: Voronoi diagram and Delaunay triangulation of six points: **(a)** *Voronoi diagram with circum circles of the triangles.* **(b)** *Delaunay triangulation with circum circles of the triangles.* **(c)** *Correspondence between Voronoi diagram and Delaunay triangulation. Illustrated with the "Voronoi diagram / Delaunay triangulation" program by Paul Chew.*

- Each edge of $D(P)$ corresponds to an edge of $V(P)$.
- Each node of $D(P)$ corresponds to a region of $V(P)$.
- The boundary of $D(P)$ is the convex hull of the sites.
- The interior of each (triangle) face of $D(P)$ contains no sites.

Chapter 5 describes a potential application of the Delaunay triangulation to segment different objects on a table. At first all table points are detected with a RANSAC-based [Fischler and Bolles, 1981] plane fit and all remaining points are used to realise a mesh based on the Delaunay triangulation. All segments (objects) of the generated mesh are extracted from the mesh by a triangle connectivity filter. This step segments the mesh into different components.

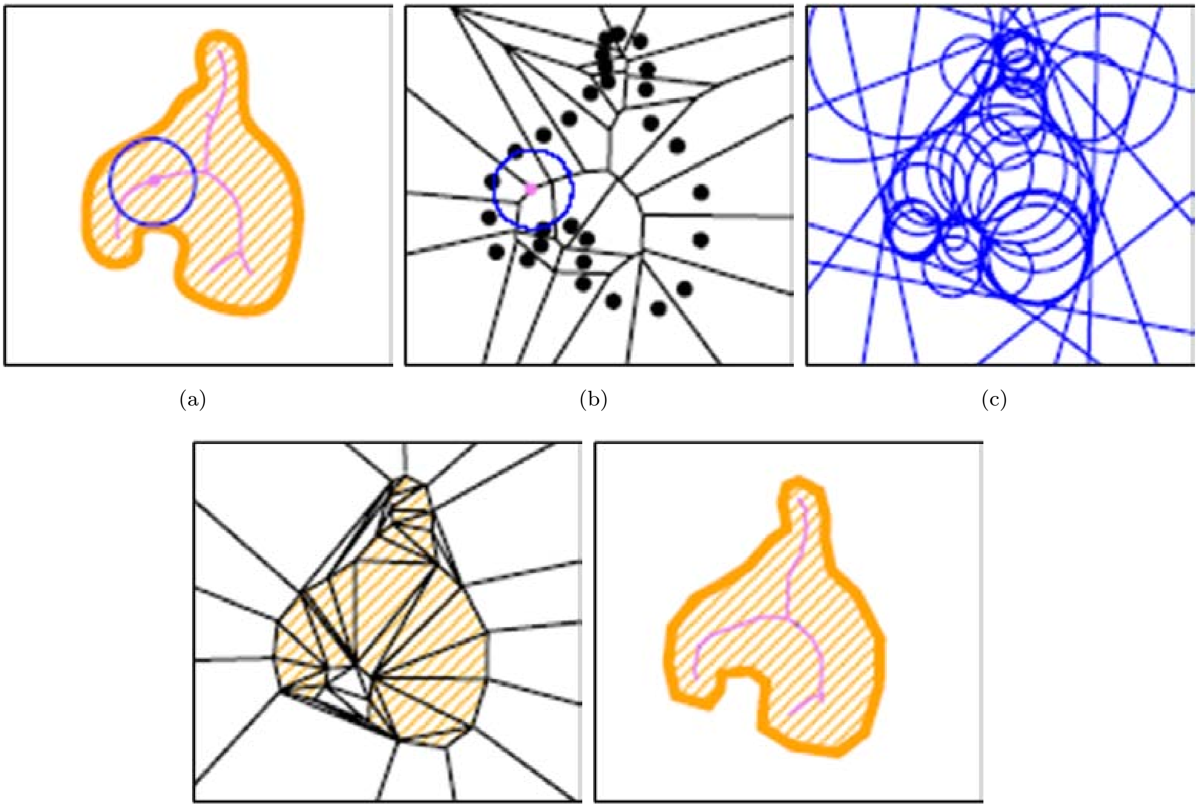
Surface Reconstruction

A high amount of surface reconstruction methods can be found by reviewing the research literature. [Boissonnat, 1984] presents a method which labels a subset of the Delaunay tetrahedra of the surface S as the interior of the solid. Another algorithm based on the Delaunay triangulation is the α-shape algorithm of [Edelsbrunner and Mücke, 1994]. Hereby α-shapes are a generalisation of the convex hull[4] of a point set and for $\alpha = \infty$, the α-shape is identical to the convex hull. The disadvantage of α-shapes for surface reconstruction is as follows: if the sampling is non-uniform it is sometimes impossible to choose α to balance hole-filling against loss of detail. [Bernardini et al., 1999] developed a system based on α-shapes while avoiding the computation of the Voronoi diagram, making reconstruction of larger models possible.

The power crust algorithm for surface reconstruction [Amenta et al., 2001] of 3D models is a very interesting method, because this algorithm delivers very good results and is quite fast. It realises a construction which takes a sample of points from the surface of

[4] A convex hull for a set of points w in a real vector space is the minimal convex set containing w.

a 3D object and produces a surface mesh and an approximate medial surface axis. The approach approximates the medial axis transform (MAT) of the object. The medial axis transform is a skeletal shape representation of the object. It represents a solid by the set of maximal balls completely included in the interior space rather than by the set of points on the boundary. Then it uses an inverse transform to produce the surface representation of the MAT. Fig. 2.7[5] represents a two-dimensional version of the algorithm. The



(a) (b) (c)

(d) (e)

Figure 2.7: Two-dimensional example of the power crust reconstruction method. **(a)** *An object with its medial axis. One maximal interior ball is shown.* **(b)** *The Voronoi diagram of S with the Voronoi ball surrounding one pole. In 2D it is possible to select all Voronoi vertices as poles, but not in 3D.* **(c)** *The inner and outer polar balls. Outer polar balls with centres at infinity degenerate to half spaces on the convex hull.* **(d)** *The power diagram cells of the poles.* **(e)** *The power crust and the power shape of its interior solid [Amenta et al., 2001].*

disadvantage of this algorithm is the sensitivity to noise and outliers. [Kolluri et al., 2004] developed the noise-resistant *eigencrust* algorithm for reconstructing a watertight surface from point cloud data. The output of this algorithm is a triangulated surface composed of triangulated faces, where an inside tetrahedron meets an outside tetrahedron. This

[5] I would like to thank Prof. Nina Amenta from the University of California department of computer science for providing Fig. 2.7

procedure guarantees that the output surface is watertight.

2.6.2 Manifold Geometry of Surfaces

This section studies the numerical computation of geodesic distances on Riemannian manifolds. In Riemannian geometry, a Riemannian manifold[6] is a real differentiable manifold, in which each tangent space is equipped with an inner product, which varies smoothly from point to point. The metric is a positive definite metric tensor, which allows one to define various notions such as angles, lengths of curves, areas, volumes, curvatures, gradients of functions, and divergence of vector fields [Hirsch, 1997].

Riemannian Manifold

A parameterised surface is embedded in some Euclidean domain $\mathbb{R}^k$, which allows the definition of a local metric based on the first fundamental form I_φ. For an embedded manifold $\mathcal{M} \subset \mathbb{R}^k$, the first fundamental form is [Gray et al., 1997]:

$$I_\varphi = \left(\frac{\partial \varphi}{\partial u_i}, \frac{\partial \varphi}{\partial u_j} \right)_{i,j=1,2}. \tag{2.29}$$

It is possible to consider directly a field of positive definite tensors on a parametric domain $\mathcal{D} = \mathbb{R}^n$ (normally: n = 2 for surfaces and n = 3 for volumes). A Riemannian manifold is an abstract parametric space $\mathcal{M} \subset \mathbb{R}^n$ equipped with a metric $x \in \mathcal{M} \mapsto H(x) \in \mathbb{R}^{n \times n}$ positive definite. Using the Riemannian metric one can compute the length of a curve $(\gamma(t))_{t=0}^{T}$:

$$L(\gamma) = \int_0^T \sqrt{\gamma'(t)^T H(\gamma(t))\gamma'(t)}dt. \tag{2.30}$$

Geodesic Distances

The local Riemannian metric allows the definition of a global metric on the space $\mathcal{M}$ using the shortest path, which corresponds to geodesic distances. A geodesic on a surface is a curve connecting two points that is shorter than any other curve that connects the two points. This is known as a *geodesic curve*. The arc length of that curve is called the *geodesic distance* [Schneider and Eberly, 2003]. For two points on a plane the shortest distance is the line connecting the points. For example, an ellipsoid has two paths between antipodal points and a sphere which has infinitely many paths between two antipodal points. In general, this is a difficult problem for smooth surfaces. The problem becomes simpler when restricted to manifold triangle meshes, but it is still difficult to implement.

2.6.3 Multi-Dimensional Scaling

To realise a pose-invariant mesh representation, multi-dimensional scaling (MDS) is used. MDS [Cazals et al., 2009] is a generic name for a family of algorithms that constructs

[6] A manifold is a mathematical space that on a small enough scale resembles the Euclidean space of a certain dimension.

a configuration of points in a target metric space from information about inter-point distances (dissimilarities), measured in some other metric space [Bronstein et al., 2006, Bronstein et al., 2008, Rosman et al., 2008]. In the experiments, dissimilarities are defined as geodesic distances δ_{ij} between all vertices v_i on the mesh $\mathcal{M}$ in a symmetrical dissimilarities matrix $\Delta = \mathcal{N} \times \mathcal{N}$ between N points on a Riemannian manifold $\mathcal{S}$.

Methods to calculate the dissimilarity matrix more effectively are based on the fast marching method on triangulated domains [Kimmel and Sethian, 1998] or parametric fast marching [Spira and Kimmel, 2004].

It will be differentiated between metric and non-metric MDS (Shephard-Kruskal). Metric MDS (see Eq. (2.31)) preserves the intervals and the ratios between the dissimilarities while the non-metric MDS (see Eq. (2.32)) only preserves the order of the dissimilarities. The goal is to minimise the embedding error, i.e. minimising the sum of distances between the optimally scaled data $f(\delta_{ij})$ and the euclidean distances d_{ij}, where f is an optimal monotonic function in order to obtain optimally scaled similarities. Thereby a stress function $\mathcal{F}_s$ will be used to measure the degree of correspondence of the distances between vertices given by:

$$\mathcal{F}_{s_m} = \sqrt{\sum_{i \neq j} (f(\delta_{ij}) - d_{ij})^2}, \tag{2.31}$$

$$\mathcal{F}_{s_{nm}} = \sqrt{\frac{\sum_{i \neq j} (f(\delta_{ij}) - d_{ij})^2}{\sum_{i \neq j} d_{ij}^2}} \tag{2.32}$$

and each vertex in MDS space corresponds to a vertex in euclidean space.

[Bronstein et al., 2006], [Rosman et al., 2008] developed one of the most efficient algorithms to minimise the stress using an iterative optimisation algorithm (see Eq. (2.33)), the scaled gradient descent algorithm (SMACOF). A short basic description of the algorithm is given as follows:

$$min_x \sum_{i<j} (d_{ij}(X) - \delta_{ij})^2. \tag{2.33}$$

The gradient of the stress function is:

$$\nabla_X s(X) = 2UX - 2B(X)X, \tag{2.34}$$

where U denotes matrix pseudoinverse,

$$u_{ij} = \begin{cases} -1 & \text{if} \quad i \neq j \\ N-1 & \text{if} \quad i = j \end{cases} \tag{2.35}$$

and $B(X)$ is a $N \times N$ matrix depending on X with the elements:

$$b_{ij} = \begin{cases} -\delta_{ij} d_{ij}^{-1}(X) & \text{if} \quad i = j \quad \text{and} \quad d_{ij} \neq 0 \\ 0 & \text{if} \quad i = j \quad \text{and} \quad d_{ij} = 0 \\ -\sum_{j \neq i} b_{ij} & \text{if} \quad i = j \end{cases}. \tag{2.36}$$

The gradient descent step can be performed with a multiplicative update:

$$X^{(k+1)} = U^{\dagger} B(X^{(k)}) X^{(k)}. \tag{2.37}$$

The result is a monotonous non-increasing sequence of stress values and is equivalent to a steepest scaled descent iteration with constant step size. [Rosman et al., 2008] describe this algorithm to realise MDS in greater detail.

2.7 Grasping

In the last few decades, the problem of grasping novel objects in a fully automatic way has gained increasing importance in robotics. The following literature review focuses on the grasping of objects from a biological, psychological, and a technical view.

2.7.1 Biological and Psychological View

This section reviews the selective processing of visual input information for manual reaching and grasping manipulations. First, a biological fundamental model of infant learning is discussed, followed by different psychological views. The psychological point of view analyses approaches in object learning by object interacting. Furthermore, the influences of the object pose for grasping and the effects of potential obstacles on the table during grasping are discussed. An explicit connection between these views and the developed grasping method is presented in Chapter 5. Thus the biological and psychological findings have been integrated in the algorithm for the grasping of unknown objects.

After observing infant movements for a long time the ILGM (Infant Learning to Grasp Model) found out that early arm movements of infants are related to the development of object direction reaching which finally leads to grasping. Infants prefer to grasp an object by visualising the top surface of the object, thus the final grasping pose and the trajectory to reach the object are directed towards the form and characteristics of the top surface [von Hofsten, 1982]. The term ILGA (Infant Learning Grasping and Affordances) contains the visual input provided by the shape of the target object. Visual feedback is used to bring the fingers into contact with shape features like handles, top surfaces or task-specific parts of the object.

[Fagg and Arbib, 1998] developed the FARS model, which focuses especially on the action-execution step. [Aarno et al., 2007] presented an idea that the robot should, like a human infant, learn about objects by interacting with them, forming representations of the objects and their categories.

The brain has often been characterised as a sensory-motor interface that selects visuospatial[7] information about the environment and transforms it into goal-directed movements. Grasping different objects in different poses is a difficult task, for which humans have developed a wide variety of movement patterns in the course of their evolution [Kuhtz-Buschbeck et al., 1998]. Vision is the most important source of information needed to successfully interact with objects [Schneiberg et al., 2002], and it is impossible to prepare arbitrarily many movement goals at once.

[7] *"Pertaining to the ability to comprehend visual representations and their spatial relationships" [Mosby, 2009].*

Grasping an object starts with a reach-to-grasp movement that brings the hand close to the object of interest, followed by a second phase in which the appropriate grasp type is formed and the movement is piloted to the points of application [Smeets and Brenner, 1999]. In a manipulation phase, the grasp is stabilised and grip forces are continuously adjusted to compensate for the changing forces of gravity. [Schulz et al., 2005] developed a prosthetic hand that closely approximates the grasping abilities of a human hand. The used five-finger hand has 15 DOF driven by small-sized flexible fluidic actuators.

[Smeets and Brenner, 1999] considered the multiple movement goals with a pattern analysis where the thumb and index finger point towards selected positions on the surface of an object. Object features, such as: weight, surfaces or the centre of gravity are relevant object information for a successful grasping process and cannot be obtained only by visual information.

[Johansson et al., 2001] studied eye-hand coordination during grasping. During the experiments their participants initially had to grasp a bar and then to use it to press a button without colliding with an obstacle along the transportation path. However, [Brouwer et al., 2009] pointed out that in this task setup only one point of application was visible to the actor, the second contact point was occluded by the rear side of the object. From this point of view, the study of course cannot resolve the question of which surface parts of an object are visually selected before or during grasping movements. [Brouwer et al., 2009] further investigated the fixation behaviour of human actors during grasping under conditions where all potential grasping points of application were clearly visible. An interesting result of their work is that open fixation behaviour in a grasping task becomes different from gaze control in a free-viewing condition. Another result is that the initial fixation can select only one of the two points of interest, the application points of index finger and thumb. They relied on the fact that thumb and index finger presumably play different roles in precision grasps. Previously it had been shown that the thumb generally guides the hand to the object, whereas the index finger seems to be used in order to regulate the hand's aperture during the final grasping phase. Nevertheless, the gaze can only be at one location at a time and the span of time to complete the reach-to-grasp movement is not sufficiently reached to scan the object. [Brouwer et al., 2009] claims that the centre of gravity of the object also has an essential effect.

Normally, we do not act on isolated objects but often in cluttered or even crowded visual environments. The presence of an obstacle plays an important role for the visual preparation and the motor programming of a successful grasp. Obstacles need to be actively avoided and therefore integrated into the action plan [Tresilian, 1998, Tresilian, 1999].

[Tresilian, 1998] showed that people generally keep their hands outside of a region that surrounds the obstacle by a minimum preferred distance. Several studies have shown that the initial reaching component takes longer if an obstacle is present than compared to the same movement executed without any obstacles [Biegstraaten et al., 2003, Jackson et al., 1995].

2.7.2 Technical View

This section summarises grasping from a technical point of view. The described methods are separated into the learning of objects, 2D and 3D input data based algorithms, path planning, a grasp simulator to check potential grasps, and grasp affordances.

[Saxena et al., 2008] have developed a learning algorithm that predicts the grasp position of an object directly as a function of its image. Their algorithm focuses on the task of identifying grasping points that are trained with labelled synthetic images of a different number of objects. [Detry et al., 2009] addressed the problem of learning and representing object grasp affordances. They analysed learned grasp hypothesis densities from both imitation and visual cues, and presented grasp empirical densities learned by a robot from physical experience. [Speth et al., 2008] propose fast 2D contour-based grasp planning algorithms. Images of the target object from different points of view are used to recover critical 3D information like the size, location and the pose.

[Kragic and Bjorkman, 2006] have analysed a vision-guided grasping system. Their approach is based on integrated monocular and binocular cues from five cameras to provide robust 3D object information. The system is applicable to well-textured, unknown objects. A three fingered hand equipped with tactile sensors is used to grasp the object in an interactive manner. [Bone et al., 2008] presented a combination of online silhouette and structured-light 3D object modelling with online grasp planning and execution with parallel-jaw grippers. Their algorithm analyses the solid model, generates a robust force closure grasp and outputs the required gripper pose for grasping the object. They consider the complete 3D model of one object segmented into single parts. After the segmentation step each single part is fitted with a simple geometric model. A learning step is finally needed in order to find the object component that humans choose to grasp.

[Stansfield, 1990] presents a system for grasping 3D objects with unknown geometry using a Salisbury robotic hand placing every object on a motorised and rotating table under a laser scanner to generate a set of 3D points combined to form a 3D model. [Wang and Jiang, 2005] have developed a framework of automatic grasping of unknown objects by using a laser range scanner and a simulation environment. [Boughorbel and Zhang, 2007] have observed industrial bin picking tasks and have developed a system that provides accurate 3D models of parts and objects in the bin to realise precise grasping operations, but their superquadrics based object modelling approach can only be used for rotationally symmetric objects. [Richtsfeld and Zillich, 2008] have published a method to calculate possible grasping points for unknown objects with the help of the flat top surfaces of the objects based on a laser range scanner system. However, different approaches for grasping quasi planar objects exist [Sanz et al., 1999]. [Huebner et al., 2008] have applied a method to envelop given 3D data points into primitive box shapes by a fit-and-split-algorithm with an efficient minimum volume bounding box. These box shapes give efficient clues for planning grasps on arbitrary objects. Another 3D model-based work is presented by [El-Khoury et al., 2007].

The problem of automatic grasp generation and planning for robotic hands have been analysed by [Ekvall and Kragic, 2007]. Shape primitives are used in synergy to provide a basis for a grasp evaluation process when the exact pose of the object is not available.

The presented algorithm calculates the approach vector based on the sensory input and, in addition, tactile information that finally results in a stable grasp. “GraspIt!”, an interactive grasp simulator for different hands, hand configurations and objects has been developed by [Miller and Allen, 2004]. In the beginning they use shape primitives, by modelling an object as a sphere, cylinder, cone or box [Miller and Knoop, 2003]. A set of rules to generate possible grasp positions is used. [Xue et al., 2008] applied this simulator for an initial grasp by combining hand pre-shapes and automatically generated approach directions. Their approach is based on a fixed relative position and orientation between the robotic hand and the object, all the contact points between the fingers and the object are efficiently found. A search process tries to improve the grasp quality by moving the fingers to the neighbouring joints and evaluates the grasp quality. [Borst et al., 2003] show that although it is not necessary in every case to generate optimal grasp positions, they reduce the number of candidate grasps by randomly generating hand configuration dependent on the object surface. Their approach works well if the goal is to find a fairly good grasp which is suitable and as fast as possible. [Goldfeder et al., 2007] present a grasp planner which considers the full range of parameters of a real hand and an arbitrary object including physical and material properties as well as environmental obstacles and forces.

A framework for the development of robotic applications on the synthesis and execution of grasps has been created by [Recatalà et al., 2008]. [Li et al., 2007] realised a data-driven approach for grasp synthesis. Their algorithm uses a database of captured human grasps to find out the best grasp by matching hand shape to object shape.

Chapter 3

Seam Following for Automated Industrial Fibre Mat Stitching

"Nothing shocks me. I'm a scientist."
– *Harrison Ford as Indiana Jones*

In 1961, the first industrial robot "Unimate" joined the assembly line at a General Motors plant to sequence and stack hot die-cast metal components [Nocks, 2007]. Since that time, the robot has been increasingly applied in different fields of industrial automation. Robots that are able to do a variety of different tasks in sequence, like picking up an object from defined places, performing manufacturing tasks such as assembly, inspection, welding or painting, and moving it to another defined place are more useful than those that can do only one job. To realise a higher flexibility of the robots and to operate in more complex production environments sensors are mounted on the robots. The most suitable sensors for a robot to work autonomously are vision systems. In the last few decades a lot of research work has been done with the aspect to realise a robotic lot-size-one production.

The introductory chapter of this thesis starts with an industrial project of automated fibre mat stitching based on range image processing dealing with low batch sizes. In this project "REDUX" (Continuous Process Chain for Robot Stitched Preforms) sensors are used to interact with the robot control unit for a fully automatic stitching process. Carbon fibre mats are draped on a model and stitched together by a sewing robot. A laser range sensor is mounted in front of the sewing head to guide the robot along the carbon fibre edges. The novel contribution to this project is a fully automated sensor guided sewing robot for carbon fibre textiles. During the stitching process, these sensors detect possible obstacles and adjust the trajectory along the edges of the mats in relation to a given distance between the seam and the edge.

This chapter summarises the defined tasks, describes the system for carbon fibre mat stitching, presents the newly-developed method to realise robust edge detection and tracking, and finally discusses the results.

3.1 Project REDUX

Today, automobile and aeronautics industry increasingly use robust and light weight parts made of Carbon Fibre Reinforced Polymer (CFRP). Besides the mechanical properties, CFRP elements have the advantage of integrating external elements easily into the CFRP elements due to the specific production process. Several different production techniques for CFRP elements are possible. However, the production of small batches or single elements is a time-consuming and complex task.

Under the leadership of EADS[1] Deutschland GmbH, a research union of industrial and scientific partners was formed in October 2005 to improve the production process of CFRP elements. The project was called REDUX[2] and the main goal was to gain a higher degree of automation during the production process. The name "REDUX" was chosen as an acronym for the German "Realisierung einer durchgehenden Prozesskette zur effizienten Produktion von CFK-Strukturen in textiler Preform- / RTM-Technik".

A continuous process chain for robotic stitching of preforms from CAD (Computer Aided Design) planning to the final output was developed to realise a simpler production of prototypes. Robust and light weight components are manufactured by infiltrating the carbon fibres with resin.

Teach-in programming to specify different points on the trajectory for a flexible production is no longer state-of-the-art. Sensors are used to interact with the robot control unit for a fully automatic process [Sim et al., 2002]. Using such a visual approach, a higher reproducibility and precision of the seams is expected [Richtsfeld et al., 2007a, Richtsfeld et al., 2007c, Biegelbauer et al., 2007].

The main part of this work was to find a reliable and robust method to detect the different edge types between two overlapped carbon fibre layers, because this data is the main perceptive input for the robot control unit.

3.1.1 CFRP Composites Production Process

Carbon fibres are available in many different forms and there are many types of carbon fibre mats, e.g. unidirectional mats, where all fibres are oriented in the same direction or woven and plaited mats, where the bundles of fibres can be layered and oriented in the opposite direction. Fig. 3.1 shows an example of such carbon fibre mat. The white fibres hold and stabilise the black and reflective fibre texture. Different impregnation techniques can be used to infiltrate the preform with the resin matrix. The most popular techniques are the Resin Transfer Moulding (RTM) and the Vacuum Assisted Process technique (VAP).

The Resin Transfer Moulding (RTM) technique is a frequently used technique in the group of Closed Moulding or Resin Injection techniques [Murphy, 1998]. RTM describes a system in which the mat is placed in a mould and then the mould is closed by a counter mould. After this step the liquid resin is injected through an intake and another aperture

[1]European Aeronautic Defence and Space Company, http://www.eads.com

[2]This project was founded by the German Federal Ministry of Education and Research (Project No.: 01RI05089 - 01RI05096) and by the Austrian Science Foundation (Project No.: 810568/1553 SCK/SAI).

Figure 3.1: Example of carbon fibre mats.

is used to exhaust air from the closed mould. The composite is cured inside the closed mould and the mould can be heated to shorten the cure time. During the injection step, the resin is either pumped into the mould or drawn by a vacuum. After the cure process the finished cured element can be removed.

A special form of Open Moulding techniques is the Vacuum Assisted Process (VAP) [Li et al., 2004] method. The mat is placed onto a mould and a special membrane is mounted over the element. Next the assembly is sealed and a connected vacuum vent draws the resin into the carbon fibre mats.

Other techniques to apply the matrix to the carbon fibres are to spray the resin on the mats inside a mould or to apply the resin by hand using a hand roller. These and several other techniques can be found in [Murphy, 1998].

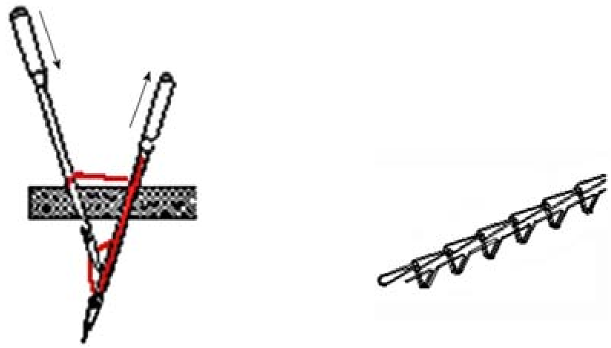
However, the gluing of these carbon fibre mats stresses the parts in the final hardening step and thus a new method is needed to improve the preforming process. Sewing instead of gluing increases the final properties and achieves better solidity and durability.

Over the last few years several different automated sewing machines and robots have been developed by garment industries [Krockenberger and Nollek, 1991, Gershon, 1990], whereas a robotic sewing machine handling carbon fibres has only just been proposed [Filsinger et al., 2003]. There is a difference between one side stitching with two needles [Witting, 2001], tufting [Sickinger and Hermann, 2001], and blind stitching technologies [Dewing et al., 1999].

In 2001, Witting presented a one side stitching technology with two needles. The thread is sewn under the material as illustrated in Fig. 3.2.

In 2001, Sickinger described another one side stitching method using tufting technology. Fig. 3.3 shows that the thread is stuck into the material by a needle and stays there due to friction forces and Fig. 3.5(a) shows a tufting head made by KSL[3].

[3]KSL Keilmann Sondermaschinenbau GmbH, http://www.ksl-lorsch.de



(a) (b)

Figure 3.2: One side stitching principle with two needles: **(a)** *one side stitching needles;* **(b)** *stitch pattern [Witting, 2001].*

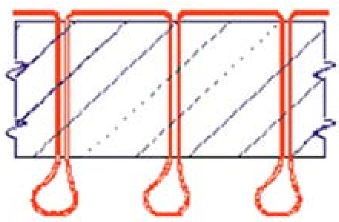


Figure 3.3: The tufting principle [Sickinger and Hermann, 2001].

Single side stitching offers many opportunities for composites. First, one has to distinguish between two different kinds of stitching methods: the structural method, which improves the mechanical properties of composites, and the assembly method, which involves stitching in order to enhance handling of the so called preforms. The advantage of this technology is the accessibility to the preform. In 1999, Dewing described an industrial robot capable of blind stitching, a technology where the thread is moved into the material by a curved needle [Dewing et al., 1999]. The stitching head mounted on the robot is patented as "Blind stitching apparatus and composite material manufacturing methods". Fig. 3.4 shows the principle of blind stitching and the stitch pattern and Fig. 3.5(b) shows a blind stitching head.

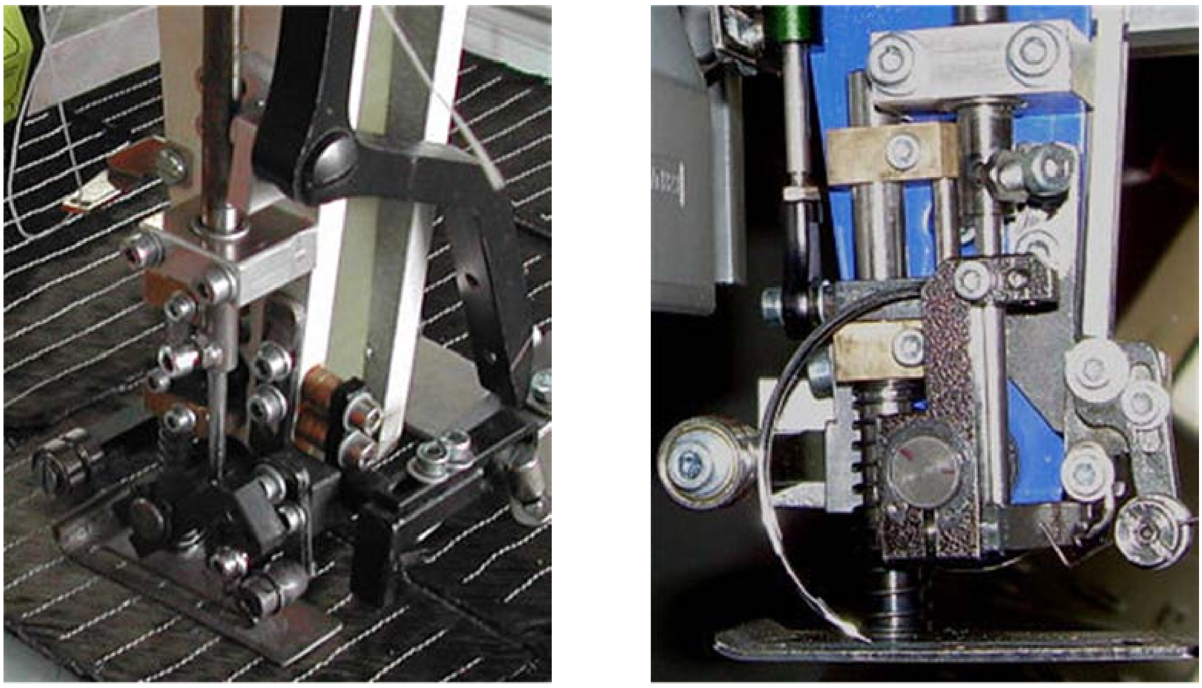
First industrial applications such as the Airbus A380 rear pressure bulkhead as illustrated in Fig. 3.6 have been realised by SAERTEX[4] and KSL have shown the potential of blind stitching technology (see Fig. 3.7). Note that the stitching head is mounted on a portal and the sewing process is performed in 2D [Filsinger et al., 2003].

However, for 3D applications, the need for higher flexibility is obvious. One possible way to achieve this goal is to use an industrial robot with a stitching head adopted for robotic applications and a laser stripe sensor mounted on the stitching head. In this new system, faults in the seam track due to imprecise mat cutting or application to the model should be minimised.

[4]SAERTEX GmbH and Co. KG, http://www.saertex.com

(a) (b)

Figure 3.4: The blind stitching principle: **(a)** *blind stitching needle;* **(b)** *stitch pattern [Steinhilber, 2006].*



(a) (b)

Figure 3.5: Sewing head, equipped with a: **(a)** *tufting system;* **(b)** *blind stitching system.*

3.1.2 System Overview

The goal of REDUX is to realise a continuous process chain for robot stitching preforms starting with CAD design and continuing with the application of carbon fibre mats until the end product is achieved. After the design of the CFRP element, a stitching track will be included into the CAD model. Then the fibre mats are cut and inserted into the model or draped onto the negative model and a stitching robot will sew the mats as specified in the CAD model. The application of the sensor guided stitching process makes it possible for the robot to handle minor inaccuracies autonomously. This becomes essential when the carbon fibre mats are not precisely draped onto the model and the fibre mat edge deviates significantly from the CAD path planning. These stitching programs are created

Figure 3.6: Airbus A380 rear pressure bulkhead fabricated with CFRP composites.

Figure 3.7: Portal sewing machine.

by an offline programming tool and are uploaded to the robot control unit. To achieve edge tracking, a laser stripe sensor is mounted on the stitching head to facilitate an autonomous path followed by the robot as illustrated in Fig. 3.8. Sensor data processing detects the seam and sends the actual edge position to the robot control unit, which transfers the deviation of the desired path to the robot and with this information the robot corrects the position of the stitching head online.

Fig. 3.8 shows a blind stitching head realised by KSL, which is mounted on a KUKA[5] 125/2 robot. This 6 axis articulated robot can handle loads up to $125kg$ at an arm length of $1000mm$ and has a repeatability accuracy of $\pm 0.2mm$. The maximum speed of the robot is $2m/s$. Fig. 3.9 gives an overview of the control loop for the edge detection process.

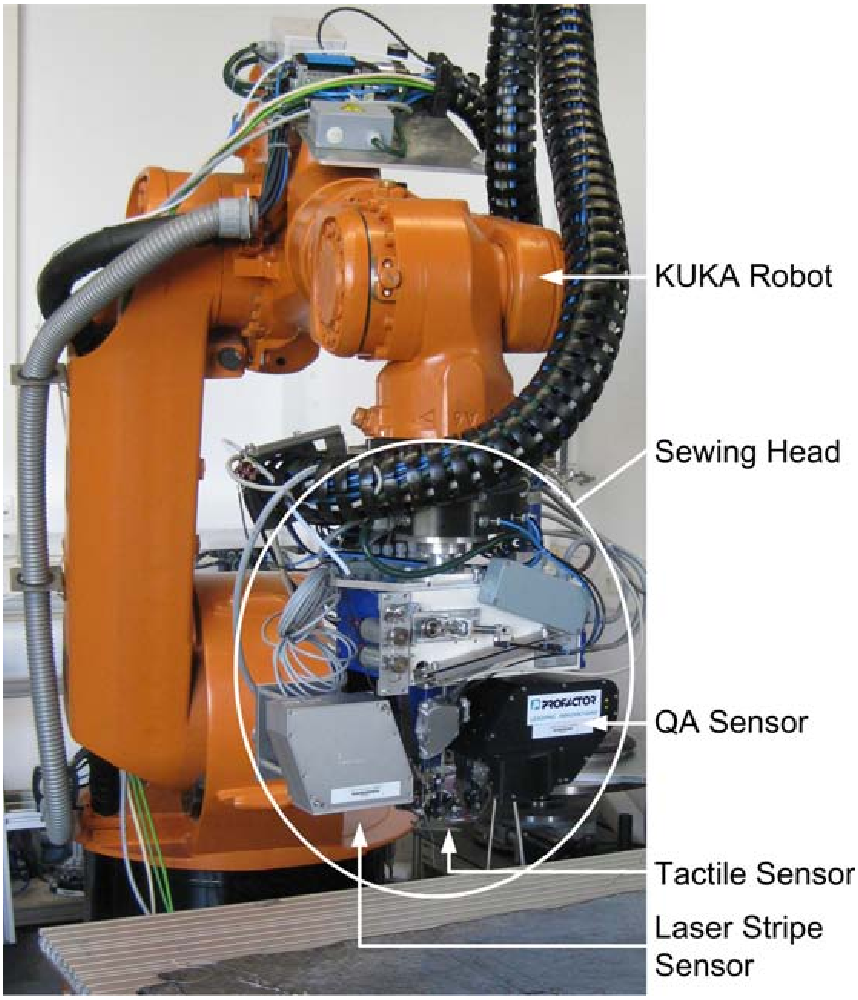


Figure 3.8: Industrial stitching robot with blind sewing head. The sewing head is equipped with three different sensors to correct the sewing path and to determine the seam quality directly after the sewing process.

To control the stitching process depending on the information of the sensors, a control system merges the information from all sensors and the desired stitching track to an actual movement track sent to the robot as illustrated in Fig. 3.10.

Fig. 3.8 shows three sensors mounted on the sewing head. Two sensors determine the position and orientation of the carbon fibre mats to be sewed and the QA (quality

[5]KUKA Robotics, http://www.kuka-robotics.com/usa/en

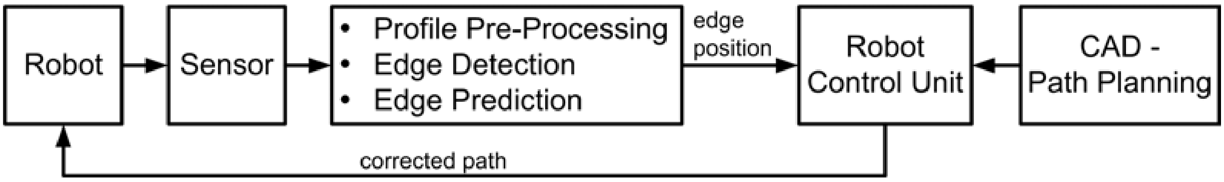


Figure 3.9: Robot guiding control loop.

assurance) sensor determines the seam quality after sewing. The determination of position and orientation is carried out by a tactile sensor and a laser stripe sensor. The tactile sensor gives information about the vertical distance between the sewing head and the surface. The objective of the laser stripe sensor is to give information about the horizontal position of the desired sewing track.

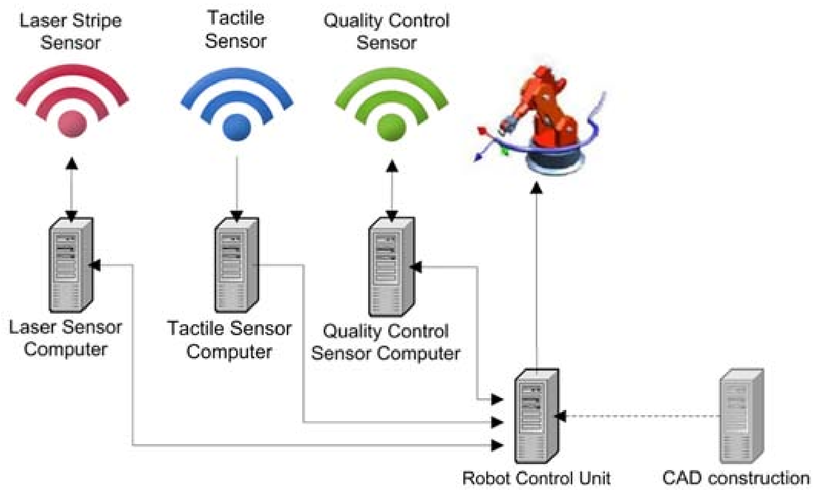


Figure 3.10: System components.

Then the real edge position is transmitted to the robot control unit. The robot control unit compares the desired distance between the seam and the edge, determined by the CAD data and the transmitted distance. The control unit includes this information in the desired sewing track and corrects the coordinates of the seam track to the current edge position as illustrated in Fig. 3.11. The correction of the sewing track is performed continuously to realise the permanent correct sewing path.

Robust edge detection and tracking for different types of edges in order to control the robot motion is needed. The difficulties to handle are the black and reflective carbon fibres, an edge height less than $0.5mm$, and outliers caused by spiky filaments. Fig. 3.12 shows typical profile data. An additional requirement is to detect the edges at a minimum rate of 30Hz to enable a reasonably fast seaming.

Summarising to the best knowledge of the author, the system described in this chapter is the first one using a laser range sensor for fully automated carbon fibre sewing. It is standard for the task of robotic seam tracking to use the laser range

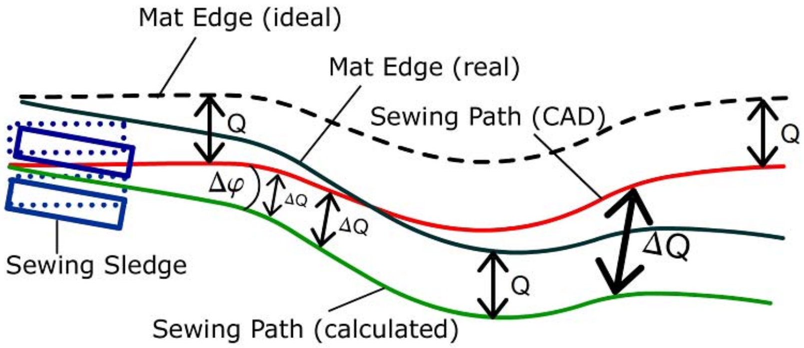


Figure 3.11: Correction of the sewing path with sensor information [Schöffmann, 2008].

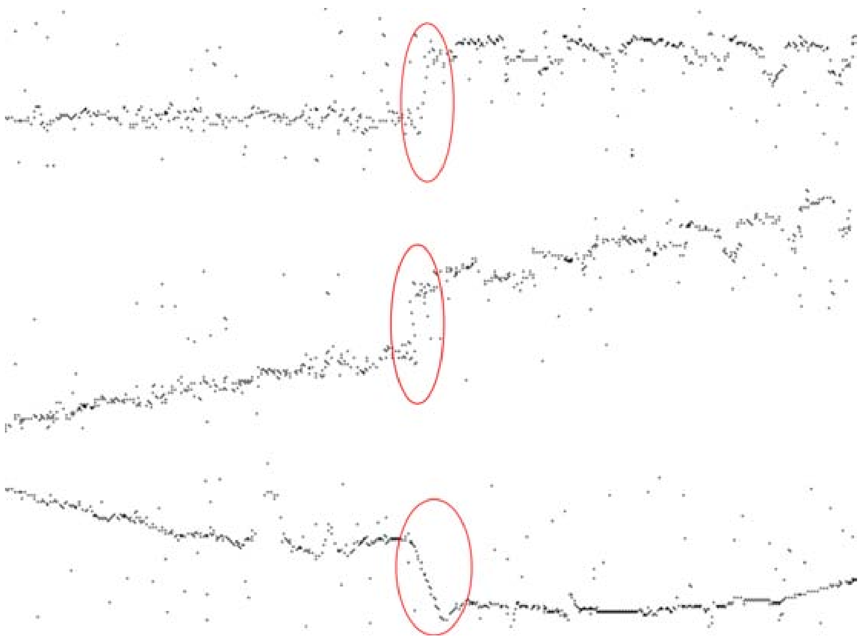


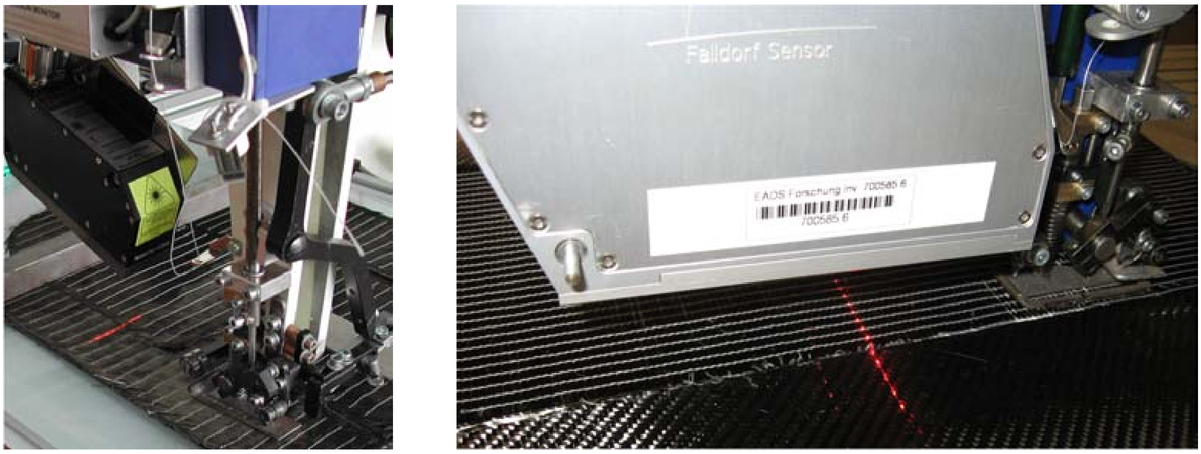
Figure 3.12: Close-up view of different laser stripe profiles including the edge. The edge to be detected is indicated with a red ellipse.

sensors mainly applied in welding applications [Pritschow et al., 2002]. With welding applications, the seams are geometrically well defined and comparatively easy to detect in the range data because there is no reflection. Nevertheless, first attempts have been made following textile seams on a planar workplace using a CCD camera [Gershon and Porat, 1988]. Meanwhile, vision systems in textile robotics are mainly used to check the quality of the seam after the sewing process because the sewing thread is clearly visible [Dorrity, 1995, Bahlmann and Heidemann, 1999].

The sensors used in this special application have been developed and produced by the company Falldorf Sensor[6] GmbH. In the beginning, the standard "Base Sensor", which

[6]Falldorf Sensor GmbH, http://www.falldorfsensor.de

produces a 2D profile of the carbon fibre surface was used as illustrated in Fig. 3.13(a). Later on, a newly developed range sensor the "3 Lines Sensor", which produces two 2D profiles of the carbon fibre surface was preferred. The first scan line is fixed and the second scan line is alternately switched between the two positions. The setup of the new laser stripe sensor and the stitching head is presented in Fig. 3.13(b).



(a) (b)

Figure 3.13: Detailed overview of the laser stripe sensor on the left and the stitching head on the right: **(a)** *Standard "Base Sensor" with one laser scan line.* **(b)** *Newly developed "3 Lines Sensor" with the* 1*st* *fixed laser scan line and the* 2*nd* *laser scan line is alternately switched between two positions.*

The next section presents the proposed method in detail and starts with the flow chart of the framework in Fig. 3.14 of the overall approach. It shows the main processing steps from the required range image of the sensor to the actual edge position sent to the robot control unit per XML data packages.

The method starts with a profile pre-processing step followed by the edge detection, and edge prediction step. The normalisation of the laser stripe profiles is necessary to handle the different profiles and to guarantee that every method is able to work on this input data. For a robust edge detection and voting an edge is classified as detected correctly if at least two out of three methods find the edge within a certain tolerance. To increase the detection rate, several profiles are considered depending on the robot velocity. The robot control unit needs the actual edge position to correct the path trajectory every $100ms$. The sampling rate of the laser range sensor varies from 30Hz to 100Hz, depending on the reflectivity properties of the carbon fibre mats, and adjusts the exposure time as necessary. Therefore, three to ten profiles are available to determine the correct edge position. Assuming linear fibre mat edges inbetween two sampling instances of $100ms$, all detected edge positions - including the last transmitted edge position - are used for edge prediction. The final edge point is selected by using a RANSAC-based line fit [Fischler and Bolles, 1981]. Additionally, this increases robustness of edge detection.

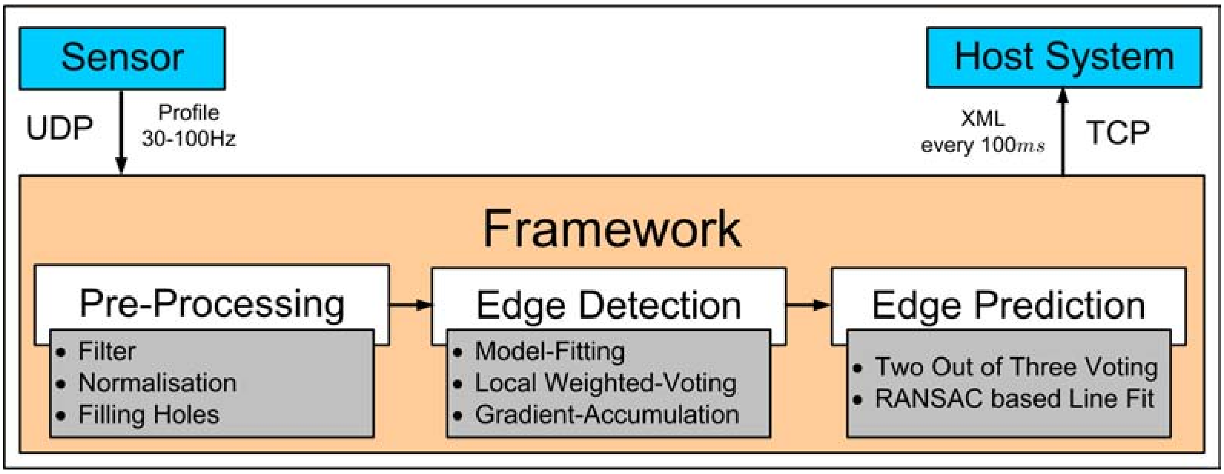


Figure 3.14: Flow chart of the edge detection approach. The interrelation of the main processing steps of the control loop are illustrated.

3.2 Profile Pre-Processing

The laser stripe sensor delivers unfiltered point clouds that exhibit a non-linear relationship between the pixel (px) position in the camera frame and the real world distance as illustrated in Fig. 3.17(a). The height profile is transformed from camera coordinates to real world coordinates, which is solved by a bilinear transformation using a look-up table and a defined calibration object. The calibration object is scanned from different distances and the profiles are evaluated. The resulting look-up table describes a curve progression between the pixel (px) distance and the real world distance in mm. This curve is used to determine a function 4^{th} order for the conversion. In practice this method allows a flexible possibility of calibrating different sensors on the same system.

The task of profile pre-processing is to filter the data from the sensor, normalise curved and slanted profiles as illustrated in Fig. 3.15, and fill the holes caused by the filtering procedure.

To eliminate the worst outliers and noises, the laser range data of one profile is filtered sequentially by a geometrical filter. Geometric filtering is the most effective filter technique, whereas the histogram filter tested achieved comparatively bad results. The geometric filter analyses each data point and counts how many more data points in the neighbourhood occur within a certain rectangular distance window.

The second task of profile pre-processing is to handle the different types of profiles, shown in Fig. 3.15. In practice, as demonstrated by manufacturing examples, data alignment is required to obtain a flat profile. This step is necessary to apply global edge detection methods. Global methods for edge detection work with the complete set of raw data points of one laser stripe profile, whereas local methods only take a small number of pixels from the profile. Thus the algorithms for global methods cannot find the edge of convex/concave or slanted edges [Lee and Park, 1990].

To normalise the different profiles, a quadratic least-squares fit is used, as in Fig. 3.16.

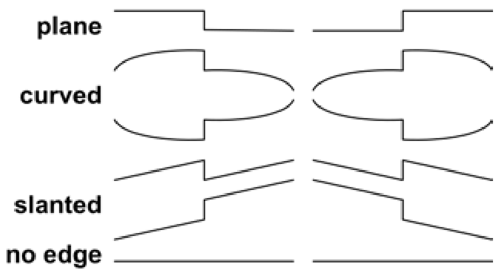


Figure 3.15: Types of different edges and no edge types.

This least-square distance function fits the curve to the profile data by finding the minimal distance to all data points. The curve fitting preserves the edge and a normalised data profile is generated.

The drawback is that the original information about the height related to the distance of the sensor to the fibre mat is lost. To avoid this, the original unchanged data is saved in the system together with the new normalised data profile.

As a direct result of normalising the data, the holes caused from the previous filtering step can be filled by connecting boundary points with horizontal lines without causing artificial edges.

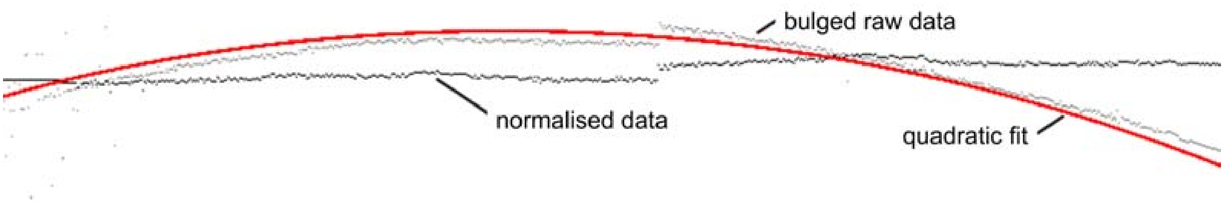


Figure 3.16: Normalisation of the laser stripe profile in the pre-processing step.

3.3 Edge Detection for Automated Robot Stitching

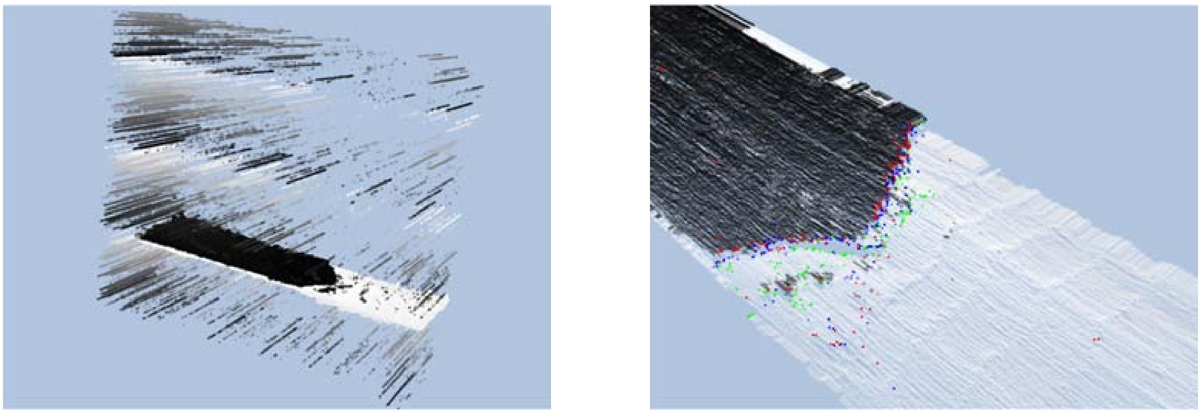
This section describes how to detect the edge between overlapping carbon fibre mats in normalised profiles. Of utmost importance is to obtain reliable detections over the range of different profiles in real-time. The idea to increase the detection rate is to use voting over three complementary individual edge detection methods, an approach taken when building dependable systems for aviation. Specifically, an edge is classified as detected correctly if at least two out of three methods find the edge within a certain tolerance of $1mm$. The complimentary methods implemented are:

- Model-Fitting: global least-squares fitting of an edge model. This is adapted from [Suetens et al., 1992] and works on the complete profile. Instead a profile pre-processing step is essential.

- Local Weighted-Voting: the neighbouring pixels of pre-selected edge candidates give their vote, weighted with the edge height. This local method works on a small subset of pixels and also achieves good results without a normalisation step.

- Gradient-Accumulation: sums up the gradient magnitude calculated with an increasing kernel size. It is inspired by the Roberts operator [Roberts, 1965] and mixes the global with the local approach.

Fig. 3.17(b) shows the noise of 256 unfiltered scan lines. The edge detection results of all three individual methods are illustrated in Fig. 3.17(b).



(a) (b)

Figure 3.17: **(a)** *Unfiltered* 256 *scan lines: a noise is clearly visible. The particularly dark and bright coloured area is the correct step edge of two overlapped carbon fibre mats.* **(b)** *Edge detection results of all three implemented individual methods based on filtered scan lines. The detected edges of the model-fitting method are green, of the local weighted-voting method blue, and of the gradient-accumulation method coloured red.*

The following subsections present the proposed edge detection methods in detail. The methods are based on laser stripe profiles with a length of $1024px$.

3.3.1 Model-Fitting

In model-based recognition the target object shape is represented by its geometry: e.g. a template represents an object as a rigid curve or an image. The advantage of model-fitting is that the model encodes the object shape, thus allowing predictions of image data. Coincidental features have less chance of being falsely recognised. To find the optimal template location, a metric or a similarity measure is necessary that reflects how well the image data matches the template [Suetens et al., 1992].

For an explicit description, the raw data points of a profile scan are defined as $f(i)$ and the model function of a positive step edge is used. Comparing the right column of

Fig. 3.15 you will see the red-marked step edge, which illustrates such a model in Fig. 3.18. The height of the minimum and maximum pixel is given by:

$$h(i,j) = \begin{cases} \min(f(i)), & i < j \\ \max(f(i)), & i \geq j \end{cases}, \tag{3.1}$$

where j is the position of the edge. The least-squares sum of a desired edge position is:

$$S(j) = \sum_{i=1}^{m} d(i,j)^2, \tag{3.2}$$

where m is the length of the profile line and $d(i,j)$ the distance function:

$$d(i,j) = f(i) - h(i,j). \tag{3.3}$$

Finding then the best fit, all pixel positions in the profile scan are evaluated and the one with the smallest sum wins:

$$\arg\min_{1 \leq j \leq m} S(j). \tag{3.4}$$

Fig. 3.18 illustrates this method with an example scan line of $1024px$. Note, due to computational efficiency, a prediction of possible edge candidates is previously passed.

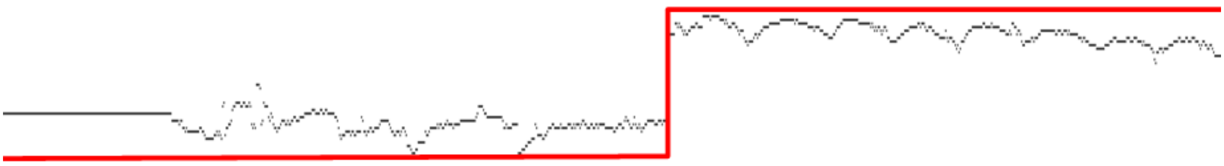


Figure 3.18: Illustration of the model-fitting method.

3.3.2 Local Weighted-Voting

The proposed approach works as a local method using weighted-voting of every pixel in a neighbourhood of the considered edge using a kernel size of usually $n = 75px$. Initially, a prediction of potential edge candidate points is made by calculating the absolute difference of neighbouring pixels. The first 30 most significant edge candidate points are used for further processing. Considering a positive step edge, the edge height – respectively the gradient – of each edge candidate is calculated with:

$$w(j) = |\max(f(j+k)) - \min(f(j-k))|, \ 1 \leq k \leq 3. \tag{3.5}$$

To further improve robustness, the following constraint was exploit. As already mentioned, spiky carbon filaments cause short but large step edges. To handle these unwanted edges the local average is calculated with:

$$a(j) = \frac{1}{2n+1} \sum_{i=j-n}^{j+n} f(i) \tag{3.6}$$

and all points which are above the average on the left side of the positive step edge or below the average on the right side of the positive step edge obtain no weight for the voting process. Hence, the voting function is written as:

$$v(i,j) = \begin{cases} j-i, & (i<j) \wedge (f(i) < a(j)) \\ i-j, & (i>j) \wedge (f(i) > a(j)) \\ 0 \;, & (i>j) \wedge (f(i) < a(j)) \\ 0 \;, & (i<j) \wedge (f(i) > a(j)) \end{cases} \tag{3.7}$$

with a voting factor $i-n$, respectively $i+n$, rating pixels stronger that are far away from the edge position. Thus pixels with a larger distance to the edge position get a higher weight, which is reasonable for strong frayed edges. The weighted voting sum is finally given by:

$$S(j) = w(j) \sum_{i=-n}^{n} |v(i,j)| \tag{3.8}$$

and the edge with the maximum sum 'considering only the pre-selected 30 edge candidates' wins:

$$\arg \max_{1 \le j \le 30} S(j). \tag{3.9}$$

Fig. 3.19 illustrates this method with an example scan line with $1024px$. Note that sometimes several fibres can be disconnected from the edge and then two new additional step edges are build. With this simple local method, the hit probability to detect the correct edge by abnormal normalised scan lines is higher compared to the model-fitting and the gradient-accumulation method.

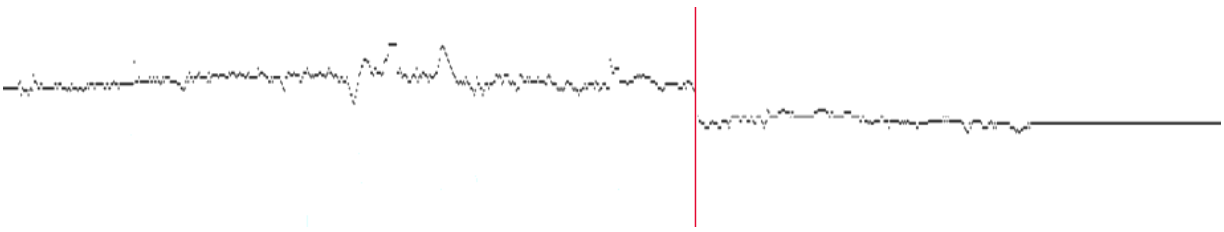


Figure 3.19: Edge detection result of the local weighted-voting method.

3.3.3 Gradient-Accumulation

Generally, the position of a step edge is characterised through the maximum peak of the first derivative. Inspired by the Roberts edge detector [Roberts, 1965], the scan line can be expressed as the continuous height function $f(x)$. Fig. 3.20 illustrates this method with an example scan line with $1024px$.

Hence, the magnitude of the gradient is given by:

$$g(x) = \frac{\partial f(x)}{\partial x}. \tag{3.10}$$

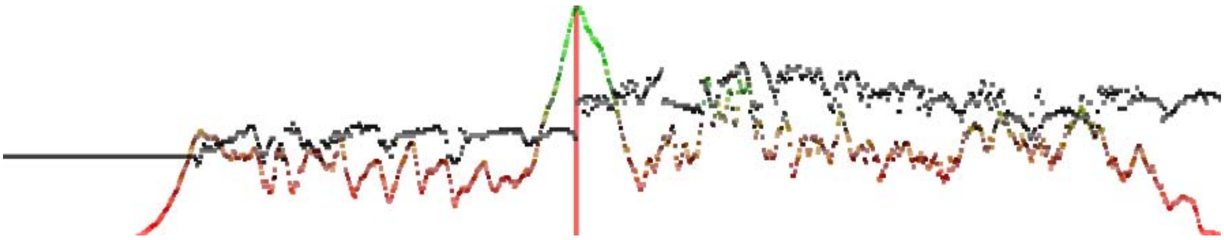


Figure 3.20: Illustration of the gradient-accumulation method.

Because of the pixel quantisation of the profile scan, the gradient has to be calculated on discrete values and can therefore be approximated with:

$$g(i) = |f(i+n) - f(i-n)|, \tag{3.11}$$

where n is the kernel size for the approximation, usually $n = 1$. Applying the gradient filter of Eq. (3.11) on the noisy profile raw data $f(i)$ results in a gradient function $g(i)$. Due to the small edge height compared to the noise, no significant gradient peak indicating the edge position can be detected. With the classical approach, outliers and artefacts cause larger peaks than the edge itself and detection will fail. Increasing the kernel size n does not change this fact.

Considering the fact that the algorithm is searching for a global step edge corrupted with noise, a multi-scale gradient calculation will suppress the noise of the local outliers and artefacts in the range data. Summing up the gradient functions for each pixel position with different kernel sizes, the correct edge is determined as illustrated in Fig. 3.21. The

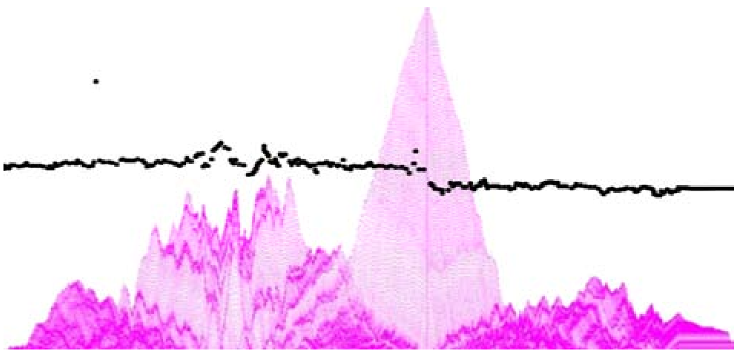


Figure 3.21: Result of the gradient-accumulation method after 10 iteration steps, the detected scan line is highlighted.

modified function $G(i)$ calculates the accumulated gradients using increasing kernel sizes:

$$G(i) = \sum_{k=1}^{n} |f(i+k) - f(i-k)|, \tag{3.12}$$

where n is the kernel size. Using a kernel size of 10% of every scan line has been found to be optimal with respect to the calculation time and performance, based on the testing of several thousand profiles. Finally, the maximum of $G(i)$ localises the position of the edge. A more detailed description of this method can be found in the master thesis of Walter Wohlkinger [Wohlkinger, 2007].

3.4 Edge Prediction

The three edge detection methods described above are executed independently on each profile. The idea is to use two out of three voting to obtain a dependable result. The edge of one profile is considered as successfully detected if at least two of three methods locate the edge within a certain tolerance of $1mm$. The evaluation in Section 3.6 and Section 3.7 will show that, due to the noisy raw data, the edge detection based on a single profile is still not satisfactory. To further increase the detection rate, several profiles are considered depending on the robot velocity.

The robot control unit needs the actual edge position for correcting the trajectory every $100ms$. The sampling rate of the laser range sensor varies from 30Hz to 100Hz depending on the reflectivity properties of the carbon fibre mats. Therefore, three to ten profiles are available to determine the correct edge position as illustrated in Fig. 3.22. Assuming linear fibre mat edges inbetween two sampling instances ($100ms$), all detected edge positions – including the last transmitted edge position – are used to smooth the final edge position.

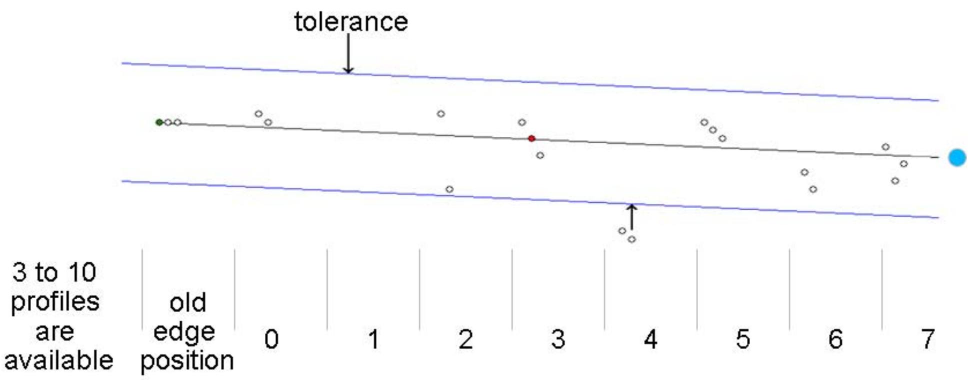


Figure 3.22: Edge prediction to find the next edge position for the robot control unit.

The final edge position is predicted by using a RANSAC-based [Fischler and Bolles, 1981] line fit. This further increases the robustness of edge detection as presented in Section 3.6.

3.5 Detection Algorithm Improvements

Two improvements are realised for the edge prediction step. The first one is a more stable way for the two out of three voting, because each of the three methods shows higher values at the edge position and this position produces the highest peak of the accumulated weights. By then the closest two edges inside a defined threshold have been used to calculate the final edge position. The second improvement is a more robust way to predict potential edge positions.

During the tests directly on the robot with the robot control unit one failure occurs. The adjustment of the sensor head calculated by the control unit was too fast. Thus, the scan line of the sensor mounted in front of the sewing head looks in the wrong direction, causing the edge prediction with the line fit to find the false edge position. To correct this incorrect edge prediction, the line fitting method is cancelled and an edge prediction based on a probability analysis is implemented. Using this method the edge detection rate improves.

3.5.1 Sum of Weight Function

As already described above, each algorithm calculates the most probable edge position and the resulting edge positions are then combined to one single result. If incorrect results are produced by the algorithm, these results have their origin in the analysed scan lines. In which case, the feature of an interfering fibre, noise or outliers which could not be filtered would be more significant than the actual edge.

Each of the edge detection algorithms focuses on different data features to calculate the edge position. If the algorithms are not able to find the correct position due to multiple influences, the highest significance of a potential edge position with all three methods is calculated. The calculated weight functions of all three algorithms are summed up before the position of the highest significance is searched and the weight values are accumulated. Each of the three algorithms shows higher values at the edge position, producing the highest peak of the accumulated weights. Meanwhile, the peaks of interferences do not obtain the weights of each algorithm and consequently have a lower peak as illustrated in Fig. 3.23. All three algorithms determine the edge position probability values in different magnitudes, so the values are normalised before summing the weights. Otherwise the impact of the different algorithms on the result would be unequal. The position determination of the algorithms is replaced. The minimum in the model-fit algorithm is exchanged by:

$$w_{mf(j)} = 1 - \frac{S(j)}{max(S(j))}. \tag{3.13}$$

Here the values of the weighting function are normalised by division through the maximum value of the function and subtracted from one, since the smallest error sum characterises the best model-fit.

The calculation of the highest voting sum is replaced by the normalised function of voting point accumulations. The normalisation is performed through division by the maximum voting sum. The new weighting function for the local weighted-voting algorithm can be written as:

$$w_{lwv(j)} = \frac{S(j)}{max(S(j))}. \tag{3.14}$$

The result of the gradient accumulation algorithm is replaced in the same way. The

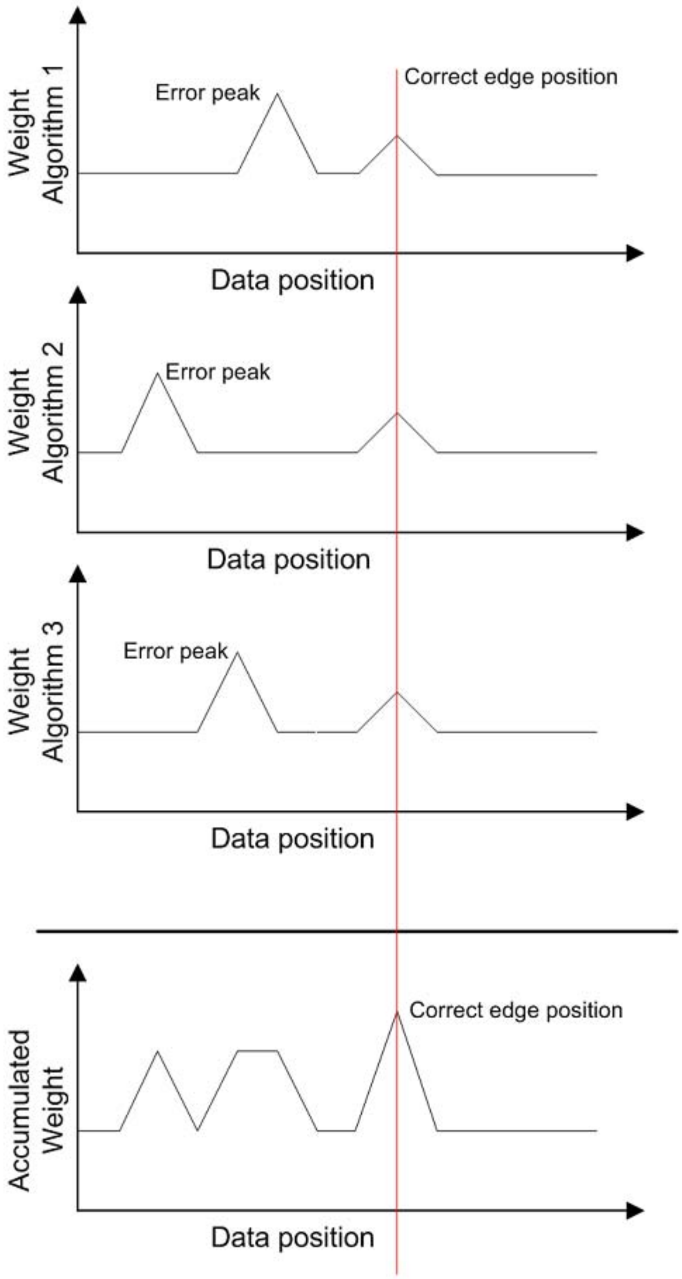


Figure 3.23: Accumulation of three algorithm results with several incorrect maxima [Schöffmann, 2008].

normalised weighting function of the gradient accumulation algorithm is:

$$w_{ga(j)} = \frac{S(j)}{max(S(j))}. \tag{3.15}$$

Now, the final result combining the single findings of the three different algorithms can be performed by summing up the individual normalised weighting functions:

$$W_{(i)} = w_{mf(i)} + w_{lwv(i)} + w_{ga(i)} \tag{3.16}$$

and the edge position with the highest summed weight of all m (between 3 and 10 depending on the sampling rate of the laser) detected edge positions is calculated by:

$$\arg\max_{1 \le i \le m} W_{(i)}. \tag{3.17}$$

As described in [Schöffmann, 2008], this procedure is more robust against false positive edge detections, since less dominant edge positions are strengthened against dominant interferences.

3.5.2 Weighting Influence by Historic Edge Positions

Above a method to smooth the edge detection results with a line fit was described. The positions of every edge step are assumed not to change rapidly and so a line fit is calculated for the past three to ten edge findings. The calculation of the line fit omits far off outliers by using the RANSAC [Fischler and Bolles, 1981] algorithm. In Fig. 3.24 the successive

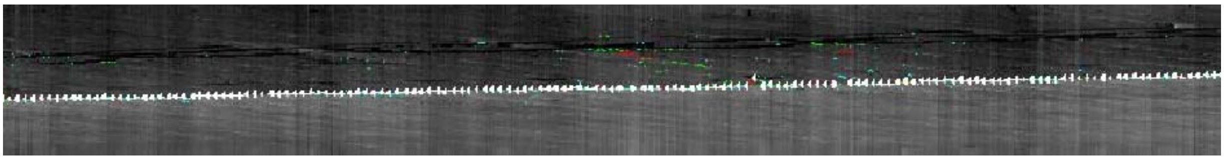


Figure 3.24: Gray value scan data image with stable edge positions.

scan data of a carbon fibre mat is illustrated as gray value image. The edge between two mats can easily be seen as transition from dark gray to light gray. The calculated final edge positions are marked with white dots. It can be noticed that the edge steps are along a straight line. This data set results from a scan recording where the sensor head was moved over the edge of overlapped fibre mats along a straight line. However, if the sensor is attached to the sewing head and the edge detection is performed, the results of the edge detection are used to correct the seam path. The adjustment of the robot head by the robot control unit will be realised by the sensor.

This situation can be seen in Fig. 3.25. A metal ruler was attached to a carbon fibre mat to simulate an edge. An edge modification in the shape of a half circle was clipped to this ruler, while the CAD seam track data showed a straight path. The darker area in the image represents the ruler, the lower part is the gray scale image of the carbon fibre mat. The area between the red marker lines shows the half circle of the edge modification recorded by the laser stripe sensor. The information about the divergence to the CAD data was processed by the robot control unit and the robot head was moved to the right side. In relation to the sensor, the ruler moved to the left side, which can be seen in the area marked with blue lines. The area of the adjustment is slightly larger because the

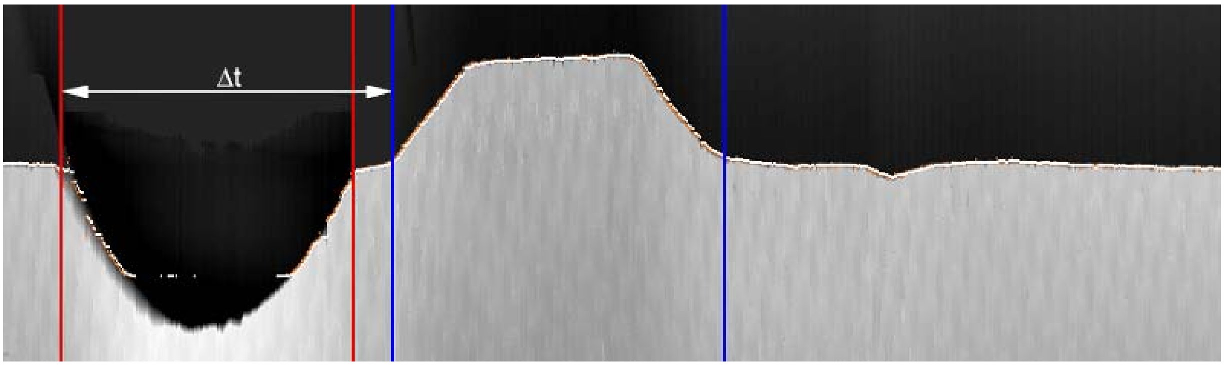


Figure 3.25: Gray value scan data image with mirrored correction movement.

correction is smoothed. The difference Δt between the first occurrence of the modification in the data and the beginning of the adjustment represents the distance between the scan line and the centre of the sewing sledge in relation to the movement speed. Here the scan line was approximately $80mm$ in front of the seam centre. The scan data shows a shifting in the edge position at a moment where the edge in reality always is at the same position. In this experiment, the robot head was only laterally shifted and not rotated. In the final setting, the robot head is supposed to rotate to achieve the correct adjustment. This rotation in the centre has even more impact on the sensor data, since if the distance between centre and laser line is $80mm$, a rotation of only $10°$ results in a shifting of the edge of $14mm$. Assuming if there are many minor adjustments during the stitching process, it can be concluded that the edge positions in real life scan data are not situated on a straight line. Therefore the basis for edge prediction by a line fit is not given any longer.

An alternative solution was searched to utilise the information about the previously found edge positions. The condition that the edge position in the real world system does not change abruptly is still valid, although the representation in the sensor data is more volatile than previously assumed. In the following, an algorithm for including the information of edge positions into the prediction of a new position is presented.

If x and $x1$ are two points on a line, the Gaussian function:

$$f_x = e^{-\frac{(x-x1)^2}{2\sigma^2}} \tag{3.18}$$

weights the distance of x to the point $x1$. A point x close to $x1$ gets a higher value, a point further away a lower value, depending on the parameter σ which defines the width of the Gaussian bell curve. Fig. 3.26 shows the Gaussian bell curve to weight the distance of x to $x1$. If $h(t)$ is a list of previously detected edge positions and n the length of the history list, $h(1)$ is the oldest detected edge and $h(n)$ the latest detected edge stored in the list. Thus:

$$g_x = \sum_{t=1}^{n} t e^{-\frac{(h(t)-x)^2}{2\sigma^2}} \tag{3.19}$$

gives the value of closeness to each of the stored edge positions and additionally adds

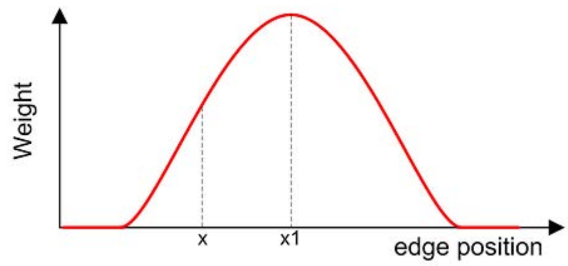


Figure 3.26: Gaussian bell curve weighting (the distance of x to x_1) [Schöffmann, 2008].

a linear weight for the currency of the historic edge position. The weight gained from previously found edge positions is added to the function of accumulated algorithm weights before searching the maximum. In order to regulate the maximal influence of historic edge positions on the weighting function, a factor is included into the weighting of historic edge positions. This factor limits the maximal value to one half of the current weight value. The maximum of the bell curve is 1. The resulting function of edge probability weights including the old position information $W_h(i)$ is:

$$W_h(i) = W(i)\left[1 + \frac{1}{n(n+1)} \sum_{t=1}^{n} j e^{-\frac{(h(t)-i)^2}{2\sigma^2}}\right] \tag{3.20}$$

and the highest summed weight of Eq. (3.17) is replaced by the search of the maximum in $W_h(i)$ by:

$$\arg \max_{1 \leq i \leq n} W_h(i). \tag{3.21}$$

3.5.3 Minimal Edge Step Size

The drawback of searching the maximum values in the function of probability weights is the fact that a maximum always exists. There are basically two reasons for the absence of an edge step: either a single mat was scanned without an overlapping edge or the edge was too small and respectively the scan data too noisy. In any case, if the data does not include an edge step, the edge detection should not find any edges. To be able to detect whether no edge exists in the data, a minimal edge step size is introduced. After searching the area with the highest edge position probability, the step size at the found position is tested to see whether it exceeds the threshold value. If the threshold is not reached, the algorithm reports that no valid edge step could be found in the data. The data in an area of $\pm 10px$ around the calculated edge position is smoothed by a median filter for the calculation of the step height. After smoothing the scan line, the local minima and maxima in this area are searched and the difference between these values is considered the edge step height. Currently the threshold is fixed to $0.3mm$, which corresponds with the smallest fibre mat type.

3.6 Detection Results Based on the "Base Sensor"

Experiments are carried out acquiring 2D laser scan lines using the REDUX prototype to evaluate the behaviour and the robustness of the edge detection method on real settings. During REDUX, two different laser range scanners were used: at first, the standard "Base Sensor", and in a later stage of development, the "3 Lines Sensor" as illustrated in Fig. 3.13. Both sensors have their own benefits and drawbacks, especially since the "3 Lines Sensor" is not completely developed and the results reflect this factor.

Firstly, the detection results of the "Base Sensor" with one scan line and different special cases are presented. In these tests, the edge prediction with the RANSAC-based line fit and the two out of three voting within a small pixel tolerance are implemented. Afterwards, the detection results based on the "3 Lines Sensor" are presented. Here the weighting influence by historic edge positions and the sum of weight function are implemented. At the end of this section, a comparison between both systems is given.

More than 50 different seam examples and about 30,000 profiles have been used to evaluate the above methods and improvements. All tests were performed on a AMD Athlon 64X2 Dual Core 4800+ processor with 2GB RAM. First the edge detection behaviour is shown on three examples of difficult cases, that is: a double edge, a slant edge, and a convex edge profile. Then the performance of the methods is generally evaluated concerning the computational effort and the detection rate.

3.6.1 Double Edges

Fig. 3.27 illustrates that whilst draping the fibre mats on the model, the fibre layers shift in some cases and the resulting edge looks like a double edge. It shows that the model-fitting method detects the correct edge (vertical red line), because the sum of the distances is minimal in this position. The local weighted-voting method detects the same edge, because the voting factors increase the probability to detect the correct edge. The gradient-accumulation method detects also the correct edge by looking after the maximum in the sum of the first derivatives, which can be seen in the course of the summed up gradient.

3.6.2 Slant Edges

Another problematic case is shown in Fig. 3.28, which plots a laser stripe profile with a slant edge after normalisation. The interesting point in this case is that every method detects the correct edge, but at different positions in the area of the slant edge. This naturally occurs because every method has its own edge detection criteria.

In the edge voting process, at least two edge detection results in one profile must be located within a tolerance of up to $1mm$ for the line fitting calculation. In contrast to the example of the double edge, where all three methods detect the same edge position, the model-fitting and the gradient-accumulation results agree.

Note that the slant edge in Fig. 3.28 extends over about $2mm$ and the seam deviation can be several centimeters. When using the edge voting method, only three times the robot

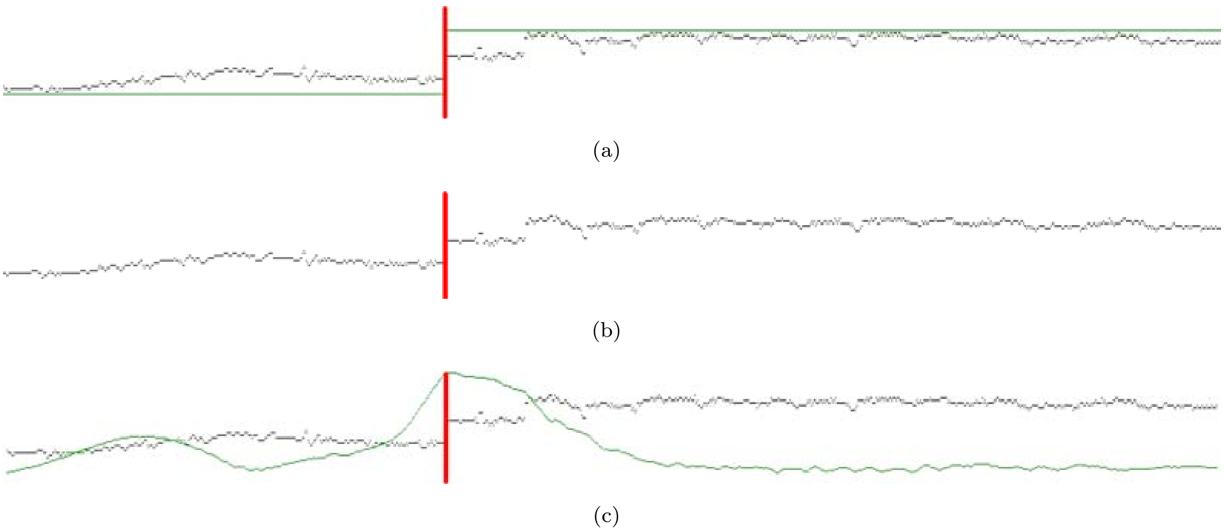


Figure 3.27: All three different methods applied on a double edge: **(a)** *Model-Fitting;* **(b)** *Local Weighted-Voting;* **(c)** *Gradient-Accumulation.*

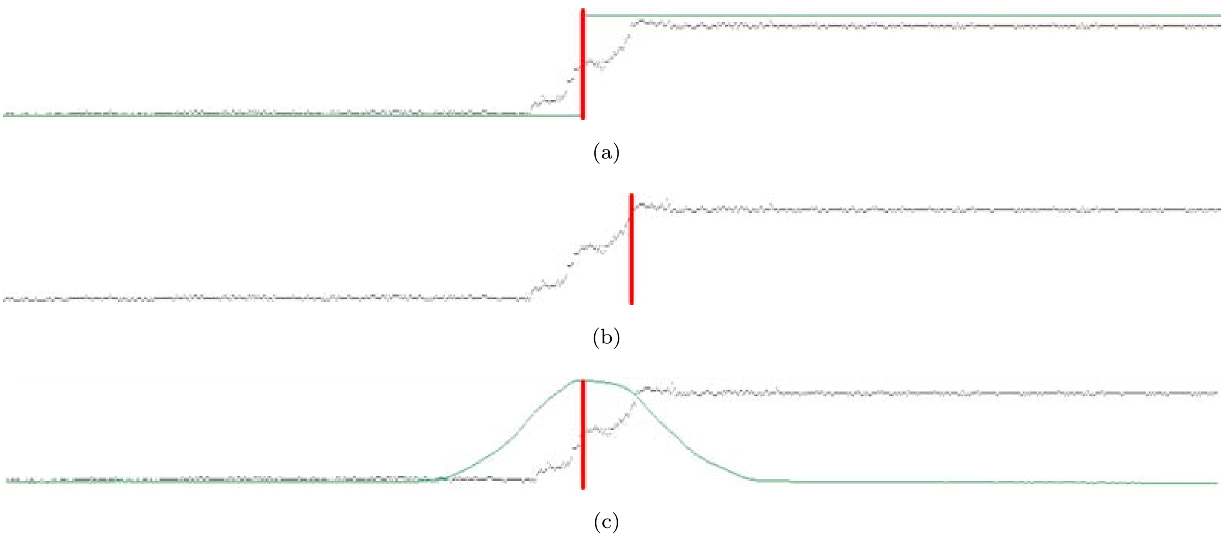


Figure 3.28: All three different methods applied on a slant edge: **(a)** *Model-Fitting;* **(b)** *Local Weighted-Voting;* **(c)** *Gradient-Accumulation.*

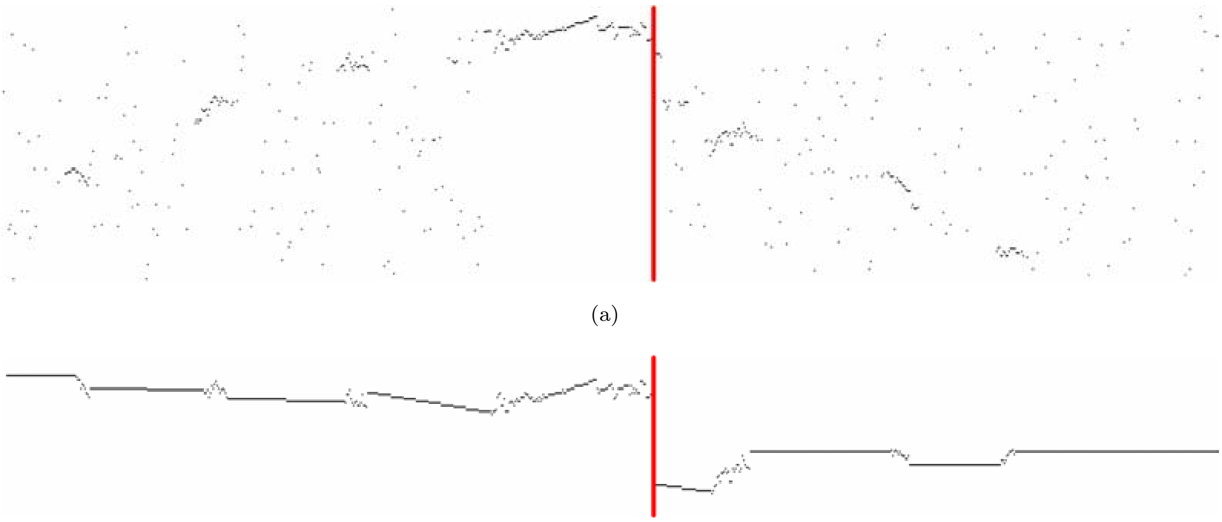
control unit could not be served with a valid edge position. The maximum deviation of the correctly detected edges was $1mm$.

3.6.3 Convex Edges

Sometimes the carbon fibre mats are draped on a cylinder, in this example with a radius of $30mm$, and thus the resulting edge looks like a convex edge as illustrated in Fig. 3.29.

Fig. 3.30 shows that the two out of three voting method detects the correct edge with a normalisation step of the profile.

Figure 3.29: Fibre mats draped on a cylinder with a radius of 30*mm.*



(a)

(b)

Figure 3.30: Laser scan line: (a) *original scan line;* (b) *normalised scan line. The detected edge is marked with a red vertical line.*

Note that Fig. 3.27, Fig. 3.28, and Fig. 3.30 show a single profile result, whereas the detected edges used for the edge voting are indicated with a thick vertical line.

3.6.4 Runtime and Detection Rate

To obtain the result in real-time, all methods including pre-processing must be sufficiently fast. For robotic stitching, real-time is dependent on the frequency of the sensor, which in turn depends on the sewing speed of the robot. All tests have been made under the same conditions: all tests have to be realised on the same fibre mats, respectively the same laser stripe profiles for a statistical analysis with an exposure time of $12ms$. Tab. 3.1 shows the results. When using all methods the maximum runtime was $4ms$ and confirms the suitability of the method to obtain real-time operation. Adding the exposure time of obtaining a profile, an effective scan rate of 62Hz was achieved for these experiments. Finally, the detection rate granted a certain tolerance of $1mm$ was evaluated for the

Methods	Min	Average	Max
Profile Pre-Processing	0.638ms	0.679ms	1.314ms
Model-Fitting	0.129ms	0.132ms	0.397ms
Local Weighted-Voting	0.208ms	0.249ms	0.502ms
Gradient-Accumulation	0.375ms	0.384ms	1.744ms
Sum including edge voting	1.445ms	1.445ms	3.957ms

Table 3.1: Runtime of every method.

three edge detection methods and the combined edge voting method. Evaluation is based on $30,000$ semi-automatic hand labelled profiles from different mats, as exemplified in Fig. 3.12. A positive detection is defined as a detection result within a tolerance of $1mm \widehat{=} 12px$ of the labelled edge. Tab. 3.2 and Fig. 3.31 show the results achieved under the same conditions as before.

Methods	$\pm 3px$	$\pm 10px$	$\pm 12px$	$\pm 18px$
Model-Fitting	85.7%	93.4%	94.3%	95.2%
Local Weighted-Voting	81.5%	84.4%	86.7%	91.2%
Gradient-Accumulation	65.3%	87.4%	94.3%	96.3%
Combined Result	80.4%	97.5%	99.3%	99.5%

Table 3.2: Detection rate of every method: "Base Sensor".

During the experiments it was noticed that the mat type and the edge position have a strong influence on the detection rate, but the evaluation is made over all possible mat types during the production process.

3.7 Detection Results Based on the "3 Lines Sensor"

The developed "3 Lines Sensor" pointed out clear disadvantages in comparison with the previously used "Base Sensor" with a single scan line. The only benefits are a robust mat end and obstacle detection. The sensor determines a possible pixel position (the brightest pixel point in the scan line) in each scan line of the CCD-chip for the 2^{nd} scan line. If no pixel position can be recognised in a line due to bad reflection or absorption characteristics of the material, the pixel with the maximum brightness is determined. The pixels with the maximum brightness can be adjusted anywhere in the scan line, usually from the previous

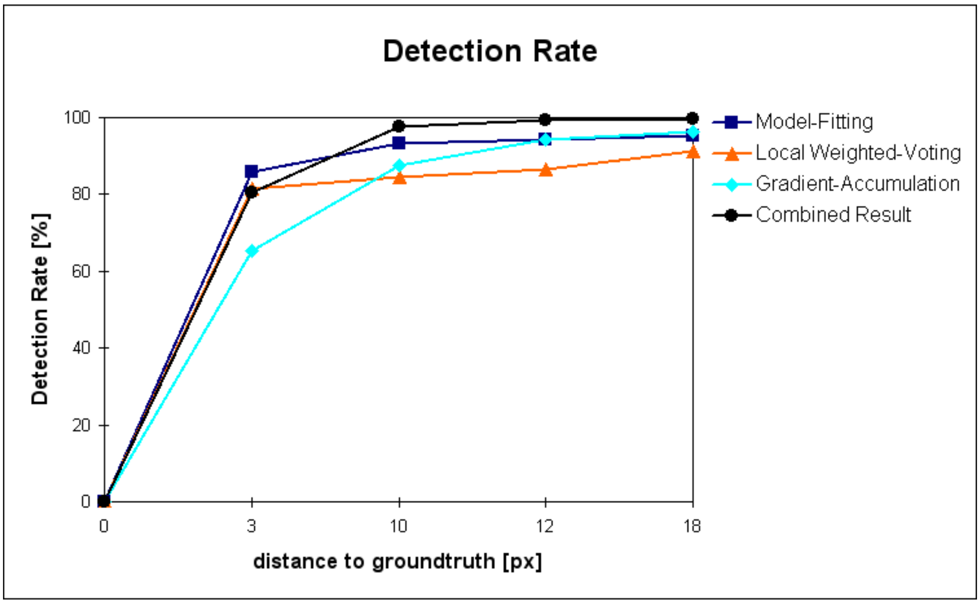


Figure 3.31: Detection rate of every method: "Base Sensor".

scan line. A noisy signal with so called "phantom lines" results as illustrated in Fig. 3.32. Consequently, the 2^{nd} scan line is only used for mat end or obstacle detection. The surface

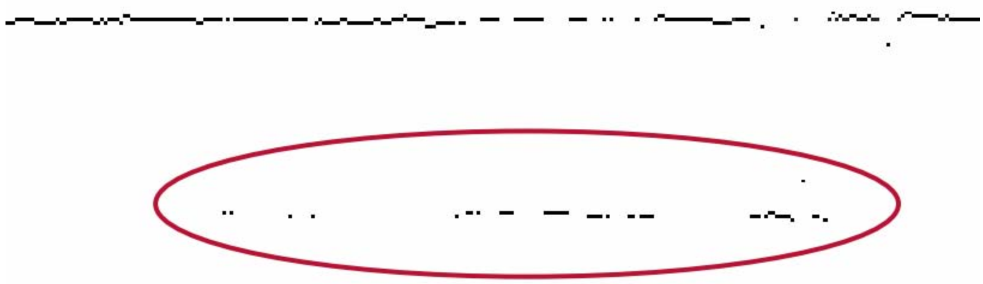


Figure 3.32: Phantom lines: In the 2^{nd} scan line (alternatively switched between two positions) the pixel with the maximum brightness is determined, thus a wrong pixel position is possible.

of carbon fibre mats is very uneven and reflects the laser light into different directions, causing a high amount of noise in the surface profile data. Fig. 3.33 illustrates the high amount of noise with five overlapping laser scan lines.

- Mat End Detection: The mat end is detected at the last moment if the alternately switched scan lines reach the mat end. The mat end will be detected if no possible edge is found with a minimum size of $0.3mm$.

- Obstacle Detection: Obstacles which are lying on the sewing surface are detected as soon as they are within the sensor range, as long as they are bigger than the

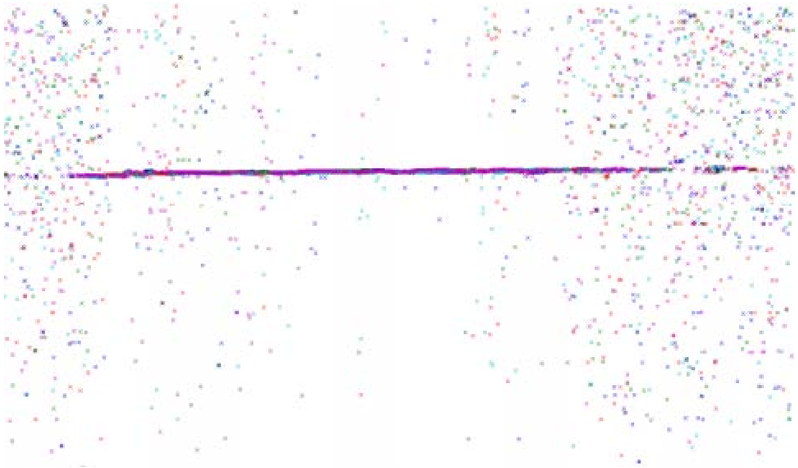


Figure 3.33: Five overlapping scan lines with 1024px *per scan line. A heavy noise with more than 40% is clearly visible.*

minimum size of 3mm. The obstacle is finally detected if it reaches the second alternatively switched position (near the needle) of the 2^{nd} scan line.

3.7.1 Runtime and Detection Rate

The runtime and detection rate granted a certain tolerance of 1$mm \widehat{=}$18px has also been evaluated for the new laser range scanner with 3 scan lines. The runtime of the three implemented methods is nearly the same as for the "Base Sensor": the pre-processing step needs approximately 0.15ms longer because of the heavy noise as illustrated in Fig. 3.33. With regard to the detection rate, the evaluation is based on more than 10,000 semi-automatic hand-labelled profiles from different mat types. Note that the mat type and the edge height has a strong influence on the detection rate. A positive detection is defined as a detection result within a tolerance of 1mm, which is totally sufficient for the sewing task. Tab. 3.3 and Fig. 3.34 show the achieved results. For the edge detection task, only the 1^{st} fixed laser scan line is used. The alternatively switched 2^{nd} laser scan line is used for the mat end and the obstacle detection.

Methods	$\pm 3px$	$\pm 10px$	$\pm 12px$	$\pm 18px$
Model-Fitting	44.2%	68.5%	68.8%	73.8%
Local Weighted-Voting	38.2%	63.3%	64.2%	69.2%
Gradient-Accumulation	20.3%	65.2%	69.3%	78.3%
Combined Result	43.2%	70.4%	74.6%	79.9%

Table 3.3: Detection rate of every method: "3 Lines Sensor".

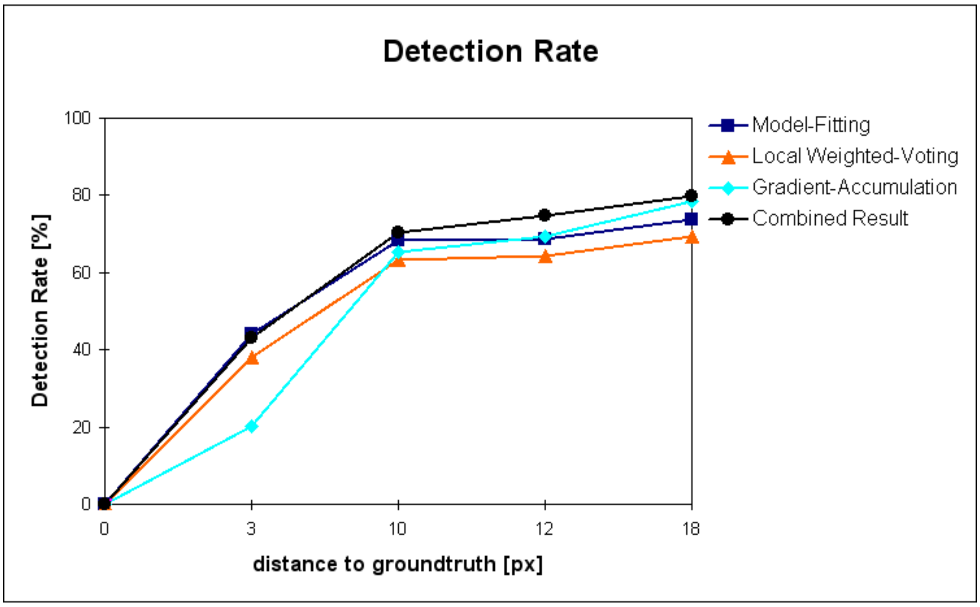
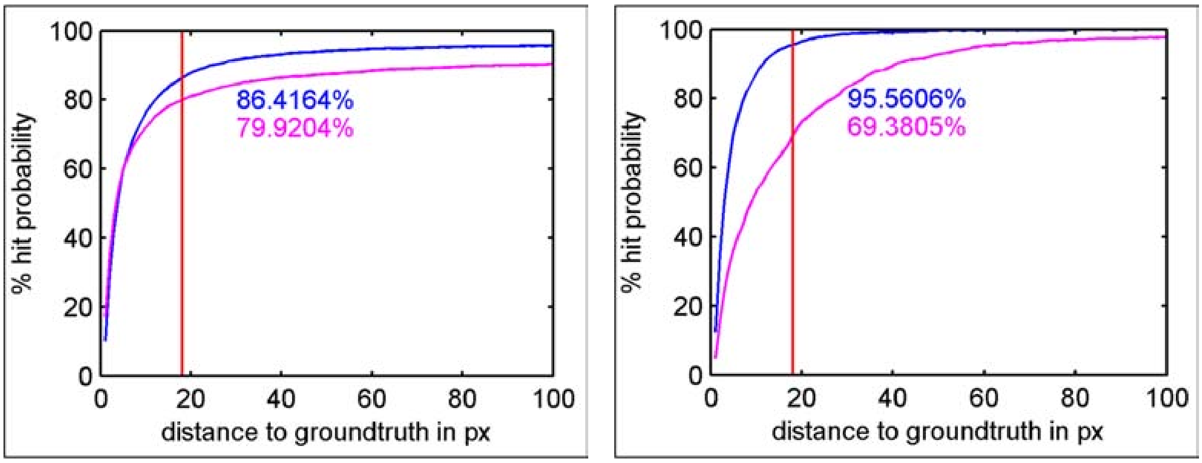


Figure 3.34: Detection rate of every method: "3 Lines Sensor".

3.7.2 Influence of Detection Algorithm Improvements

The results of the single data sets were accumulated. Due to the origin of the recordings of different mat types and scan positions, a general view of the detection quality has been provided by accumulating the results of all data sets. The original edge detection method for the "Base Sensor" provides a detection rate of 79.9% based on the scan lines from the "3 Lines Sensor". This value was improved to 86.4% by replacing the algorithm results with sum of weight functions and historic edge positions as illustrated in Fig. 3.35(a).

Fig. 3.35(b) shows that the average detection quality is at 95.56% for edges in the centre of the scan line (ignoring the influence of the mat type). The quality falls to 69.38% for edges at the boundaries of the scan line. A more detailed evaluation of the edge detection results, influence of edge positions, mat types, and improvements can be found in [Schöffmann, 2008].



(a) (b)

Figure 3.35: **(a)** *Influence of the weighting functions on the detection quality. The vertical red line marks the required accuracy of* 1mm*. The pink line shows the detection quality of the original edge detection method and the blue line shows the detection quality of the improved method [Schöffmann, 2008].* **(b)** *Influence of the edge position on the laser scan line with regard to average detection quality [Schöffmann, 2008].*

3.8 Discussion

A summary of the advancement for real-time edge detection algorithms is presented. The improvements focus on the treatment of very noisy and disturbed data. An efficient and stable method is developed to address the special needs of real-time as well as the importance of robust methods.

Contributions in this field have been made by introducing a real-time approach where existing and new developed edge detection methods are combined in a voting scheme to increase the edge tracking robustness [Richtsfeld et al., 2007a]. The novelty is based on different edge detection methods and their combination to carry out a robust edge tracking in the presence of outliers and artefacts in noisy range data using an industrial robot. The individual developed methods show very high reliability [Biegelbauer et al., 2007].

The experiments show that a two out of three voting over three methods achieves a better detection result than the individual methods. Thus the voting scheme for edge detection and localisation is suitable for use in related industrial applications under difficult conditions. Two different laser stripe sensors have been tested, whereby the developed edge detection methods and the voting scheme highlighted their flexibility and robustness.

Several detailed analyses of the data profiles were accomplished. These analyses showed good results during the evaluation of the "Base Sensor" data profiles in the beginning. However, the first evaluation of the "3 Lines Sensor" profiles produced bad results with the same methods, whereupon a new statistical method was implemented. The advanced edge detection method does not use a RANSAC-based line fit to smooth the detected edges because of the fast rotations of the sensor head during the stitching process. The average of an additional scan line leads to a higher redundancy of the data profiles, but at the same time the tracking of strongly curved seams requires an additional scan line.

The developed "3 Lines Sensor" pointed out clear disadvantages in comparison with the previously used "Base Sensor" with a single scan line. The sensor determines a possible pixel position for the 2^{nd} scan line (the brightest pixel point in the scan line) in each scan line of the CCD-chip. If no pixel position can be recognised in a line due to bad reflection or absorption characteristics of the material, the pixel with the maximum brightness from the previous scan line is determined. Consequently, a noisy signal with so called "phantom lines" results. To eliminate this effect with the help of a filter is not 100% possible, since with a stronger filter the searched edge can be also removed.

Project "REDUX" illustrates the feasibility of a continuous process chain for robotic stitching from CAD planning and a sensor guided edge detection to realise a path adjustment, which can be applied for a sewing process that realises lot-size-one production. In order to make an industrial application possible, still further research work must be investigated in order to develop a sensor that also works on the difficult CFRP material. In the stitching process, one must also consider, if it is essential to use a sensor, which has a very high frame rate.

Chapter 4

Detection of Cylindrical Objects in Tabletop Scenes

"You must do the thing you think you cannot do"
– *Eleanor Roosevelt*

Vision systems are increasingly used in the fields of industrial automation and home robotics. In the near future, service robots will support people to improve the quality of their lives.

One of the required key technologies is the grasping ability of the robot. Assistance robots must be able to perform tasks such as executing commands like "James, please bring me my cup!". In science fiction films like "Star Wars", all robots are able to do these jobs. In the real world, however, there is no robot able to support people in such a way at home.

The aim of this work is the detection and grasping of a cylindrical object with given shape parameters (radius and height as described in Tab. 2.1) from a clutter of different objects on the table, which the robot arm delivers to a target position or to the user. The developed method is robust under changing light conditions during a full day and suitable for soft real-time[1] processing.

Since there is no colour information available the objects cannot be detected and segmented so easily. In contrast to industrial applications, environmental influences like constant light conditions or pre-defined object positions cannot be assumed. That is (the reason) why the distance between the objects has to be calculated for a region-growing segmentation 2.4 step.

The cylinder grasping method is based on scanning the objects on the table with a rotating laser range scanner and the execution of subsequent path planning and grasping motions. The 2.5D point cloud obtained is analysed to detect the specific cylindrical object, whereupon the robot calculates and performs a collision-free path to the given cylindrical object and to the handover point.

2.5D means that the rear side of an object is not visible due to self-occlusion and the front side may also be occluded by other objects. In case of a possible failure, feedback is

[1] A soft real-time system tolerates some time delay to fulfil a defined task.

given to the user. The described system and the algorithm were tested during a full day of life demo presentation in Vienna at YO!tech[2] in 2007 and 2008.

The presented cylinder fitting method based on *Random Circle Fitting* (RCF) is compared to the standard *Least-Squares Cylinder Fitting* (LSCF) method. Additionally, the presented method can be easily extended to detect different given cylindrical objects. However, the drawbacks of the presented method are: inability for cover the detection of cylinders in an arbitrary position or incomplete range data of the cylindrical object.

Section 4.1 gives an overview of the developed system. Section 4.2 introduces a robust method to detect a cylindrical object in a noisy point cloud with different surrounding objects. In Section 4.4, the performance and the robustness of the algorithm is tested and the cylinder detection method is evaluated and compared to standard least-squares cylinder fitting. The comparison between the presented method and the standard least-squares cylinder fitting is realised on synthetically-generated models to guarantee perfect 2.5D range images, without any outside influences. Section 4.5 finally concludes this chapter with a short discussion about the evaluation results.

4.1 System Overview

Before the robot arm is able to handle the given cylindrical object, the system needs information about the position, orientation, and possible surrounding objects based on range images. A laser range scanner is used to acquire range images in which the 3D shape of the objects on the table has to be directly recorded. A detailed description of the used laser range scanner system is available in [Nössing, 2004].

The approach is based on scanning the objects on the table with a laser range scanner on a pan/tilt unit and the subsequent path planning and the final grasping motion. The spatial relation between the AMTEC[3] robot arm with 7DOF and the scanning unit is known.

First, the laser range scanner records the table scene and delivers a 2.5D point cloud. A high resolution sensor is needed in order to obtain a reasonable number of points of the objects with sufficient accuracy.

A red-light LASIRIS laser from StockerYale[4] with a wavelength of 635nm and a MAPP2500 CCD range camera from SICK IVP[5] mounted on a PowerCube Wrist from AMTEC robotics are used.

Fig. 4.1 shows that the robot arm is equipped with a human-like hand prosthesis of the company Otto Bock[6], which is used as gripper. The hand prosthesis has three active fingers: the thumb, the index, and the middle finger; the last two fingers are just for cosmetic reasons as illustrated in Fig. 4.2. It is a caliper gripper and can only perform approximately a tip grasp or a cylindrical grasp. The integrated tactile force sensors are

[2]YO!tech, http://www.yo-tech.at
[3]AMTEC robotics GmbH, http://www.amtec-robotics.com
[4]StockerYale Inc., http://www.stockeryale.com
[5]SICK IVP Inc., http://www.sickivp.se
[6]Otto Bock GmbH, http://www.ottobock.de

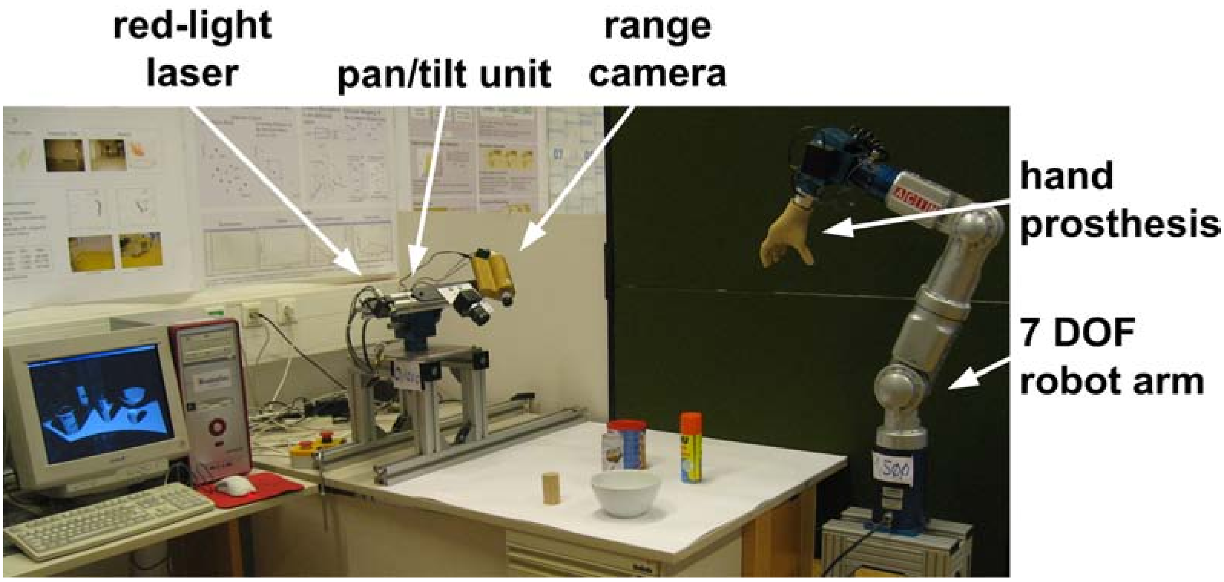


Figure 4.1: Overview of the system components and their interrelations.

used to detect a potential sliding of the grasped object, which initialises a readjustment of the grip force applied by the pressure of the fingers.

Figure 4.2: Hand prosthesis of the company Otto Bock. Caliper gripper with three active fingers: thumb, index, and middle finger. A tactile force sensor (SUVA sensor [Altrichter, 2008]) in the thumb is integrated to detect a potential sliding of objects and an additional sensor is integrated in the thumb leverage.

It is thought that people will accept this type of gripper rather than an industrial gripper due to the form and the optical characteristics. The virtual centre between the fingertip of the thumb and the index finger is defined as tool centre point (TCP) and the TCP can be optionally changed. The seventh degree of freedom of the robot arm is a rotational axis of the whole hand, required to enable complex object grasping or manipulation tasks and allow some flexibility for avoiding obstacles. When objects are positioned closer to each other, the grasping tests detect difficulties in grasping the cylindrical object. A minimum distance of $20mm$, equal to the diameter of the thumb of the hand prothesis, has to be observed between the objects.

4.2 Method for Rapid Cylinder Detection

This section presents the method for rapid cylinder detection. Fig. 4.3 outlines the model-based cylinder detection approach. It illustrates the main processing steps in gathering the range data, from the recorded table scene to the actual pose of the cylindrical object. The challenge of this approach leads to the segmentation of different objects on the table and the detection of the pose of the cylindrical object in a 2.5D point cloud.

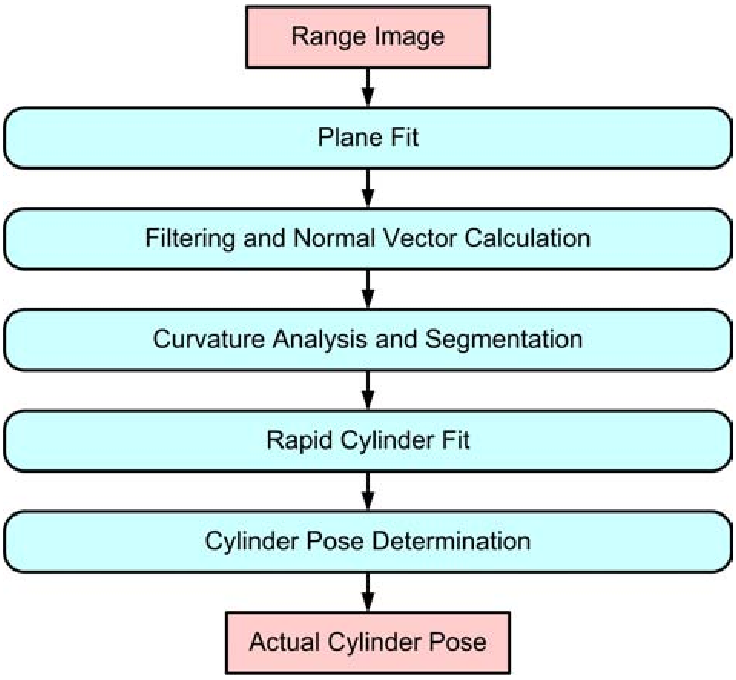


Figure 4.3: Flow chart of the cylinder detection approach.

Inspired by the work of Biegelbauer et al. [Biegelbauer and Vincze, 2000] demonstrating a method on how to detect a bore hole in noisy range data, based on circle fitting, the following method was developed. In [Biegelbauer and Vincze, 2000], the rough position of the bore hole was defined through CAD data and only an additional plane fit to detect the ground plane was needed. From above, the complete circumference of the aperture of the bore hole was clearly visible. The detected circumference points along the aperture were used to calculate the radius of the bore hole with a transformation in a defined 2D plane, based on the detected normal surface vector of the ground plane. Finally, a 2D circle fit was used to calculate the radius of the bore hole. The existence of a cylinder was not checked because the bore hole was defined through CAD data.

The difference between [Biegelbauer and Vincze, 2000] and the presented cylinder detection approach is the fact that there is no CAD data available and the searched cylindrical object is on a table in a large search area with cluttered adjacent objects. Additionally, the table top surface cannot be used to detect the radius of the cylindrical objects because parts of the rear side of the cylindrical object may be occluded by other objects or by the two shadows of the laser and camera, as illustrated in Fig. 4.4. For that reason only can the top curvature points be used for a 3D circle fit.

The proposed method analyses and segments the curvature of different objects on the table and detects a cylindrical object with a cylinder fitting method. The calculated normal surface vectors are used to calculate the final cylinder axis orientation with a 3D line fit.

Fig. 4.3 illustrates that the method starts with a dominant plane fit, followed by a geometrical filter to detect potential outliers and calculate the normal surface vectors in Section 4.2.2. Section 4.2.3 describes curvature analysis and the segmentation of the detected curvature points. The normal vectors are used to detect the strongest curvature points. Additionally, a line-based edge detection method is implemented to detect significant edge points. The segmentation of the detected curvature points is needed to detect the cylindrical object. The normal surface vectors are also needed for the rapid cylinder fit as described in Section 4.2.4. This section deals with the problem of determining the pose of the cylindrical object from 2.5D range data.

4.2.1 Dominant Plane Fit

Most of the table scene range images include the table plane. However, this plane is not needed for the object detection step, it slows down the object detection process and raises the likelihood of false detection. The first step is to detect and remove the raw data points of the table plane. In the cylinder detection approach, the table plane is defined as the dominant plane in the range image associated with more raw data points than any other planes in the rest of the scene. Finding the dominant plane is achieved by randomly fitting planes 100 times in the point cloud. A plane fit is generated by three randomly picked points. From three points the normal surface vector is calculated as the cross product of the direction vectors between two pairs of points. Each table plane hypothesis is verified against the other 99. After 100 calculated plane hypotheses, the one with the most points included with a defined threshold of $2mm$ wins. The distance of $2mm$ equals the average distance between two neighbouring points in 100 range images calculated with a kd-tree and closest point detection. Afterwards, all detected table points are used to calculate the normal vector of the table plane again to achieve higher accuracy. Fig. 4.4 illustrates the detected $17,277$ table plane points of the original $n = 75,863$ points.

The calculated normal vector $\vec{n}_t$ of the table plane P_t, a point of the plane p and the origin of the camera coordinate system O_c are used to remove all points under the plane with the plane equation:

$$\vec{n}_{t_x} p_x + \vec{n}_{t_y} p_y + \vec{n}_{t_z} p_z + D = 0, \tag{4.1}$$

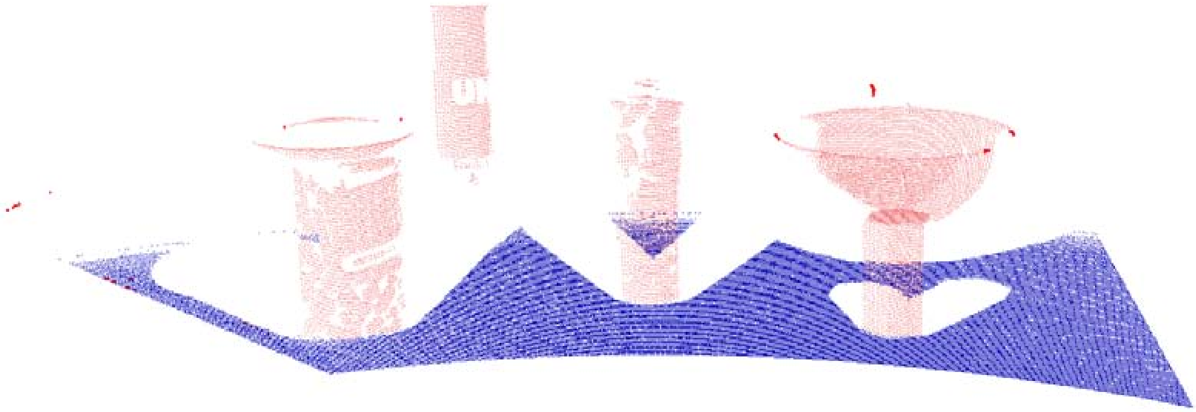


Figure 4.4: Raw point cloud with 75,863 points. The detected table surface is coloured blue. The two shadows from laser and camera are clearly visible. The noise and outliers detected by the geometrical filter are marked as red points.

$$D = -\vec{n}_{t_x} p_x - \vec{n}_{t_y} p_y - \vec{n}_{t_z} p_z. \tag{4.2}$$

The value F_c represents the position of the camera coordinate system to the table plane and F_{p_i} the position of the currently selected point p_i to the table plane:

$$F_c = \vec{n}_{t_x} O_{c_x} + \vec{n}_{t_y} O_{c_y} + \vec{n}_{t_z} O_{c_z} + D, \tag{4.3}$$

$$F_{p_i} = \vec{n}_{t_x} p_{i_x} + \vec{n}_{t_y} p_{i_y} + \vec{n}_{t_z} p_{i_z} + D. \tag{4.4}$$

If $F_{p_i} \cdot F_c > 0$, the point p_i and the origin of the camera coordinate system are on the same side, and if $F_{p_i} \cdot F_c < 0$, the point p_i is under the dominant plane and will be removed.

4.2.2 Raw Data Pre-Processing and Normal Vector Calculation

The remaining point cloud $\mathbf{P} = \{p_0, \ ... \ , p_{n-1}\}$ of the objects are filtered to reduce noise and outliers, which can arise by reflections. A geometrical filter based on the density allocation of the points is used to remove outliers and the threshold parameters are calculated with the help of the compression rate. This filter calculates the distance to the nearest neighbour based on a kd-tree [Bentley, 1975] for each point p_i, and then the minimum d_{min}, maximum d_{max}, and average d_a of these distances. The distances are used to calculate the compression rate τ:

$$d_a = \frac{d_{min} + d_{max}}{2}, \tag{4.5}$$

$$d_k = \frac{\sum_{i=0}^{n-1} d_i}{n}, \tag{4.6}$$

$$\tau = \frac{d_k}{d_a}. \tag{4.7}$$

Then all N_a points inside a sphere with the radius d_a and all N_k points inside a sphere with the radius d_k around a given point p_i are used to decide with the compression rate τ if the given point p_i is an outlier or not. If $\alpha < \beta$, calculated with Eq. (4.8) and Eq. (4.9), the given point p_i is an outlier and will be removed, as shown in Fig. 4.4.

$$\alpha = (d_a/N_a)^\tau \tag{4.8}$$

$$\beta = (d_k/N_k) \tag{4.9}$$

To successfully analyse the curvature of the objects in the table top scene all normal vectors of the points p_i are used. To achieve that the normal vector calculation of a point p_i is approximated by a planar surface patch close to the point: The surface patch is represented by a set of surrounding points $\mathbf{P} = \{p_1, \, ... \, , p_m\}$ of the given point p_i based on a local neighbourhood of $5mm$, where $p_i^\top$ is a 3D point of this set of m points, and the average of the point set is defined by:

$$\overline{p_x} = \frac{\sum_{i=1}^m \ p_{x_i}}{m}, \quad \overline{p_y} = \frac{\sum_{i=1}^m \ p_{y_i}}{m}, \quad \overline{p_z} = \frac{\sum_{i=1}^m \ p_{z_i}}{m}. \tag{4.10}$$

Using a local neighbourhood of $5mm$ shows best results with the used laser range scanner and is evaluated in 100 scans. The maximum distance between two closest points detected by nearest neighbour searching [Arya et al., 1998] over 100 point clouds is used. The average distance da of all 100 maximum distances is approximately $5mm$, as illustrated in Fig. 4.5.

The covariance of a coordinate pair is given by:

$$\operatorname{cov}(p_x, p_y) = \frac{1}{m-1} \sum_{i=0}^{m-1} \ (p_{x_i} - \overline{p_x}) \cdot (p_{y_i} - \overline{p_y}) \tag{4.11}$$

and the covariance matrix:

$$C = \begin{bmatrix} \operatorname{cov}(p_x, p_x) & \operatorname{cov}(p_x, p_y) & \operatorname{cov}(p_x, p_z) \\ \operatorname{cov}(p_y, p_x) & \operatorname{cov}(p_y, p_y) & \operatorname{cov}(p_y, p_z) \\ \operatorname{cov}(p_z, p_x) & \operatorname{cov}(p_z, p_y) & \operatorname{cov}(p_z, p_z) \end{bmatrix} \tag{4.12}$$

is the initial for the principal component analysis [PCA] [Xu et al., 1992]. The normal vector of the surface patch corresponds with the vector $\vec{n}$, which is determined by the eigenvalue problem $C\vec{n} = \lambda_{min}\vec{n}$, where λ_{min} is the smallest eigenvalue.

The determination of the normal vector with eigenvectors results in two possible and opposite directions. The correct normal vector orientation is defined by the normal vector with the smallest angle difference to the origin of the sensor coordinate system. Fig. 4.6(a) shows 10% of the calculated normal vectors. In Fig. 4.6(b), all normal vectors are coloured for a better visibility of the strongest curvature points. The (x, y, z) values of the normal vectors multiplied by 255 are allocated to (R, G, B) values.

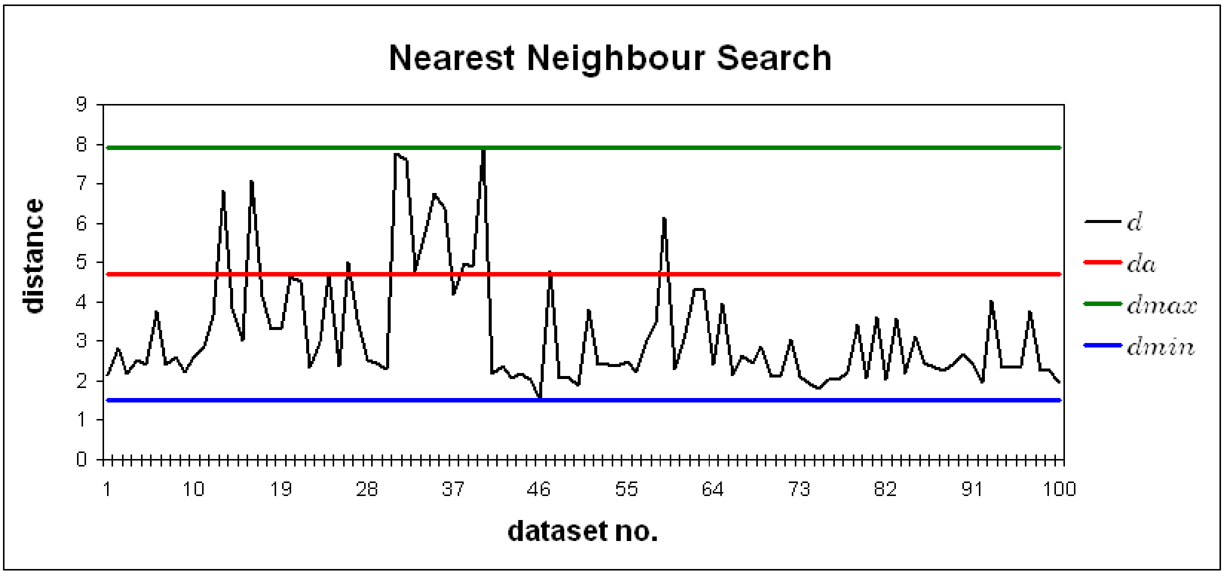
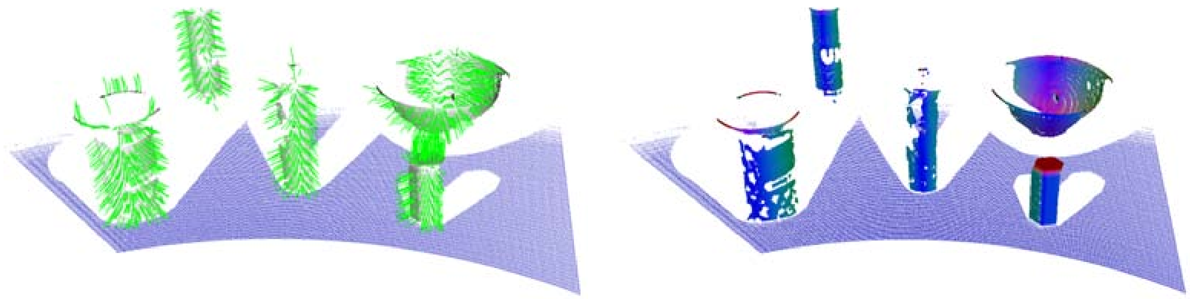


Figure 4.5: Maximum distances of two closest points detected by nearest neighbour searching [Arya et al., 1998] over 100 *whole point clouds. The average distance da of all maximum distances is approximately* 5*mm.*



(a) (b)

Figure 4.6: Close up of the calculated normal vectors. **(a)** *Representation of 10% of the calculated normal vectors. The correct normal vector orientation is defined by the normal vector with the smallest angle difference to the origin of the sensor coordinate system.* **(b)** *Coloured normal vectors for a better visibility of the strongest curvature points.*

4.2.3 Curvature Analysis and Segmentation

To robustly detect the given cylindrical objects on the table, the different objects have to be segmented. In this way, the individual objects are analysed, curvatures calculated, and (finally) the given cylindrical objects detected. Standard region-growing [Bab-Hadiashar and Gheissari, 2006] to segment all objects based on the whole point cloud $\mathbf{P} = \{p_0, \ldots, p_{n-1}\}$ without the table top surface is a very time-consuming step. A faster method is the segmentation of only the strongest curvature points of the objects. Based on the calculated normal vectors $\vec{n}$ and an angle-based edge detection method, the strongest curvature points are detected and only these points are used to detect the cylindrical object. After the curvature analysis, the detected curvature points are segmented into fragments. For that procedure, a region-growing based recursive flood-filling function [Burger and Burge, 2007] is used, which enables soft real-time processing for the cylinder detection approach, because of the reduced number of points.

The curvature analysis starts by detecting a set of surrounding points $\mathbf{P} = \{p_1, \dots, p_m\}$ of the given point p_i based on a local neighbourhood of $5mm$. The normal vectors $\vec{n}_t$ of these m points are used to calculate the angle difference to the given normal vector $\vec{n}_i$. If $\alpha_i > 20°$ (see Eq. (4.13)) the given point p_i is classified as a strong curvature point, as illustrated in green points in Fig. 4.7(a).

$$\cos \alpha_i = \sum_{t=1}^{m} \frac{\vec{n}_i \cdot \vec{n}_t}{||\vec{n}_i|| ||\vec{n}_t||} \tag{4.13}$$

Each point p_i of the point cloud is stored line by line, since as the table scene is recorded by the laser-stripe sensor. So to detect strong edge points, a local line-based edge detection method is used. The algorithm computes the distance respectively from the direction vector to the point in front p_{i-1}, and after from p_{i+1} to the given point p_i. Next, it calculates the angle β_i between these three points. If the outer angle β_i between these three points is smaller than 110°, the (given) local edge of three points fulfils the conditions as a strong curvature and the point p_i is classified as a strong curvature point, illustrated in red points in Fig. 4.7(a). An angle approximately of $\alpha = 20°$ and $\beta = 110°$, determined by 1000 trials, has an optimal balance between the edge detection result and the average curvature allocation of the cylindrical object:

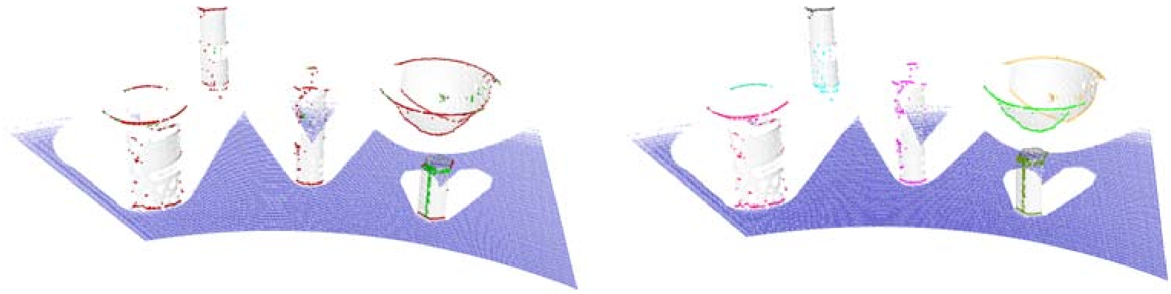
$$\cos \beta_i = \frac{(p_i - p_{i-1}) \cdot (p_i - p_{i+1})}{||(p_i - p_{i-1})|| ||(p_i - p_{i+1})||}. \tag{4.14}$$

An object or part is defined as a set of points with distances between neighbours below a threshold d_{ca}. A kd-tree [Bentley, 1975] is built to find neighbours and the recursive flood-filling function [Burger and Burge, 2007] is used to identify connected point sets. d_{ca} is the average distance between all m curvature points p_{c_i}, calculated by nearest neighbour point p_{cn_t} searching [Arya et al., 1998]. This step segments the curvature points of the different objects on the table into different components or fragments:

$$d_{ca} = \frac{\sum_{t=1}^{m} |p_{c_i} - p_{cn_t}|}{m}. \tag{4.15}$$

To belong to a fragment of the object, the distance d between a fracture element p_w and the given point p_i must be smaller than the average distance d_{ca}. Fig. 4.7(b) shows the segmentation result of the curvature points. These segmented curvature parts are used in the next step to detect the given cylindrical objects:

$$d = \sqrt{(p_{x_i} - p_{x_w})^2 + (p_{y_i} - p_{y_w})^2 + (p_{z_i} - p_{z_w})^2}, \tag{4.16}$$



(a) (b)

Figure 4.7: Curvature analysis and segmentation: **(a)** *Detected curvature points. The detected curvature points based on the normal vectors are coloured green and the edges based on a local line edge detection method are coloured red.* **(b)** *Segmented curvature fragments to produce a rapid cylinder detection.*

$$d < d_{ca}. \tag{4.17}$$

The segmentation of the $1,391$ strongest curvature points needs about $0.485s$ and enables soft real-time performance. In comparison, the segmentation based on recursive flood-filling function [Burger and Burge, 2007] of all $17,277$ object points requires about $30s$ and Fig. 4.8 shows the segmentation result of all object points [Richtsfeld et al., 2008].

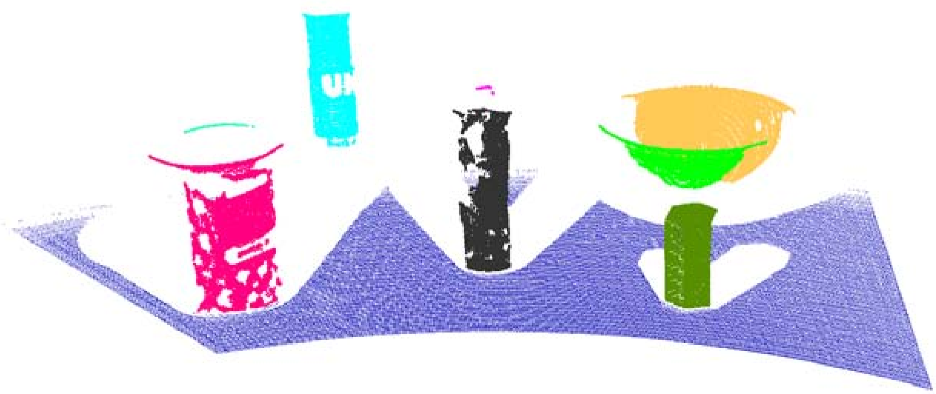


Figure 4.8: Segmentation result of the whole point cloud.

4.2.4 Rapid Cylinder Fit

The next step is a sequential model-based cylinder fit into the segmented top curvature points. The problem with global or local optimisation methods, e.g. the Levenberg-Marquardt method [Levenberg, 1944], [Marquardt, 1963], is the instant estimation of all parameters, which requires a good initial value. This method is very time-consuming because of iterative comparison of the complete object with the fit criteria.

The proposed method is a model-based cylinder fit starting with a sequential circle fit to the top rim curvature points to detect the given cylindrical objects and, finally, the position and orientation of the axis. For computational efficiency, the previously calculated normal vectors $\vec{n}_i$ are used to calculate the cylinder axis orientation. The task is to detect the given cylindrical objects based on the segmented top curvature points in the point cloud. Note that the top surface of the objects must be recorded with the range scanner to guarantee a successful detection of the cylindrical objects.

The next sections describe the circle fitting method into the top curvature points and the calculation of the final rotation axis to detect the pose of the cylindrical object.

- **Radius Calculation**

The estimation of the axis orientation and cylinder detection is realised by a 3D circle fit into the top curvature points. The top curvature points are detected with the previously calculated normal vectors. The normal vector in x-direction is bigger than in y- or z-direction, $n_x > n_y$ and $n_x > n_z$. The radius and the height of the given cylindrical object are used to detect the cylindrical object on the table based on the segmented curvature points. Standard least-squares circle fitting fails because of the cylinder fragments and due to the noise of 2.5D range data. As described in [Biegelbauer and Vincze, 2000], a robust method of estimating the radius can be achieved by circumscribing a circle to a triangle. Fig. 4.9 shows that three curvature points ($A,\ B,\ C$) of one segmented part are randomly picked and the radius r is calculated by:

$$\begin{aligned} d_{ca} &= (C-A)\cdot(B-A), \\ d_{ba} &= (C-B)\cdot(A-B), \\ d_{cb} &= (A-C)\cdot(B-C), \end{aligned} \tag{4.18}$$

$$\begin{aligned} n_1 &= d_{ba}\cdot d_{cb}, \\ n_2 &= d_{cb}\cdot d_{ca}, \\ n_3 &= d_{ca}\cdot d_{ba}, \end{aligned} \tag{4.19}$$

$$r = \frac{\sqrt{(d_{ca}+d_{ba})(d_{ba}+d_{cb})(d_{cb}+d_{ca})/(n_1+n_2+n_3)}}{2}. \tag{4.20}$$

The centre c_m of the circle is calculated by:

$$c_m = \frac{(n_2+n_3)A+(n_3+n_1)B+(n_1+n_2)C}{2(n_1+n_2+n_3)}. \tag{4.21}$$

The triangle area A and the circumference u are calculated by the radius r:

$$A = r^2\pi, \tag{4.22}$$

$$u = 2r\pi. \tag{4.23}$$

To realise a robust radius calculation, i.e. a robust cylinder detection in noisy range data, the random selection of three curvature points and the radius calculation is repeated

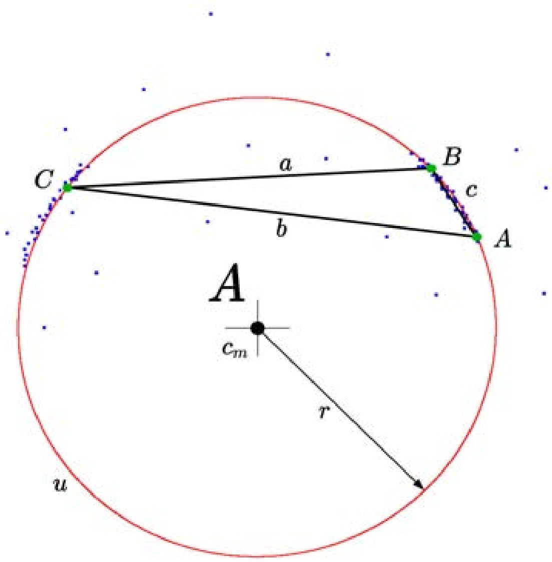


Figure 4.9: Radius calculation by circumscribing a triangle built from three randomly picked points of the segmented top curvature points of the cylinder fragment.

100 times and the minimum distance d_{min} of all t top curvature points p_c of the segmented fragment to the circumference wins:

$$d_j = \sum_{i=0}^{t-1} |r - p_{c_i}|, \tag{4.24}$$

$$\arg \min_{1 \leq j \leq 100} d_j. \tag{4.25}$$

Then all top curvature points of the fragment are used to decide if the analysed fragment is part of a cylindrical object with a range tolerance of $2mm$. Remember, the distance of $2mm$ equals the average distance between two neighbouring points in 100 scans. For an explicit description, the curvature points are defined as p_c, and c_m is the previously calculated circle centre with a radius r. The error must be smaller than a defined threshold:

$$|\|\vec{p_c} - \vec{c_m}\| - r| \ \leq \ 2. \tag{4.26}$$

If more than 80% of the top curvature points agree with the calculated circle, the analysed fragment is defined as a fragment of a cylindrical object. A threshold of 80% is chosen to eliminate potential noise. Then all radii r_f of the f detected circle fragments are used to detect the cylindrical object with the known radius r_{def}, and the radius with the smallest deviation wins:

$$\arg \min_{1 \leq i \leq f} |r_i - r_{def}|. \tag{4.27}$$

The result of the circle fitting method is illustrated in Fig. 4.10(b). The curvature points of this run and all remaining points along the circle axis p_{b_i} with a threshold of $2mm$, which fulfil the fit criteria, are examined more closely with the cylinder fit to calculate the 3D pose of the cylinder axis. The points p_{b_i} are illustrated in Fig. 4.11 as blue points.

- **Rotation Axis Calculation**

To finally compute the 3D pose of the cylinder axis, the cylinder normal surface vectors $\vec{n}_i$ and the calculated radius r are used. Fig. 4.10(a) illustrates that all normal surface vectors of the cylinder barrel approximately cross the cylinder axis. This fact is used to calculate the cylinder axis $\vec{v}$. Points of the cylinder axis p_{a_i} can be found by multiplying all normal surface vectors $\vec{n}_{b_i}$ of the points p_{b_i} with the calculated radius r. It must be guaranteed that the normal vectors arise in the direction of the cylinder axis. The axis of the cylinder corresponds with the vector $\vec{v}$, which is determined by the eigenvalue problem $C\vec{v} = \lambda_{max}\vec{v}$ and λ_{max} is the largest eigenvalue of the complete point set. The final axis vector $\vec{v}$ is calculated by a 3D line fit.

The calculated cylinder axis $\vec{v}$, the top curvature points, the detected table normal surface vector $\vec{n}_{tab}$, and the height h of the cylindrical object are used to calculate the circle centre c_m again. This step is important to guarantee that the circle centre c_m is along the cylinder axis $\vec{v}$.

Fig. 4.10 shows 10% of all calculated normal vectors and the result of the axis fit. The final cylinder axis is defined by a vector $\vec{v}$ and a point c_g which is the centre of gravity of the cylindrical object. c_g is calculated with the the cylinder axis vector $\vec{v}$, the circle centre c_m, and the height h of the selected cylindrical object:

$$c_g = c_m - \frac{h}{2} \cdot \vec{v}. \tag{4.28}$$

Fig. 4.11 shows the final result of the cylinder detection process. All points p_{b_i} which fulfil the fit criteria are illustrated as blue points. In this case, the task was to detect two given cylindrical objects. Alg. 1 explains the final axis pose estimation.

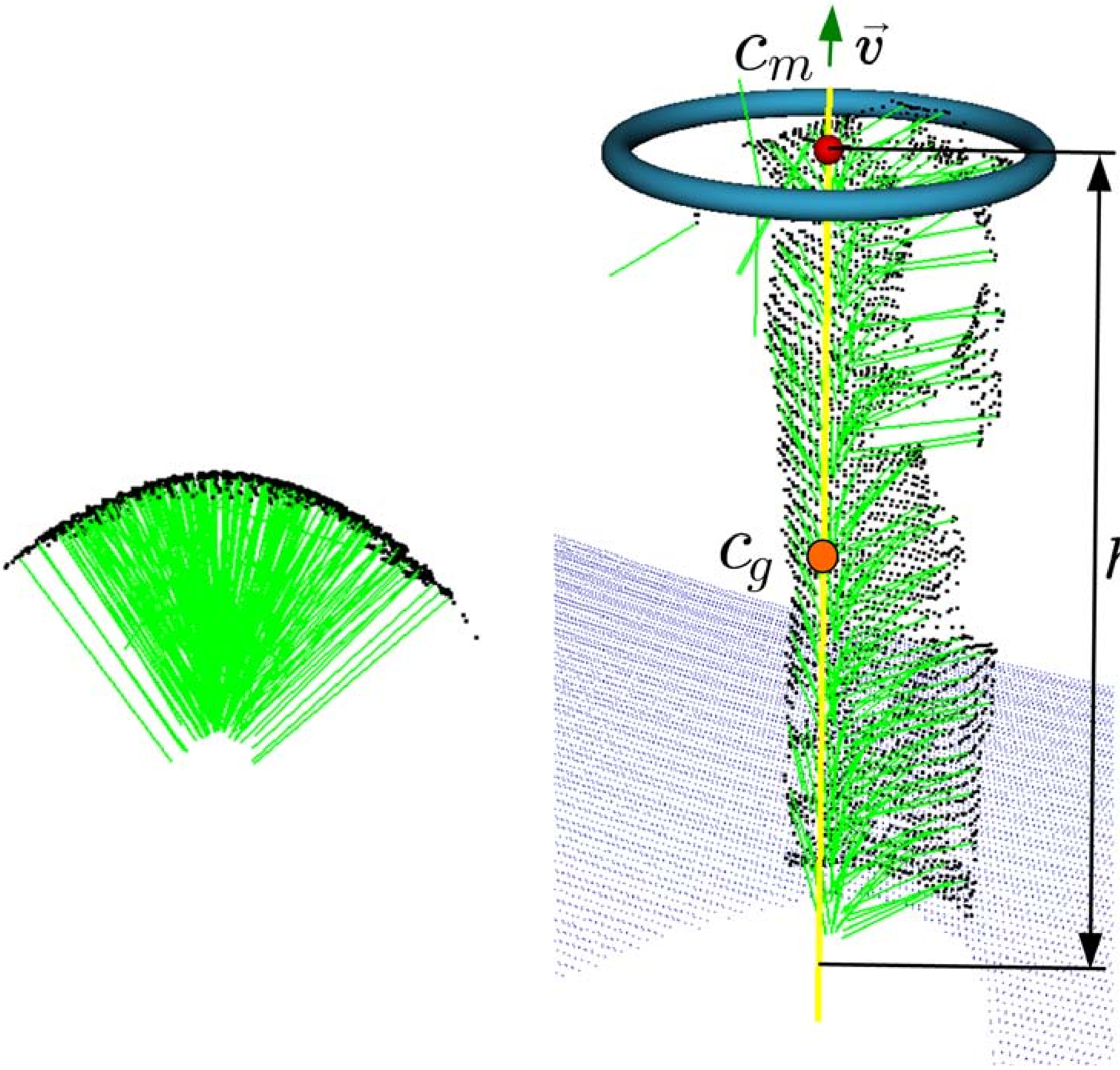
Algorithm 1 *Axis Pose Estimation.*

```
begin
    calculate cylinder axis points: p_{a_i} = p_{b_i} + n_{b_i} · r
    3D line fit across the cylinder axis points p_{a_i} to calculate the cylinder axis: v
    calculation of the circle centre c_m again
    calculation of the centre of gravity of the cylinder: c_g
end
```

4.3 Planning of the Robot Motions

The task of this section is to calculate a collision free robot path and to execute the grasping activity safely. The commercial path planning tool THOR (Tool Handling the Operations of Robots) from the company AMROSE robotics[7] is used. The disadvantage of this path planning tool is the fact that it is RANSAC-based, which means that for

[7]AMROSE robotics Inc., http://www.amrose.dk



(a) (b)

Figure 4.10: **(a)** *Distribution of the normal surface vectors $\vec{n}_{b_i}$ to calculate the final cylinder axis by a 3D line fit. The ends of the calculated normal vectors determine the cylinder axis.* **(b)** *Results of the axis fit. The final cylinder pose is determined by the the centre of gravity c_g and the axis orientation vector $\vec{v}$.*

the same situation and grasping pose different trajectories are possible. The first step is performed by the path planning tool from AMROSE robotics. The input into this tool is the detected cylinder pose, the environment model and a transformation between the robot coordinate system and the range scanner coordinate system. All objects on the table and the detected cylindrical object have to be transmitted to the path planner as a mesh in stl-file format (Stereo Lithography file format), calculated by a 3D Delaunay triangulation [O'Rourke, 1998]. These object models are important for the path planner to calculate a collision-free trajectory to the desired object. The TCP of the gripper is

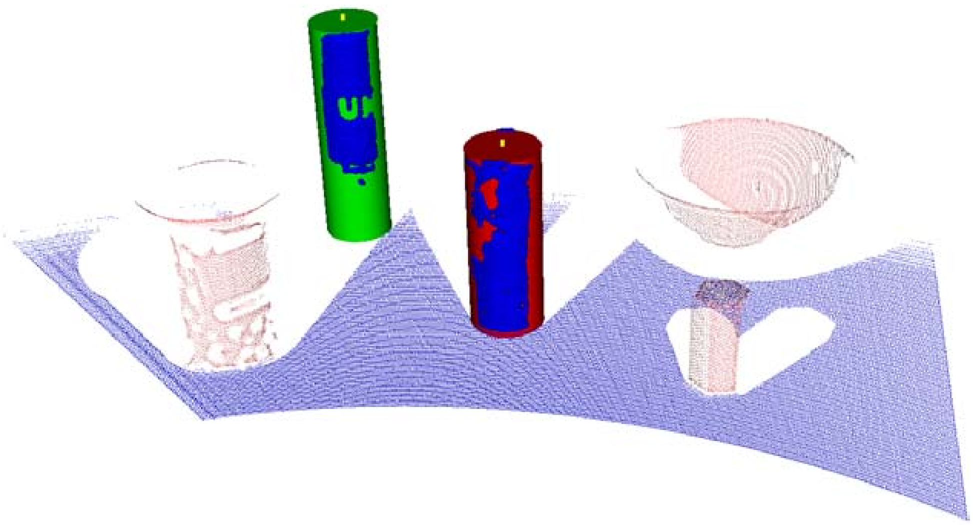


Figure 4.11: Final result of the cylinder detection process. The normal surface vectors of the marked-blue points p_{b_i} are used to calculate the final cylinder pose. In this case two given cylindrical objects are successfully detected. The shape parameters (radius and height) of the third cylindrical object are not given.

defined as the centre between the thumb, the index, and the middle finger.

The TCP of the gripper and the calculated centre of gravity c_g of the cylindrical object agree in the final grasping pose. It must be guaranteed that both the centre c_g of the cylindrical object and the cylindrical object itself are high enough, so that the gripper or the robot arm does not collide with the table. Otherwise the path planner is not able to calculate a collision-free path. In addition, the grasping approach of the robot arm and the grasping orientation must be defined, which can result in difficulties, especially when several objects surround the cylindrical target object on the table.

The start position to achieve the final grasping pose is defined as $200mm$ above the selected cylindrical object along the calculated cylinder axis $\vec{v}$ to guarantee that the robot and the gripper do not collide with the table or one of the surrounding objects. The calculation of the final grasp orientation of the hand prosthesis is part of the path planning tool and depends on the position of all the surrounding objects. Before the grasping task is approved, the user can check a simulation of the calculated trajectory and decide if it is safe enough to handle the object (see Fig. 4.12). Then the robot arm executes the offline programmed trajectory and the user can initiate the closing of the gripper, which initiates feedback about the successful execution of a grasping task. As soon as the gripper encloses the object, the robot motion to the transfer point starts. Finally, the desired object can be placed at a defined position or be directly handed over to the user. The algorithm is implemented in C++ using the Visualisation Tool Kit (VTK)[8].

[8]Freely available open source software, http://public.kitware.com/vtk

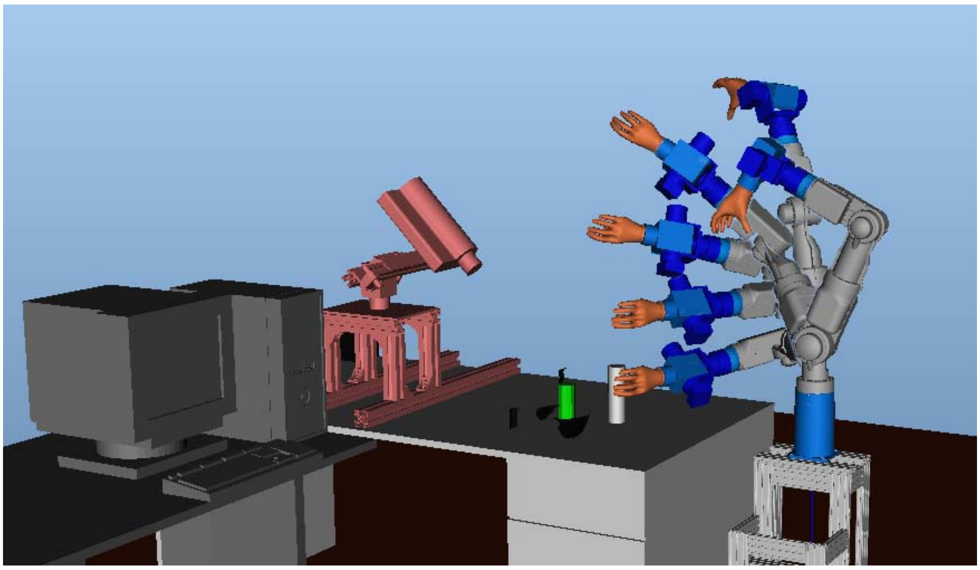


Figure 4.12: Visualisation of the trajectory with a simulation tool. The white cylinder is the grasping object. The green cylinder is a second given grasping object and the black object fragments are the obstacles.

4.4 Experimental Evaluation

First, the performance of the cylinder detection method itself is evaluated and then the results of the method are compared with a standard least-squares cylinder fitting method. Additionally, this section demonstrates that an approach based on random samples is very efficient with respect to the computational cost. The presented method shows robustness during a live demo presentation in Vienna at YO!tech[9] in 2007 and 2008. On the demonstration day, about 50 runs were performed, during which two given cylindrical objects with $r = 25.75mm$, $h = 173mm$ and $r = 26.75mm$, $h = 134mm$ were to be detected in the point cloud and grasped by the robot. The main problem that rarely appeared was a malfunction of the path planning tool, because no suitable trajectory could be found and the path planning system had to be restarted. Sometimes the last two fingers shift the grasp object, but without any effect on the success of the grasping process. Tab. 4.1 shows a short analysis of the problems within 50 runs. The presented method and does not suffer from inaccuracy and has practical implications.

problems	number of events	influence	[%]
path planning	11	11	22
hand prosthesis	4	0	0
object recognition	2	2	4
$\sum$	17	13	26

Table 4.1: Evaluation of the problems as a percentage [%] during 50 *runs.*

[9]YO!tech, http://www.yo-tech.at

The cylinder detection approach is performed by a PC with 2.0GHz Intel Core 2 Duo processor and takes about $4.6s$, see Tab. 4.2, depending on the range image size. For visualisation of the results, VTK has been used and for time measurement, the proposed algorithm has been implemented in C++.

calculation steps	t [s]
plane fit	1.406
geometrical filter	1.938
normal vector calculation	0.312
curvature analysis	0.469
curvature points segmentation	0.485
cylinder detection	0.015
$\sum$	4.625

Table 4.2: Duration of every calculation step for the presented point cloud in Fig. 4.11.

Next, the results of the *Random Circle Fitting* (RCF) and the standard *Least-Squares Fitting* (LSF) method are presented. The processing time depends on the range image size, the normal surface vector calculation, and the curvature analysis, whereby the last two steps are most time-consuming. It is possible to reduce the radius to find the point neighbours for the local normal surface vector calculation. In this work the radius is set to $5mm$ (explained in Section 4.2.2), but with a smaller radius the required time for the normal vector calculation is dramatically reduced. However, to get accurate normal vectors, a bigger radius is essential. As described in Section 2.5, the least-squares fitting (LSF) of geometric models requires an iterative optimisation process, while the proposed method is based on three random points and their associated circumscribing circle. The last column in Tab. 4.3 shows the required pre-processing time (ppt) for the normal surface vector calculation and the curvature analysis.

The influence of the range image size on the fitting result for a synthetically-generated cylinder with $r = 10mm$, $h = 100mm$, $\vec{v} = \{1, 0, 0\}$, and of the original $20,000$ points is presented, and the number of iteration steps is fixed to 100 for the LSF and the RCF method. Tab. 4.3 illustrates the influence of the number of points of the cylinder fragments on the computation time. The tests were performed on a reduced resolution starting from $1,000$ points (5%) up to $20,000$ points (100%). Fig. 4.13 illustrates the incremental advancement of the radius deviation and angle deviation by a rising resolution of the point cloud.

Tab. 4.3 and Fig. 4.13 shows that both methods find nearly the same average radius with $r = 9.4642mm$ for the LSF method and $r = 9.6816mm$ for the RCF method over all fragments. It is clearly visible that the pre-processing step is the most time-consuming step and if it is required to realise segmentation or curvature analysis, RCF will be faster than LSF because the normal surface vectors needed are already calculated. In this case, the advantage of RCF is the computational cost. For up to $2,000$ points, RCF will be a little comparative faster than LSF.

The last point of the performance evaluation is a comparison of the presented cylinder fit with a standard least-square cylinder fit on synthetically-generated 2.5D range data

method	no. of points	%	r [mm]	$\lvert\Delta r\rvert$ [mm]	$\lvert\Delta \mathbf{v}\rvert$ [°]	$\sum$ t [s]	ppt [s]
LSF	20,000	100	9.60	0.40	0.0193	1.11	-
RCF	20,000	100	10.21	0.21	0.0146	6.23	6.22
LSF	10,000	50	9.56	0.44	0.0223	0.56	-
RCF	10,000	50	9.78	0.22	0.0210	1.68	1.67
LSF	4,000	25	9.48	0.52	0.0326	0.24	-
RCF	4,000	25	9.74	0.26	0.0320	0.32	0.31
LSF	2,000	10	9.35	0.65	0.0396	0.13	-
RCF	2,000	10	9.70	0.30	0.0390	0.10	0.09
LSF	1,000	5	9.29	0.71	0.0618	0.06	-
RCF	1,000	5	9.62	0.38	0.0525	0.02	0.02

Table 4.3: Comparison of the accuracy and the computational effort of a cylinder fit using LSF and RCF method for 100 *iterations by a synthetically-generated 2.5D cylinder with* $r = 10mm$. *The cylinder fragments have a resolution starting from* 1,000 *points (5%) up to* 20,000 *points (100%).*

of cylindrical objects. The synthetically-generated 2.5D cylinders have exact dimensions of $r = 25.75mm$, $h = 173mm$ for cylinder 1 with 29,953 points, and $r = 26.25mm$, $h = 134mm$ for cylinder 2 with 34,981 points. Regarding the fitting criteria, the final radius, the orientation and the computation time are investigated. To keep the results comparable, the cylindrical object points are synthetically-generated and the radius to find the point neighbours is set to $3mm$. Searching for a local minimum requires a good starting pose to converge to the global minimum. The PCA algorithm is used to get a good starting pose of the cylinder axis. The pose estimation based on this method works well for full 3D point data of the cylindrical object. However, with the obtained 2.5D point data, the misplacement in the first iteration steps is rather high. Tab. 4.4 compares the results of the presented RCF method and the standard LSF method with 100 iteration steps. In this case $\lvert\Delta \mathbf{v}\rvert$ is the solid angle deviation of the final cylinder axis to the cylinder axis of the synthetically-generated cylindrical object. The time requirement of the RCF method depends on the range image size as presented beforehand and the pre-processing step required for cylinder 1: $2.594s$ and for cylinder 2: $4.078s$.

method	cyl. no.	pose est.	iterations	r [mm]	$\lvert\Delta r\rvert$ [mm]	$\lvert\Delta \mathbf{v}\rvert$ [°]	t [s]
LSF	1	PCA	100	27.732	1.982	0.107	1.656
RCF	1	-	100	26.318	0.568	0.047	2.625
LSF	2	PCA	100	28.264	2.014	0.153	1.812
RCF	2	-	100	26.655	0.405	0.069	4.110

Table 4.4: Comparison of the accuracy and the computational effort of a cylinder fit using LSF and RCF method for 100 *iterations by synthetically-generated 2.5D point clouds.*

The following Fig. 4.14, 4.15, 4.16, 4.17, 4.18 and 4.19 show that an increased number of samples does not improve the result, but the influence of the outliers slightly decrease.

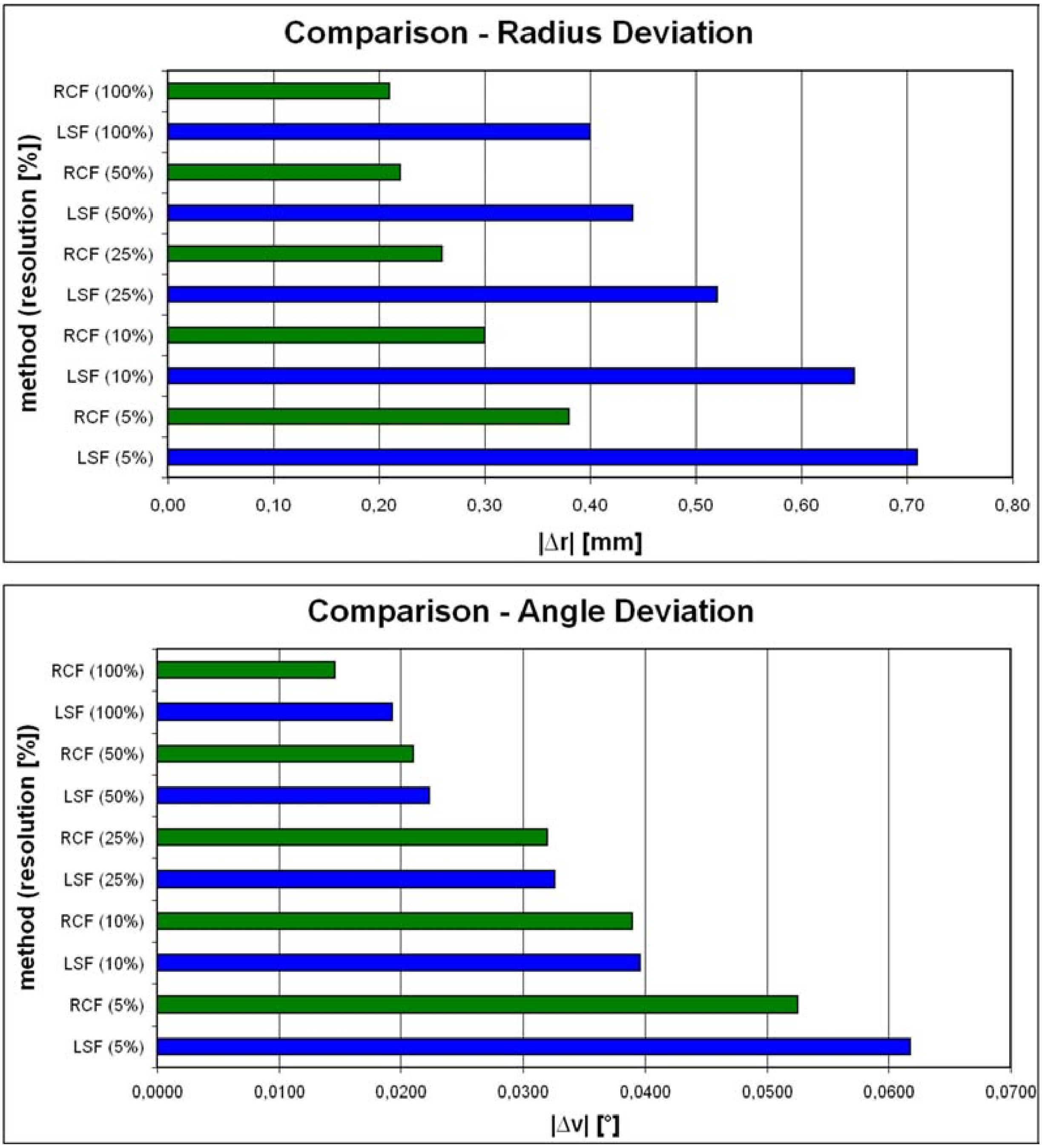


Figure 4.13: Comparison of the accuracy of a cylinder fit using LSF and RCF method for 100 *iterations by a synthetically-generated 2.5D cylinder with* $r = 10mm$*. The cylinder fragments have a resolution beginning from* $1,000$ *points (5%) of up to* $20,000$ *points (100%).*

Fig. 4.14 and Fig. 4.17 illustrate that the RCF method clearly achieves a more exact cylinder radius with less samples from the radius calculation and also better results for the angle deviation as the LSF method, see Fig. 4.15 and Fig. 4.18. The disadvantage of the presented method is the need for more time, especially for the normal surface vector calculation depending on the range image size as illustrated in Fig. 4.16 and Fig. 4.19. Nevertheless, it is essential for several different applications in range image processing to calculate the normal vectors and to realise a curvature analysis as well as adjacent methods, for which the presented method is absolutely eligible.

To conclude the experimental evaluation, the presented fitting method for cylindrical objects achieves reliable and robust results compared to standard least-squares fitting. The processing time depends on two factors: The first, that the RCF-based method achieves fast and robust results depending on the raw data points and the second, that the calculated normal surface vectors and curvature points are used for the cylinder fit.

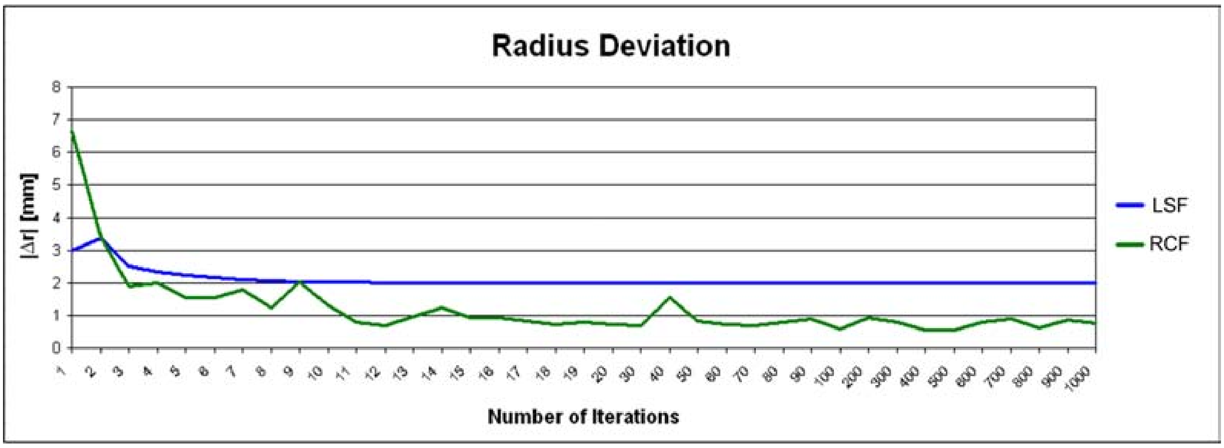


Figure 4.14: Cylinder 1 (29,953 points): Radius deviation.

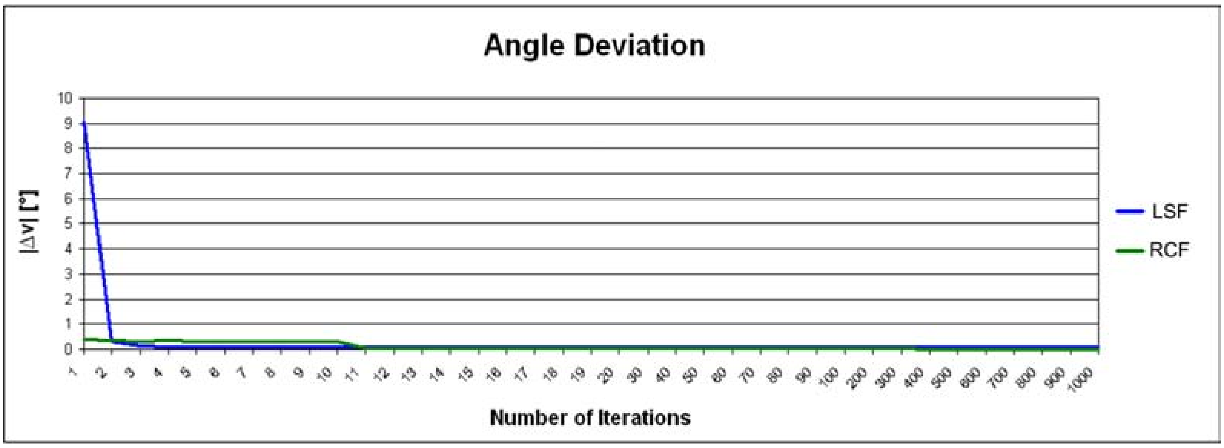


Figure 4.15: Cylinder 1 (29,953 points): Angle deviation.

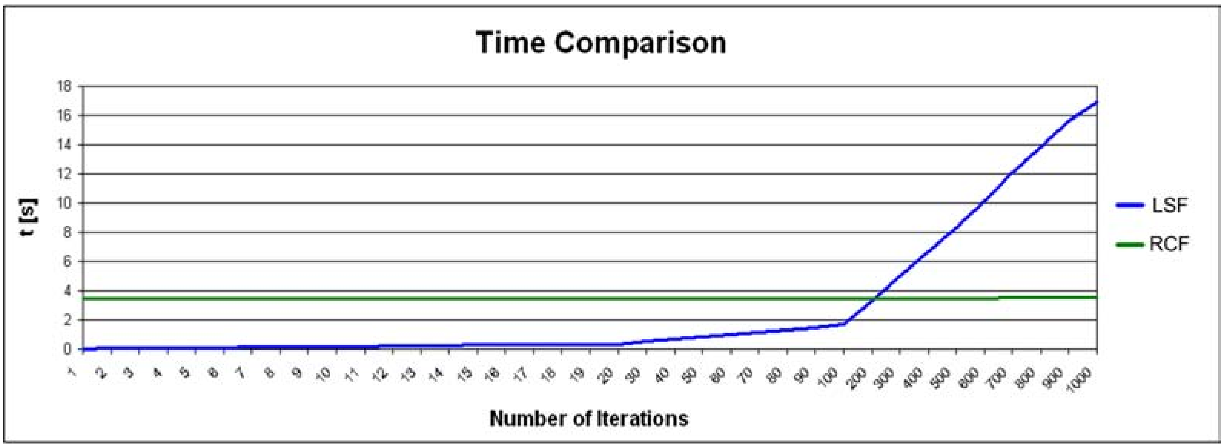


Figure 4.16: Cylinder 1 (29,953 points): Time comparison.

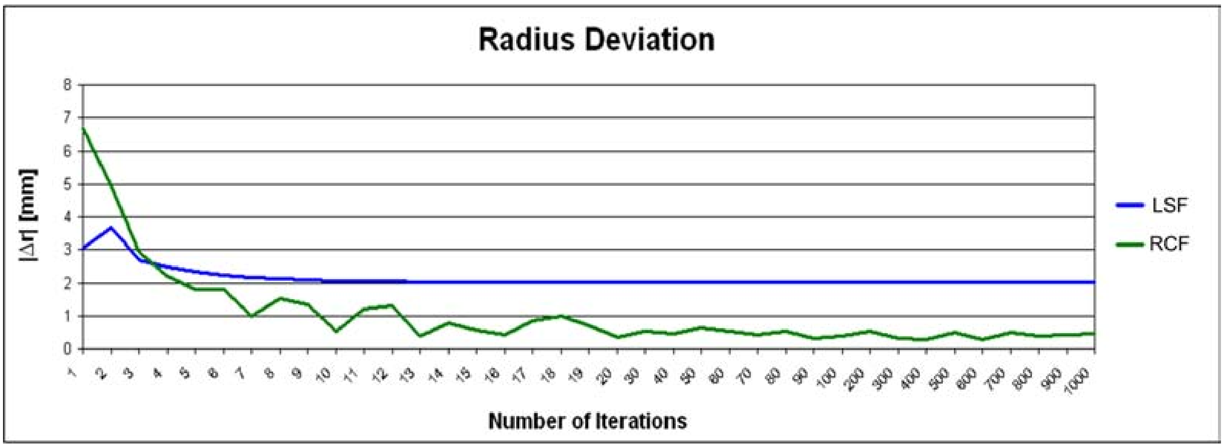


Figure 4.17: Cylinder 2 (34,981 points): Radius deviation.

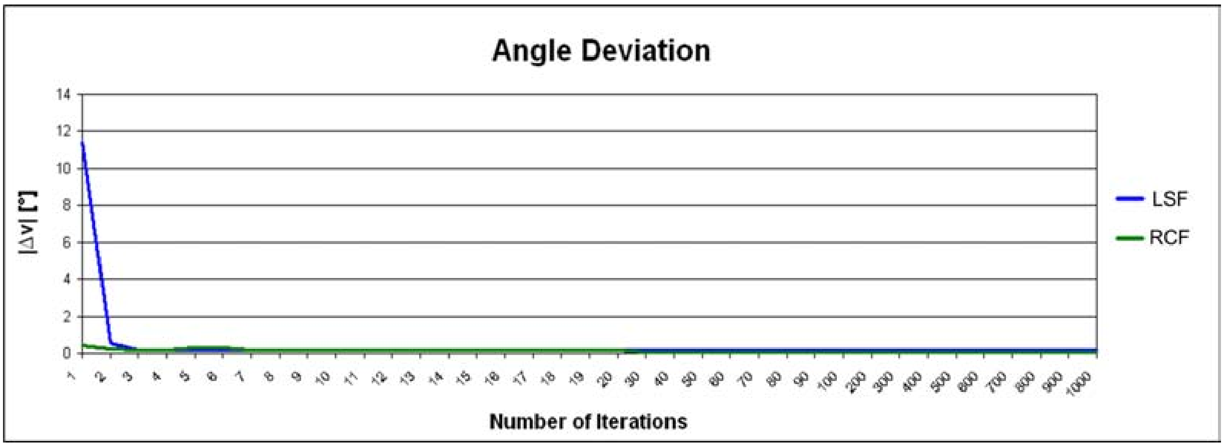


Figure 4.18: Cylinder 2 (34,981 points): Angle deviation.

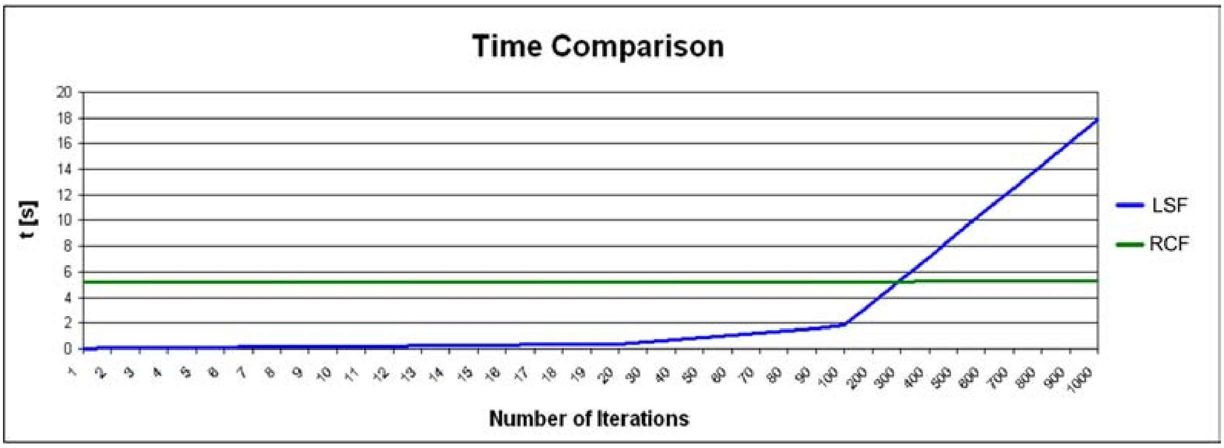


Figure 4.19: Cylinder 2 (34,981 points): Time comparison.

4.5 Discussion

This chapter presented a system with a fixed robot arm and a scanning unit on a table, which is able to detect and grasp given cylindrical objects with cluttered adjacent objects in soft real-time.

In the field of home robotics, the requirements of full 3D data, noiselessness, and obstacle-free situations are often not provided. The contribution of this work is a fast and practical method optimised for fitting cylinders in sparse and noisy range data under difficult and changing light conditions recorded from a single view. The improvements focus on the treatment of different objects on the table. The system must distinguish between them and detect the given cylindrical object.

First, the table surface is extracted by a RANSAC-based plane fit and the curvature of the objects is analysed by the normal surface vectors and a local edge detection method. The segmented curvature fragments are used to detect the cylindrical object by a RANSAC-based 3D circle fit to handle the typical outlier problem. The final circle axis is calculated with the normal surface vectors of the cylinder barrel points, the calculated radius, and a 3D line fit.

The experimental evaluation shows that the presented fitting method for cylindrical objects achieves reliable results in comparison to standard least-squares cylinder fitting. Thus the deviation of the radius, the angle deviation, and the calculation time are evaluated more precisely. The radius deviation and also the angle deviation are essentially smaller than with the standard least-squares cylinder fitting method. The processing time depends on the number of raw data points of the whole point cloud and the radius to detect neighbouring points needed to calculate the normal surface vector of a given point. Additionally, the time-consumption is linear with the number of iterations. In comparison with other standard fitting approaches, the proposed algorithm shows reliable and robust results in a fraction of time without the pre-processing step to calculate the normal vectors. While the least-squares fit does not need to calculate normal vectors, segmentation does and therefore must be calculated. One interesting point is that the presented method shows better results with a reduced resolution of the point cloud. Different resolutions were also analysed to confirm this point of view.

The entire system exhibited its practical behaviour at a live demo presentation. During the demonstration day, about 50 runs were performed in which the cylinder detection failed in only 2 cases, whereas path planning failed in 11 cases. The result indicates that this strategy is feasible to complete a grasping task automatically under this framework.

Chapter 5

Grasping Point and Pose Detection of Unknown Objects on a Tabletop

"Science is everything we understand well enough to explain to a computer.
Art is everything else."
– *David Knuth*

In the last few decades, the problem of grasping and manipulation of unknown objects in a fully automatic way has gained increasing importance, mainly due to the wide-spread use of service and rehabilitation robotics [Casals et al., 2000, Martens and Ruchel, 2001, Ivlev and Martens, 2005]. The grasping of unknown objects from a single view is an especially difficult problem because the pose and shape of different objects are unknown, and the possible hand configurations to perform a stable grasp are numerous.

The aim of this work is the detection and grasping of arbitrary objects from a clutter of different objects on the table, which the robot arm delivers to a target position or to the user.

The problem of an automatic 2.5D reconstruction to obtain practical grasping points and poses consists of several challenges. An object might be detected in several disconnected parts, due to missing sensor data from shadows or poor surface reflectance. Without any model information, adjacent objects may appear as one object in the recorded point cloud and hidden objects on the rear side of larger objects cannot be detected. However, due to the symmetry of most everyday objects, one view is often sufficient to realise a stable power grasp. As described in Section 2.7.1 [Brouwer et al., 2009] pointed out that for humans only the view of one side of the object is sufficient to successfully grasp an object.

This chapter presents a robust method of the segmentation of a 2.5D point cloud into shapes, assembly of rotationally symmetric parts into objects, and calculation of grasping points and poses despite noise, outliers, and shadows. The algorithm was developed for arbitrary unknown objects in different poses, on top of each other or side by side with a special focus on rotationally symmetric objects. If objects cannot be separated because they are stacked on top of each other they are considered as one object. A grasp will disambiguate the situation. The algorithm detects and merges clipped rotationally symmetric parts, because this object class can be robustly identified [Richtsfeld and Vincze, 2008a] and allows a cylindrical grasp as well as a tip grasp [Schulz et al., 2005] along the top rim. For all other arbitrary objects, the presented method calculates potential power grasps based on the top surfaces with a 3D model of the gripper as illustrated in Fig. 5.1. The shape information recovered from a single view is too limited and no complete object information is available, hence, to calculate force-closure grasps.

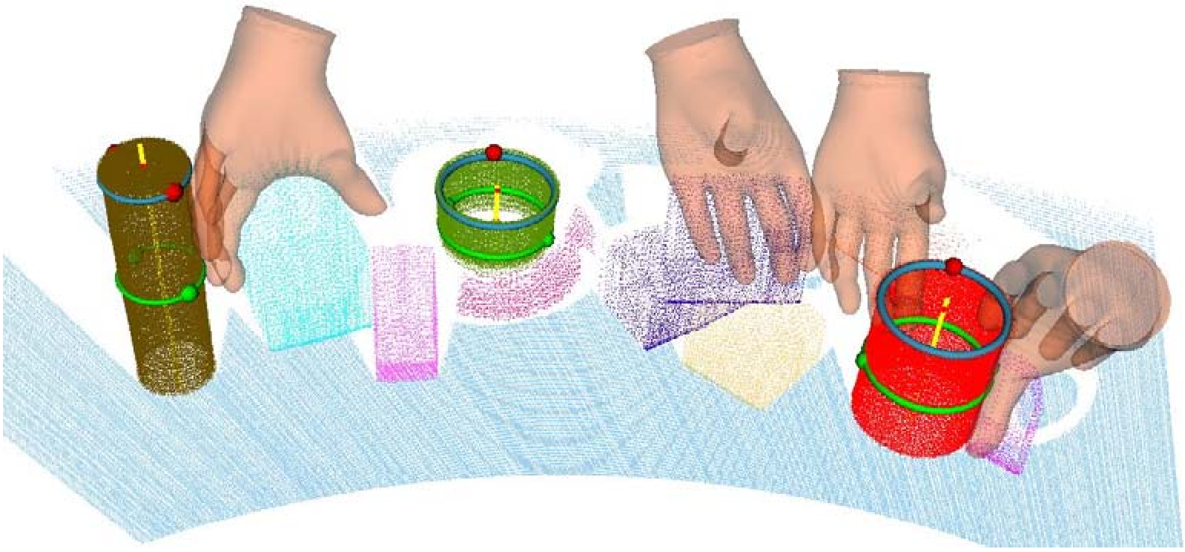


Figure 5.1: Detected grasping points and poses. The green points display the grasping points for rotationally symmetric objects. The red point shows an alternative grasp along the top rim. The illustrated hand poses show possible power grasps for the remaining graspable objects.

For a general evaluation of the presented method, the following categories have been defined: serval similar objects in different poses, connected objects, and scattered objects in a box as well as a pile of objects. Tests based on laser range and dense stereo data have been carried out to illustrate the robustness of the proposed algorithm. The experimental results demonstrate the effectiveness of the proposed method to grasp a wide range of objects.

This work was supported by the EU-Project "GRASP" with the grant agreement number 215821. The aim of this project is the design of a cognitive system to perform grasping and manipulation tasks in unforeseen situations. This project goes beyond the classical perceive-act or act-perceive approach and implements a predict-act-perceive paradigm based on human brain research. These consolidated findings are used to develop methods for the grasping process, since input information images and point clouds from table top scenes are used.

The outline of this chapter is as follows: Section 5.1 specifies different methods of segmenting a 2.5D point cloud into parts and the assembly of parts into objects, and characterises the merging of clipped rotationally symmetric objects. For the segmentation step, region-growing and mesh segmentation were tested with a special focus on the computation time. Section 5.2 details the calculation of grasping points for rotationally symmetric objects and optimal grasping poses for arbitrary objects to grasp and manipulate an object without collision. Section 5.3 shows the achieved results and Section 5.4 finally concludes this chapter.

5.1 Method of Grasping Point and Pose Detection

This section presents the developed method to grasp unknown objects in detail starting with the flow chart in Fig. 5.2 outlining the main components and their interactions. It illustrates all processing steps from scanning the objects on the table to obtain the range image, grasping point and pose detection by the grasping algorithm, and path planning for the final grasping step.

A flexible interaction and feedback between robot and user is necessary. In doing so the robot sends a feedback to the user about the successful execution of the grasping task, by measuring the aperture angle of the gripper with the SUVA sensor, see Fig. 4.2. The robot has to detect the position and orientation of the object among all surrounding objects on the table. This includes an autonomous feature detection and grasp motion planning to fulfil the task of providing objects in an arbitrary pose on a table to the user. The pose of the objects on the table is unknown, thus the algorithm tries to grasp the topmost object.

Therefore, the path planner sends feedback to the grasping algorithm about the successful calculation of a possible trajectory to reach the grasping object and the grasping algorithm obtains information about a successful grasp. This information is needed by the path planner because the grasped and handled object can be eliminated in the table scene as a potential obstacle for the next grasping or path planning task. Another possibility is to scan the tabletop scene again.

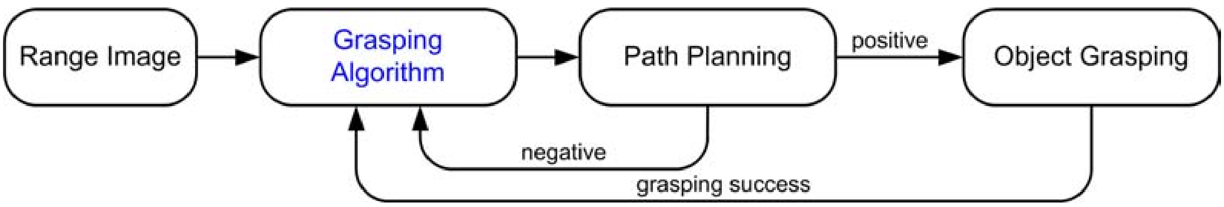


Figure 5.2: Interactions between all computation steps.

Fig. 5.3 presents the main parts of the grasping algorithm. It illustrates the processing steps from raw data pre-processing to the grasping point and grasping pose detection. The proposed method presents a robust way of calculating potential power grasps of unknown objects without collision using a 3D model of the used gripper.

The challenge of this approach is the segmentation of different objects on the table and the detection of the grasping points and poses in a 2.5D point cloud. Additionally, an object might be detected as being composed of several disconnected parts, due to missing sensor data from shadows or poor surface reflectance.

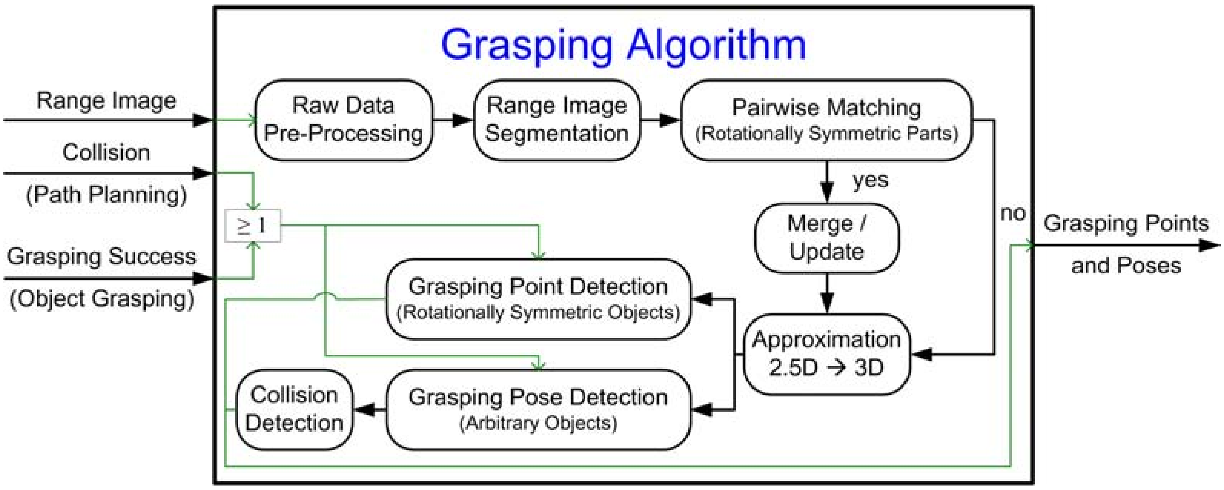


Figure 5.3: Overview of the grasping algorithm.

As illustrated in Fig. 5.3 the grasping algorithm consists of six main steps:

- Raw Data Pre-Processing: An image of the tabletop surface is created by pre-processing raw data points with a plane fit, and a geometrical filter or smoothing filter is used to reduce noise and outliers.

- Range Image Segmentation: This procedure identifies different objects or parts of fragments in the point cloud.

- Pairwise Matching: Finding top curvature points, indicating the top rim of an object part, fit a 3D circle to these points, and merge clipped rotationally symmetric objects.

- Approximation of 2.5D Objects to 3D Objects: This step is only important to detect potential collisions with the path planning tool. The algorithm distinguishes between:

- Rotationally Symmetric Objects: Adds additional points by using the calculated rotation axis.

- Arbitrary Objects: The non-visible sides of the objects will be closed with planes which are normal to the table plane.

- Grasping Point and Pose Detection:

- Grasping Point Detection: Rotationally symmetric objects allow a cylindrical grasp as well as a tip grasp along the top rim for open objects.

- Grasping Pose Detection: The power grasp for arbitrary objects is calculated with a 3D model of the used gripper.

- Collision Detection: Considers all surrounding objects and the table surface as potential obstacles, in order to evaluate the calculated grasping pose.

The proposed algorithm segments different objects on the table and calculates possible grasps for unknown objects. The system receives the table scene by an unstructured point cloud scanned from one single view. As described in Section 5.1.1, the method starts with a pre-processing step to detect the table surface with a plane fit in the range image followed by a geometrical filter to detect potential outliers and a smoothing filter to reduce noise. Section 5.1.2 presents two different segmentation methods, which have been tested and compared to get best segmentation results dependent on time. Section 5.1.3 presents the detection and merging of clipped rotationally symmetric object parts, since this object class can be robustly identified. The approximation of 2.5D objects to 3D objects as described in Section 5.1.4 is only important to detect potential collisions by the path planning tool. Section 5.2 describes the main part of the proposed grasping algorithm. The grasping point detection algorithm is limited to rotationally symmetric objects, which differentiates between open and closed objects. In addition to this object class, cylindrical grasps are calculated by the path planning tool. If no suitable grasp can be detected because of collision, an alternative tip grasp along the top rim will be calculated. For all arbitrary objects, the method calculates potential power grasps and considers all surrounding objects and the table surface as obstacles.

5.1.1 Raw Data Pre-Processing

Almost every range image of a tabletop scene includes the table surface. However, the table surface is not needed for the grasping point and pose detection. Hence, the first step is to detect and remove the tabletop plane points associated with the ground plane in the raw data points. The plane is only used to detect potential collisions of the final grasping poses and can be used as reference for the robot as described in Section 4.2.2. The table plane is normally the dominant plane in the range image associated with more raw data points than in the rest of the point cloud. This definition fails in cases with many objects on the table as illustrated in Fig. 5.4(a).

The robust detection of the table plane is achieved by calculating all normal surface vectors, which are used to evaluate all raw data points for the plane fit [Stiene et al., 2006]. On the normal vector $\vec{n}$, the calculation of a point p is approximated by a planar surface patch close to the point with a radius of $5mm$. This $5mm$ (radius) corresponds to the average distance between the two closest points over 100 whole point clouds, as discussed in Section 4.2.2. The normal vector of the surface patch corresponds to the vector $\vec{n}$ which is determined by the eigenvalue problem $C_n = \lambda_{min}$, where λ_{min} is the smallest eigenvalue and C is the covariance matrix of the surface patch point set as defined in Eq. (4.12).

At the start, a geometrical filter can be used to reduce noise and outliers, depending on the setup and the environmental influences.

The tabletop scene is recorded with the range scanner mounted on the side over the table, in order to overlook a large range of the table as illustrated in Fig. 4.1. To detect the deepest plane (associated with the table plane) the convex hull $\mathcal{H}_{ull}$ of the complete point set with n points (see Fig. 5.4(a)) is calculated:

$$\mathcal{H}_{ull} = ConvexHull\left(\bigcup_{i=0}^{n-1} p_i\right). \tag{5.1}$$

Fig. 5.4(b) shows the convex hull of the complete point set. The blue vertices[1] of the convex hull correspond with the outside points of the original raw data points. These m points are used to detect the deepest plane; by testing all possible planes on the convex hull. Fig. 5.4(c) shows the three detected hull vertices representing the deepest plane; Fig. 5.4(d) illustrates the resulting normal vector $\vec{n}_{ref}$ calculated with the cross product of the direction vectors between two pairs of the three detected points. The point p_{ref} on this plane is calculated by the circumcentre of these three plane points.

The normal vector $\vec{n}_{ref}$ and the point p_{ref} are used to detect all points of the raw data, which are part of the table plane with a normal distance of $5mm$ (average of the maximum distances of the two closest points detected by the nearest neighbour when searching over 100 whole point clouds (see Fig. 4.5)) to the plane as illustrated in Fig. 5.5(a). It is simply practical to use a threshold to minimise the number of points and to speed up the calculation time because only the points in the defined threshold are used for the final plane fit.

Finding the final table plane is achieved by randomly fitting planes in the evaluated table points 100 times. After 100 calculated plane hypothesis the one with the most included points with a defined threshold of $2mm$ wins. The threshold of $2mm$ is motivated by the average distance between two neighbouring points in 100 range images, calculated by closest point detection. Afterwards all detected table points are used to calculate the final normal surface vector of the table plane $\vec{n}_{tab}$ again to achieve higher accuracy as illustrated in Fig. 5.5(b).

Fig. 5.5(b) shows the remaining point cloud of the objects, which should be filtered to reduce noise and outliers. A geometrical filter based on the density allocation of the points is used to remove outliers and the threshold parameters are calculated with the help of the compression rate. A detailed description can be found in Section 4.2.2. To reduce noise, the point cloud of the objects is smoothed with an average filter.

5.1.2 Range Image Segmentation

Range image segmentation is one of the biggest challenges in computer vision. A correct segmentation result of the different objects on the table is an inalienable part of calculat-

[1] In geometry, a vertex is a special kind of point, which describes the corners or intersections of geometric shapes and a polygon is a set of faces.

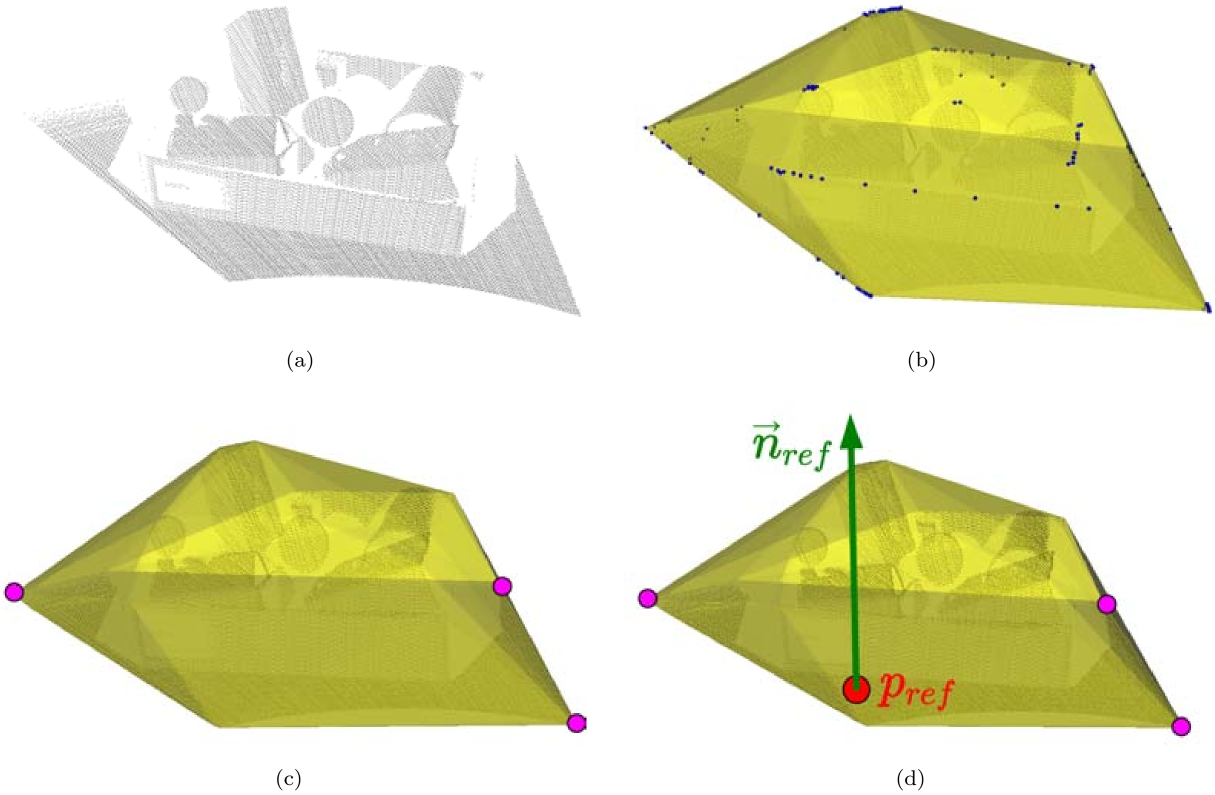
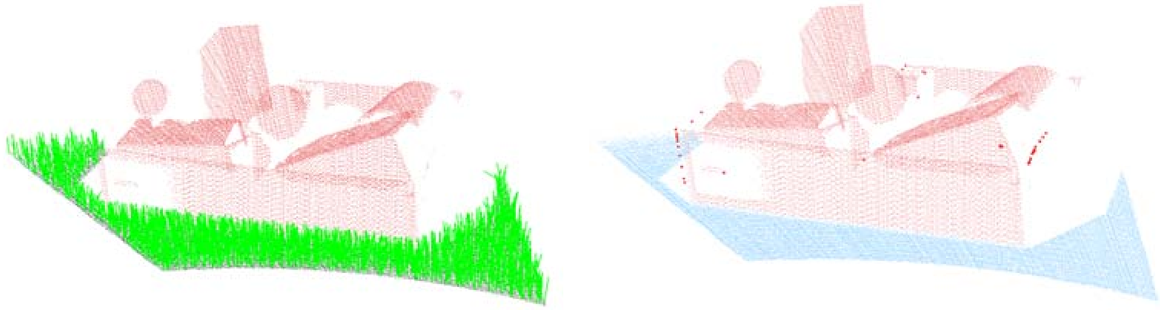


Figure 5.4: Detection of the normal surface vector of the table plane. **(a)** *Raw point cloud with* 42, 023 *points.* **(b)** *Convex hull of the complete point cloud. The* 126 *blue points illustrate the vertices of the convex hull, which correspond to the outside points of the original point cloud.* **(c)** *The blue vertices are used to detect the deepest plane of the convex hull, which is defined by the three coloured magenta vertices.* **(d)** *Calculated normal surface vector $\vec{n}_{ref}$ defined by the three vertices of the deepest plane. The red point p_{ref} on this plane is calculated by the circumcentre of these three plane points.*

ing grasping points and poses. This section presents two different possible segmentation methods and evaluates the results with a special focus on the computation time. For this comparison the region-growing algorithm [Bab-Hadiashar and Gheissari, 2006] and a segmentation method based on a mesh generation are tested.

An object or part is defined as a set of points with distances between neighbours. For that a kd-tree [Bentley, 1975] is generated and the minimum d_{min}, maximum d_{max}, and average distance d_a between two closest points points [Arya et al., 1998] as input information for the segmentation step are calculated. The result may contain an over- or an under-segmentation depending on the overlap of the objects as illustrated in Fig. 5.6 and Fig. 5.7.



(a) (b)

Figure 5.5: Exposure of the raw point cloud with 42, 023 *points. The detected table surface is coloured blue. The two shadows of the laser and the camera are clearly visible. The noise and outliers detected by the geometrical filter are marked red.* **(a)** *Evaluated points, which are used to calculate the final table plane. Representation of* 10% *of the calculated normal vectors of the evaluated points.* **(b)** *The detected table surface includes* 19, 646 *points. After the filtering step the final point cloud includes* 22, 339 *points of the original* 22, 377 *object points.*

- **Region-Growing**

Region-growing methods often give very good segmentation results that correspond well with the observed objects in the table scene. Region-growing approaches exploit the fact that points which are close together are part of the same object. Different characteristics, like the average distance between all points, colour, texture, and size can be chosen as similarity criteria.

As explained in Alg. 2, the presented region-growing algorithm is based on the recursive flood-filling function [Burger and Burge, 2007] to identify connected point sets. For this function, the average distance d_a over all remaining points after the plane detection and filtering step is used as input information for the segmentation step. As described in [Richtsfeld and Vincze, 2008a] the distance d is defined as euclidian distance with an additional weighting factor w_g derived from the angle between the normal vectors $\vec{n}$ of the neighbouring points p_i and p_m:

$$d = \sqrt{(p_{x_i} - p_{x_m})^2 + (p_{y_i} - p_{y_m})^2 + (p_{z_i} - p_{z_m})^2}, \tag{5.2}$$

$$\cos\alpha = \frac{\vec{n_i} \cdot \vec{n_m}}{\|\vec{n_i}\| \|\vec{n_m}\|}, \tag{5.3}$$

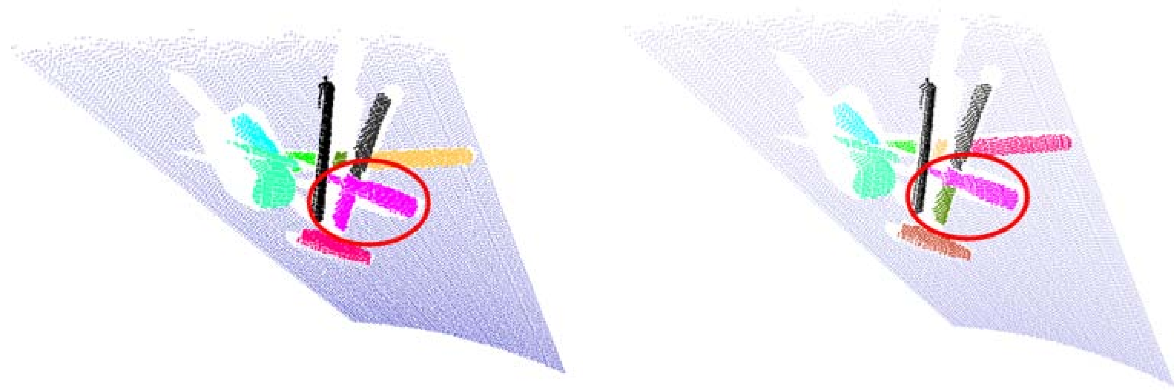
$$w_g = 1 - |\cos(\alpha)|. \tag{5.4}$$

If $d \cdot w_g < d_a$ then the considered point p_i is part of the same region as the point p_m. Fig. 5.6 illustrates that the weighting factor w_g derived from the angle between the normal vectors $\vec{n}$ of two neighbouring points has an influence on the segmentation result and helps to segment objects placed on top of each other.

Algorithm 2 *Overview of the region-growing algorithm.*

begin
1. choose the start point (more than one start point can be chosen)
2. take a neighbouring point and add it to the region if it has a smaller distance with the weighting factor than the average distance
3. repeat step 2. for each of the newly added points
4. stop if no more points can be added and choose a new starting point, which is not part of a detected region

end



(a) (b)

Figure 5.6: Segmentation result based on region-growing. Objects placed on top of each other are encircled red. **(a)** *Region-growing result based on distances only.* **(b)** *Region-growing result based on distances with additional weighting factors w_g. The successfully segmented fragments are clearly visible. However, the extension with the help of the normal vectors works by an intersection angle $> 60°$, otherwise an under-segmentation results as clearly visible by the overlapping objects on the left side (green fragments).*

- **Mesh Segmentation**

The segmentation of a 2.5D point cloud can be also achieved with a mesh generation based on the triangles calculated by a 3D Delaunay triangulation [O'Rourke, 1998], as explained in Alg. 3.

Delaunay triangulations are used to build topological structures from unstructured points. If points are injected into the triangulation, the algorithm finds the closest previously inserted point. Edelsbrunner's α-shape algorithm [Edelsbrunner and Mücke, 1994] is implemented to realise the 3D Delaunay triangulation. This method is described in detail in Section 2.6.1.

The necessary settings for the mesh generation will be achieved with d_{min}, d_{max}, and d_a between all neighbouring points [Richtsfeld and Vincze, 2008b]. d_{max} is the needed alpha value for the circumsphere. A detailed description for mesh generation based on a Delaunay triangulation can be found in Section 2.6.1. Then all segments of the mesh are extracted by a triangle connectivity filter [Belmonte et al., 2004]. This step segments the mesh into different objects or parts of fragments. An additional cut refinement is not necessary [Richtsfeld and Vincze, 2008c].

Algorithm 3 *Overview of the mesh segmentation algorithm.*

begin
1. 3D mesh generation of the complete point cloud based on a 3D Delaunay triangulation [O'Rourke, 1998, Edelsbrunner and Mücke, 1994]
2. Extraction of all objects or parts with a triangle connectivity filter [Belmonte et al., 2004]

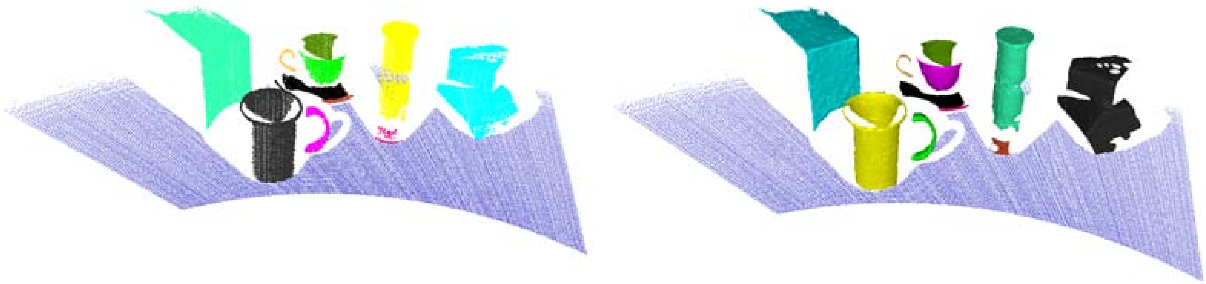
end

Figure 5.7: Segmentation result based on mesh generation. The over- and under-segmented objects are encircled red.

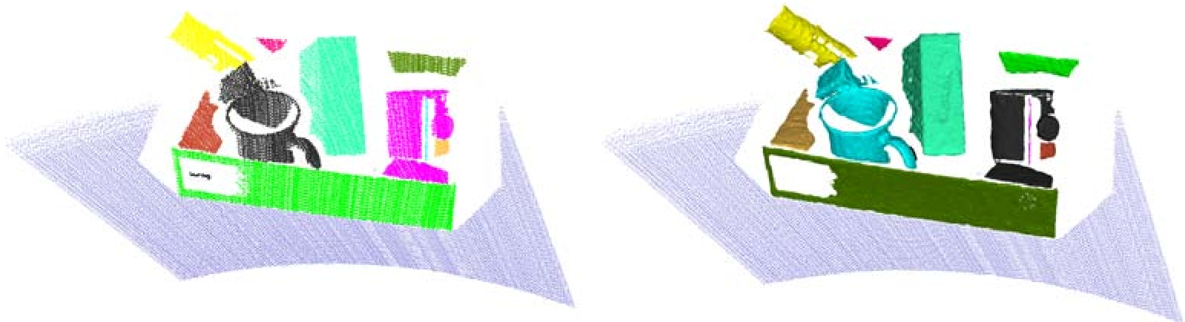
- **Time Comparison**

The results for both methods have been practical implications in the following preprocessing steps. The region-growing algorithm, however, needs more time than the mesh segmentation method. Fig. 5.8 and Fig. 5.9 compare the time performance and results of both segmentation methods by means of 5 different table scenes.

To keep the necessary computation time for the grasping point and pose detection algorithm as short as possible, the segmentation method based on a mesh generation is used.



(a) *region-growing (table scene 1,* $21,117$ *object points,* $59,397$ *table points):* $92.562s$ (b) *mesh segmentation (table scene 1,* $21,117$ *object points,* $59,397$ *table points):* $19.796s$



(c) *region-growing (table scene 2,* $20,447$ *object points,* $34,531$ *table points):* $84.906s$ (d) *mesh segmentation (table scene 2,* $20,447$ *object points,* $34,531$ *table points):* $19.360s$

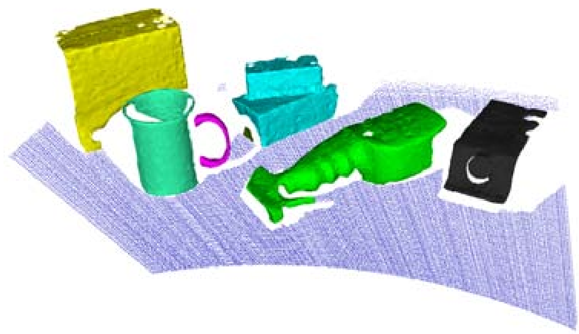
Figure 5.8: Time comparison of the region-growing algorithm and the mesh segmentation method. It is clearly visible that the segmentation based on a mesh generation is faster. However, both methods have practical implications for the following computation steps.

5.1.3 Pairwise Matching

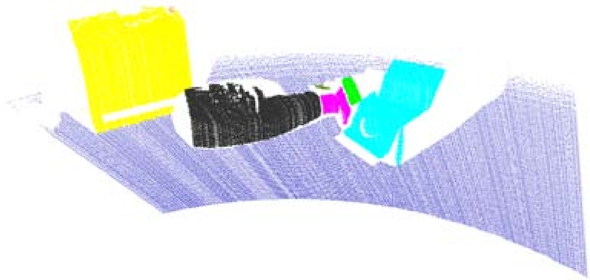
After the object segmentation step, the algorithm finds the top surfaces of all objects using a RANSAC-based plane fit and generates a 2D Delaunay triangulation. A detailed description of a RANSAC-based plane fit can be found in Section 4.2.1. For the 2D Delaunay triangulation, a method based on alpha-shapes [Edelsbrunner and Mücke, 1994] is applied. This 2D surface information is used to extract the top rim points and top feature edges of every object as illustrated in Fig. 5.10. As for the top surface detection, the algorithm uses a pre-processing step to find all vertices of the object with a vector $\vec{v}$ normal to the table plane. For example, if the x-direction of the table normal surface vector is the main direction ($n_{table[x]} > n_{table[y]}$ and $n_{table[x]} > n_{table[z]}$), the main component of the normal surface vector of the object vertices should be also $n_{object[x]}$. The normal vectors of all vertices are calculated with the faces (triangles) of the generated mesh. With the normal vectors, the top surface of the objects is guaranteed to be robustly detected, and the surface P_i with the most included points in 100 trials wins:

$$\arg\max_{1 \leq i \leq 100} P_i. \tag{5.5}$$

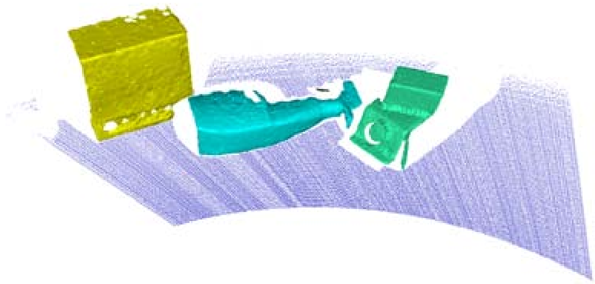
(a) *region-growing (table scene 3,* 28, 335 *object points,* 55, 151 *table points):* 168.531s



(b) *mesh segmentation (table scene 3,* 28, 335 *object points,* 55, 151 *table points):* 30.500s



(c) *region-growing (table scene 4,* 24, 395 *object points,* 61, 879 *table points):* 127.297s



(d) *mesh segmentation (table scene 4,* 24, 395 *object points,* 61, 879 *table points):* 28.672s

(e) *region-growing (table scene 5,* 16, 285 *object points,* 17, 303 *table points):* 53.156s

(f) *mesh segmentation (table scene 5,* 16, 285 *object points,* 17, 303 *table points):* 13.016s

Figure 5.9: Time comparison of the region-growing algorithm and the mesh segmentation method. It is clearly visible that the segmentation based on a mesh generation is faster. However, both methods have practical implications for the following computation steps.

The developed matching method is specifically for rotationally symmetric objects because this object class can be robustly detected and merged in a point cloud with unknown objects. To detect the top circle rim of rotationally symmetric objects, a RANSAC-based 3D circle fit [Jiang and Cheng, 1999] with a range tolerance of 2mm is used. Several tests

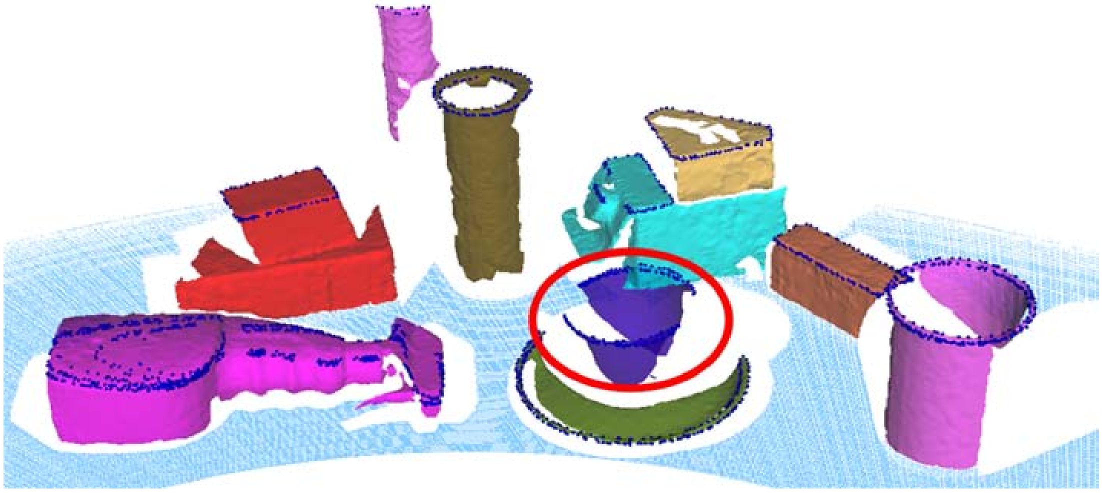


Figure 5.10: Result after the merging step. The red encircled, clipped rotationally symmetric parts are successfully merged to one object. The points marked blue show the top rim points of the objects, detected with a 2D Delaunay triangulation.

have shown that this threshold provides good results for the current laser range scanner setup. For an explicit description, the data points are defined as (p_{xi}, p_{yi}, p_{zi}) and (c_x, c_y, c_z) is the circle's centre with a radius r. The error must be smaller than the defined threshold of $2mm$:

$$||\vec{p} - \vec{c}|| - r| \leq 2. \tag{5.6}$$

This operation is repeated for every point of the top rim. The run with the maximum number n_{max} of included points wins:

$$n_{max} = |\,\{p|||\vec{p} - \vec{c}|| - r| \leq 2\}\,| \tag{5.7}$$

If more than 80% of the rim points of both parts (rotationally symmetric parts) lie on the same circle, all points of both parts are examined more closely with the fit. Thus, the distances of all points of both parts to the rotation axis are calculated, whereas section planes are arranged along the rotation axis, and only the distances of the points to the rotation axis in the respective section planes have to correspond:

$$d = (\vec{p} - \vec{c}) \times \vec{n}. \tag{5.8}$$

In Fig. 5.12 the yellow line represents the rotation axis of the object. If more than 80% of all points of both parts correspond with the neighbouring points along the rotation axis, both parts are merged to one object as illustrated in Fig. 5.10.

5.1.4 Approximation of 2.5D to 3D Objects

This step is only important to avoid potential collisions with the path planning tool because of missing model information. However, as clearly visible in Fig. 5.11, the path planning tool needs complete information to calculate a collision-free path. The closed rear side of the objects is necessary to avoid possible collisions of the robot with the objects on the table.

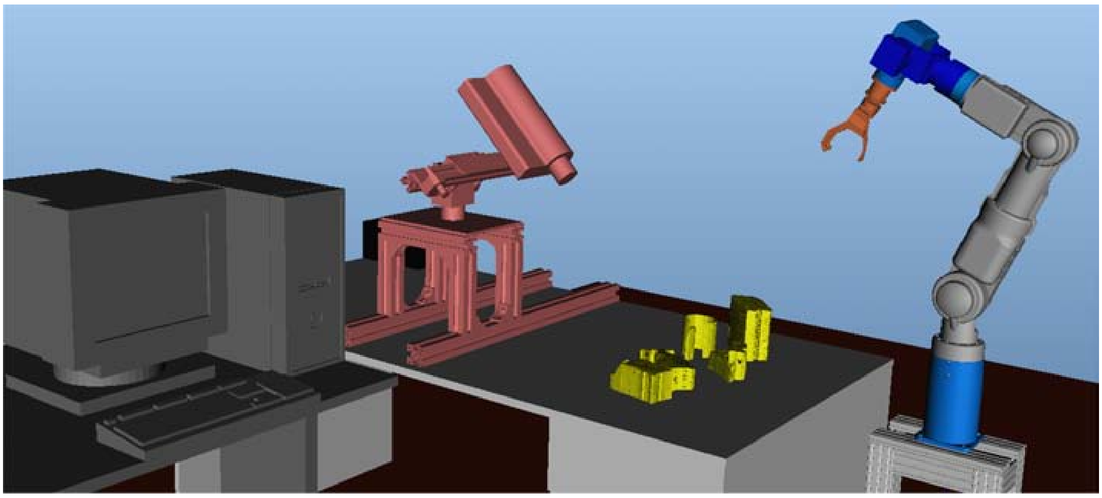


Figure 5.11: Visualisation of the experimental setup with a simulation tool, which is suitable to calculate the trajectory of the robot arm. The closed rear side of the objects on the table by an approximation of 2.5D to 3D is necessary to avoid potential collisions of the robot or gripper with the rear side of other surrounding objects.

During the matching step, the algorithm detects potential rotationally symmetric objects and merges clipped parts. With this information, the algorithm rotates points along the axis by 360° in small 5° steps, which fulfil the necessary rotation constraint. This means that the points rotated are only those, whose normal surface vector is normal to the rotation axis and which build a circle with the neighbouring points along the section plane of the rotation axis, as illustrated in Fig. 5.12. Because of this rather simple assumption, object parts such as handles or objects close to the rotationally symmetric object are not rotated. This method achieved suitable results during all the tests.

For all other arbitrary objects, the non-visible surfaces are closed with planes normal to the table plane, as illustrated in Fig. 5.12. Filling the non-visible range with vertical planes may lead to incorrect results, especially when the rear side of the objects is far from vertical. However, this step is only necessary to avoid potential collisions with the path planning tool.

5.2 Grasping Point and Pose Detection

The algorithm for grasping point detection is limited to rotationally symmetric objects and the grasping poses or, to be more precise power grasps, are calculated for arbitrary objects. After the segmentation step, the algorithm finds out if the object is open or closed with a sphere fit into the top surface. The sphere is fitted into the centre of gravity of the top surface and if there is no point of the object within the fitted sphere, the given

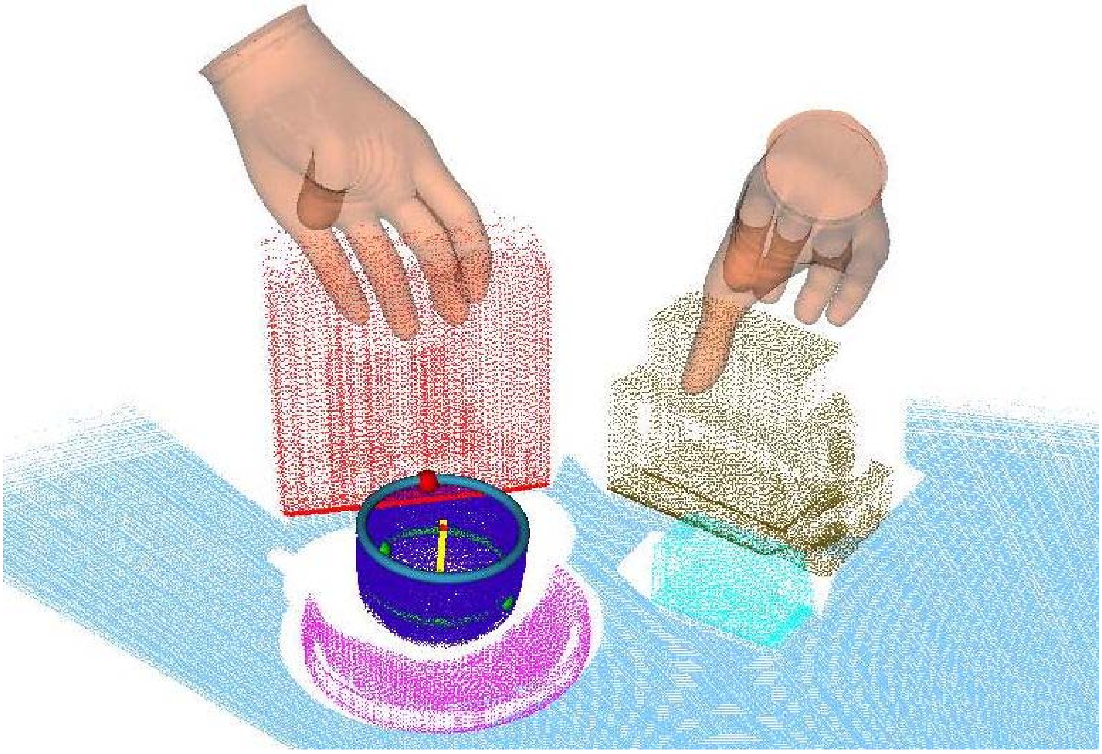


Figure 5.12: Detected grasping points and power grasps. The green points illustrate the computed grasping points for rotationally symmetric objects to perform a cylindrical grasp. The red point shows an alternative tip grasp along the top rim, with one grasping point being enough for an open object. The illustrated hand poses show possible power grasps for the remaining graspable objects.

object is open. The radius r of the sphere, depends on the distance p_d between the centre of gravity and the nearest top rim point:

$$r = p_d \cdot \frac{2}{3}. \tag{5.9}$$

Then the grasping points of all cylindrical objects can be calculated. For every rotationally symmetric object two grasping points are calculated in the middle of the object for a cylindrical grasp (points coloured green as illustrated in Fig. 5.13, object no. 7). If the path planner is not able to detect a collision-free path, the algorithm calculates alternative grasping points along the top rim of the object near the strongest curvature, as illustrated in Fig. 5.13, (object no. 7, red point). The algorithm finds out the strongest curvature along the top rim with a Gaussian curvature filter [Porteous, 1994]. If it is an open object, one grasping point is enough to realise a stable tip grasp near the top rim. The grasping points are calculated in such a way that they are next to the robot arm, mounted on the opposite side of the laser range scanner.

To successfully grasp an object, it is not always sufficient to locally find the best grasping pose. The algorithm should calculate a stable power grasp to realise a good

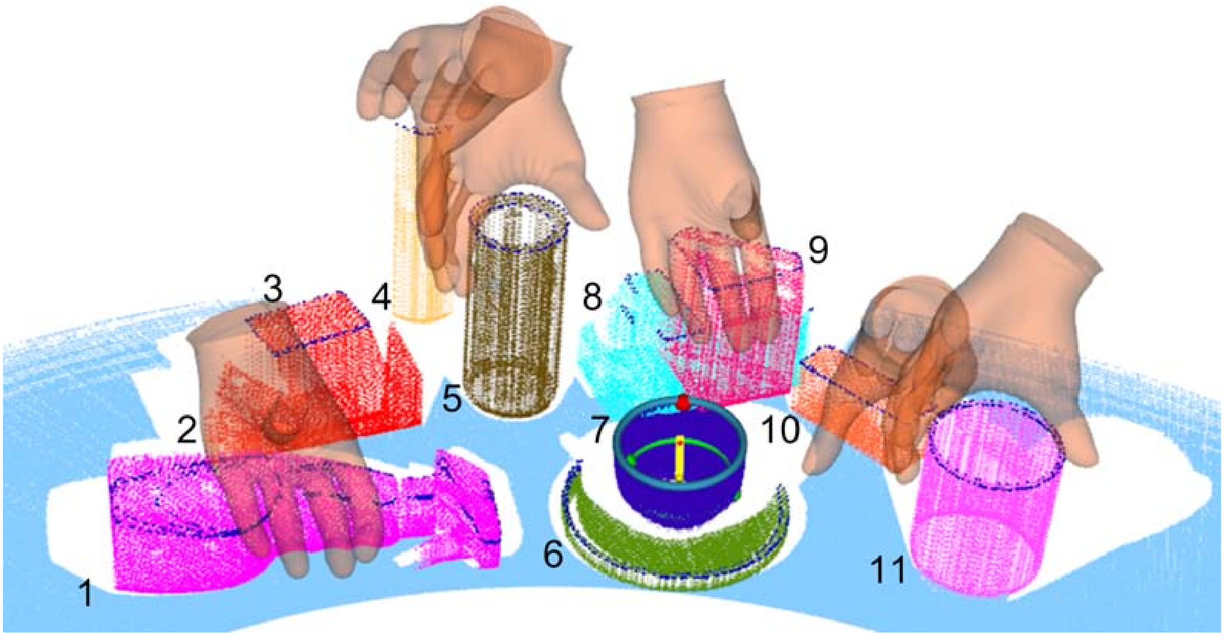


Figure 5.13: Detected grasping points and power grasps. The objects are numbered from left to right. It is clearly visible that the overlapped objects no. 2 and 3 are detected as one object. The segmented rotationally parts of object no. 7 are successfully merged to one object. Object no. 5 was not detected as a rotationally symmetric object, because only 76% *of the top surface points lie on the top circle rim. The green points on object no. 7 illustrate the computed grasping points for rotationally symmetric objects to realise a cylindrical grasp. The red point shows an alternative tip grasp along the top rim. The illustrated hand poses show possible power grasps for the remaining graspable objects.*

grasp without collision as fast as possible. In general, conventional multi-dimensional "brut force" search methods are not practical enough to solve this problem.

If there is only a limited field of view from above, which results in a 2.5D point cloud and the need to calculate collision-free power grasps, only grasps from above will be calculated. In the same way, infants prefer to grasp an object by visualising the top surface of the object to avoid potential collisions as discussed in Section 2.7.1. It is interesting to observe that people generally keep their hands outside a region that surrounds the obstacle at a minimum distance. There is a higher probability of collision with other objects if you grasp the object from the left or right-hand side. The robot might also collide with the table. In addition, the left and right hand side of the grasping object is often not sufficiently evident. Best results to successfully grasp the object are achieved if you grasp the object from above, near the centre of the object. This experience is reflected by a study of Tristian Nakagawa et al. [Nakagawa et al., 2008][2]. [Nakagawa et al., 2008] analysed the fixation duration (called heat maps) of 6 participants during the first 500ms with a head-

[2] I would like to thank Prof. Heiner Deubel from the Ludwig-Maximilians-University department of psychology for providing Fig. 5.14

mounted video based eye-tracker. First the participants registered the position of the object and then they decided where they would grasp the object from above.

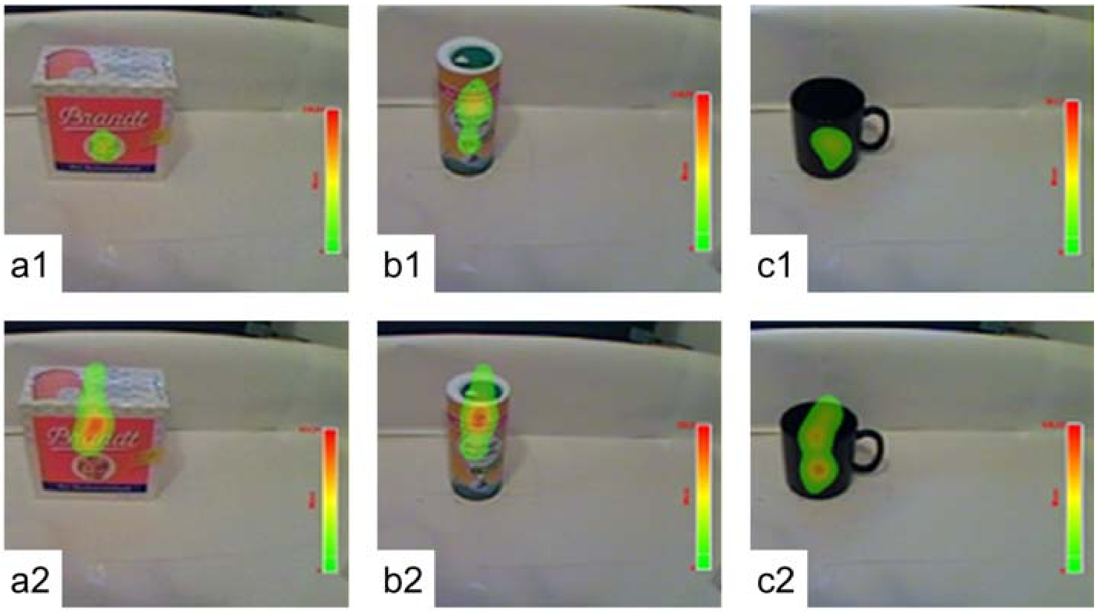
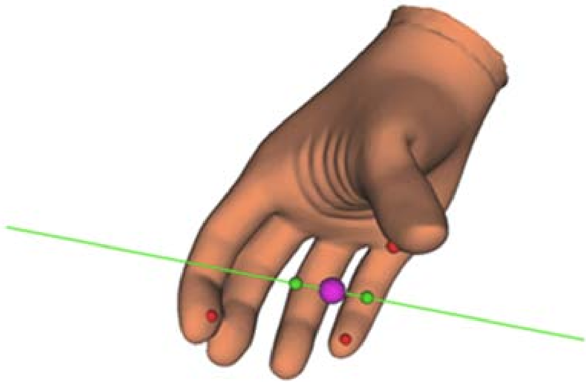


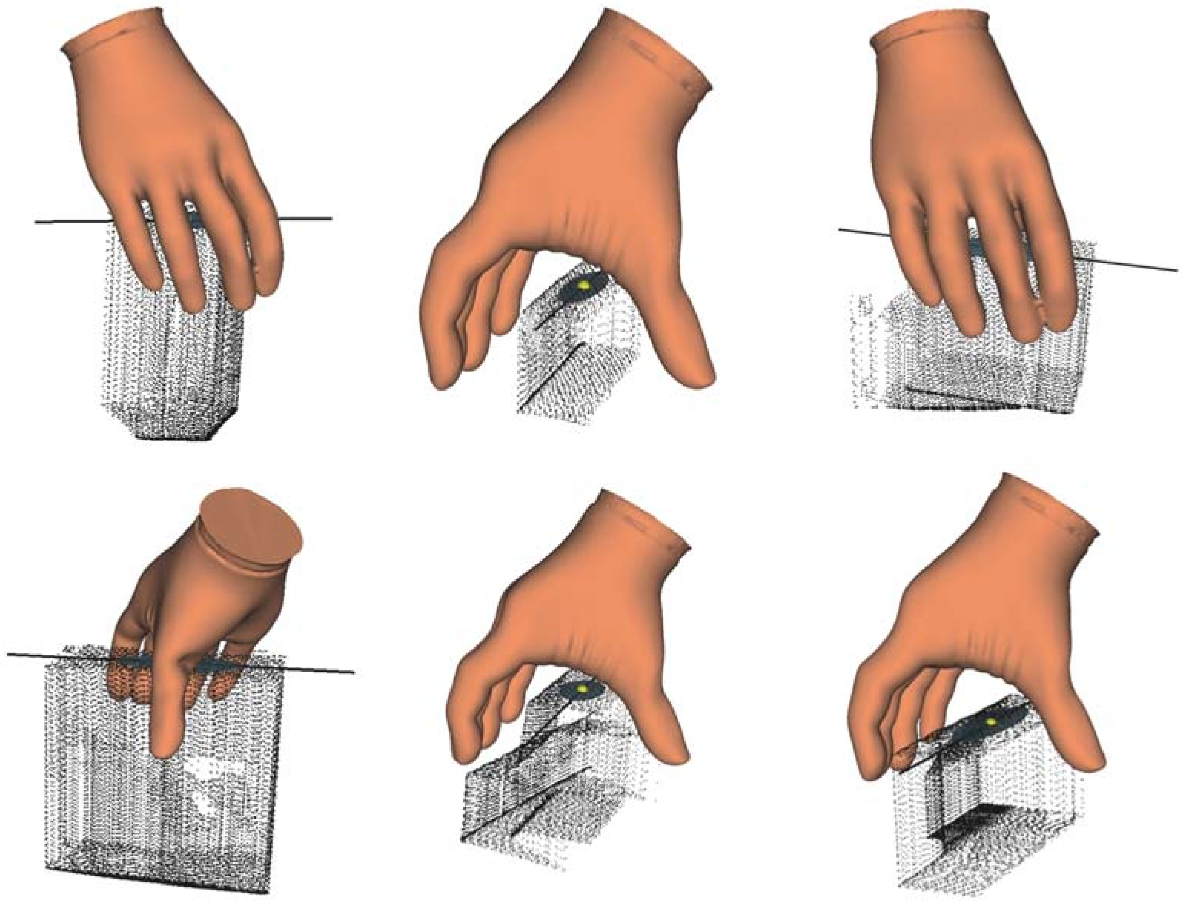
Figure 5.14: Heat maps of one participant based on fixation duration during the first $500ms$ *of each trial. The first row shows the centre of the detected object and the second row shows the central point for a grasp from above [Nakagawa et al., 2008].*

Fig. 5.14 shows the heat maps of one participant based on fixation duration during the first $500ms$ of each trial. The first row shows the centre of the detected object and the second row shows the central point for a grasp from above. Based on this expertise the following method was developed. It is clearly visible that the central point is along the top surface near the centre of the object, which reflects the position of the thumb or the fingers of the power grasp calculated with the presented method.

In the beginning, the internal centre and the principal axis of the top surface are calculated with a transformation that fits and transforms a sphere inside (see the elliptical surfaces coloured blue in Fig. 5.15(b)). If it is a Gaussian distributed point cloud and the transformation is applied to a unit sphere, the transformed sphere illustrates the covariance structure of the point cloud. After the transformation, this sphere has an elliptical form in alignment with the top surface points with the principal axis being calculated. The algorithm transforms the rotation axis of the gripper (defined by the fingertip of the thumb, the index, and the last finger as illustrated in Fig. 5.15(a)) along the principal axis of the top surface of the object and the TCP (centre of the fingertips) of the gripper are translated to the centre of the top surface c_{top}, whereby $TCP = c_{top}$ results. The hand is rotated in such a way that the normal vector of the hand aligns in reverse direction to the normal vector of the top surface. Afterwards the hand is shifted along the normal vector down to a possible collision with the grasping object. Both the possible power grasps from the left and right are tested, and the one which encloses the object most tightly wins, guaranteeing a collision-free and stable power grasp as illustrated in Fig. 5.15(b).



(a)



(b)

Figure 5.15: Detection of the optimal power grasp. **(a)** *The rotation axis of the hand must be aligned with the principal axis of the top surface.* **(b)** *The hand is transformed and rotated along the principal axis of the top surface. After this step the algorithm checks potential collisions with all surrounding objects.*

The calculated power grasp is checked to identify a potential collision with the remaining objects and the table. Thus the algorithm determines if it is possible to grasp the object depending on the other objects, as illustrated in Fig. 5.13. An obb-tree is used [Gottschalk et al., 1996] for collision detection.

5.3 Experimental Evaluation

The system presented in Section 4.1 is again used in this work. Fig. 4.1 showed that the system is equipped with an additional camera to realise a 3D dense stereo reconstruction. The camera captures two images with two defined angularities at $-4°$ (left image) and $0°$ (right image) by a pitch of $-37°$ of the pan/tilt unit to overlook the tabletop scene as illustrated in Fig. 5.18.

The tool centre point (TCP) to grasp arbitrary objects is defined by the centre of gravity of the tip of the thumb, the index, and the little finger. The TCP to grasp rotationally symmetric objects is defined by the centre of gravity of the thumb and the index finger. For this reason two grasping types: the cylindrical grasp and the tip grasp along the top rim have been implemented.

To provide a general evaluation of the presented method, different objects and categories have been defined. All the 12 defined objects for the evaluation are presented in Fig. 5.16. Different situations like several objects in different poses, touching objects, and scattered objects in a box as well as a pile of objects have been defined as grasp categories. Additionally, tests based on dense stereo data have been carried out to illustrate the robustness of the presented grasping algorithm.

Figure 5.16: 12 different objects were selected to evaluate the grasping point and pose detection algorithm, from left: ***1.*** *Salt Shaker (cylinder),* ***2.*** *Sweets (cylinder),* ***3.*** *Spread (cylinder),* ***4.*** *Sweets (half cylinder),* ***5.*** *Melba Toast (cuboid),* ***6.*** *Dextrose (cuboid),* ***7.*** *Salt Shaker (cuboid),* ***8.*** *Sweets (pentagon object),* ***9.*** *Coffee Cup,* ***10.*** *Shower Bath Gel (paraboloid),* ***11.*** *Cleaner (paraboloid),* ***12.*** *Sweets (prism).*

The detected grasping points and power grasps are tested directly on the objects with the AMTEC 7DOF robot arm and the Otto Bock hand as the gripper. The object segmentation, merging, grasping point, grasping pose, and collision detection is performed by a PC with 2.0GHz Intel Core 2 Duo processor and takes a duration of $66s$ depending on the number of objects on the table, see Tab. 5.1. The algorithm is implemented in C++ using the Visualization Tool Kit (VTK).

calculation steps	time [s]
plane fit	1.4
mesh generation	35
mesh segmentation	0.7
top surface detection	0.9
merging rotationally symmetric objects	2
approximation of 3D objects	6
grasping point detection	3.5
grasping pose detection	6.5
collision detection	10
$\sum$	66

Table 5.1: Duration of every calculation step.

5.3.1 Results Based on Laser Range Data

In the following evaluation the grasping rate, which reflects the grasping success, was analysed depending on the detection rate for the defined grasp categories. A successful robust grasp is defined as a grasp with delivering the grasped object to a determined position. In addition, the evaluation was arranged by a third person[3] in order to make a fair analysis possible.

Tab. 5.2 shows the detection and grasping rate of all 12 defined objects to evaluate the presented method. Every object was scanned 10 times in different poses and the algorithm tried to detect features to realise a stable grasp. Tab. 5.2 shows a grasping rate of 80.65%, which depends on the detection rate of 95% over all 12 several unknown objects.

object no.	object	detection rate [%]	grasping rate [%]
1	Salt Shaker (cylinder)	100	100
2	Sweets (cylinder)	80	87.50
3	Spread (cylinder)	90	77.77
4	Sweets (half cylinder)	80	62.50
5	Melba Toast (cuboid)	90	100
6	Dextrose (cuboid)	100	100
7	Salt Shaker (cuboid)	100	100
8	Sweets (pentagon object)	100	100
9	Coffee Cup	100	60
10	Shower Bath (paraboloid)	100	50
11	Cleaner (paraboloid)	100	40
12	Sweets (prism)	100	90
Overall		95.00	80.65

Table 5.2: Laser data: Detection and grasping rate of several unknown objects. Each object was tested 10 *times in different poses.*

This result illustrates the high reliability of the detected features to calculate possible grasping points and poses. The grasping rate proves that the developed method calculates stable grasps for several objects on a table. The grasping success of object no. 4 depends on the orientation of the object on the table. If the half cylinder lies with the cylinder side on the tabletop and if the top surface is not completely visible, due to absorption or reflection, the calculated grasping pose fails. In contrast object no. 11 is very heavy and has a smooth surface, which results in a stick-slip effect.

Several objects placed on a table hardly ever appear in the real world, hence touching objects are also evaluated as a separate categories. Tab. 5.3 illustrates the result of the evaluation in this case. The object grasped was in each case the highest object out of the reminder. No one ever grasps the bottom object in a carrier bag because of the collision of the hand with objects placed on top. For this category a detection rate of 87.17% was achieved with a resulting grasping rate of 77.57%.

[3] I would like to thank Dipl.-Ing.(FH) Thomas Vrca for evaluating the grasping success of the presented algorithm.

object no.	object	detection rate [%]	grasping rate [%]
1	Salt Shaker (cylinder)	100	40
2	Sweets (cylinder)	80	100
3	Spread (cylinder)	90	66.67
4	Sweets (half cylinder)	90	66.67
5	Melba Toast (cuboid)	100	100
6	Dextrose (cuboid)	90	100
7	Salt Shaker (cuboid)	100	100
8	Sweets (pentagon object)	100	90
9	Coffee Cup	80	87.50
10	Shower Bath (paraboloid)	100	30
11	Cleaner (paraboloid)	80	50
12	Sweets (prism)	60	100
Overall		89.17	77.57

Table 5.3: Laser data: Detection and grasping rate of touching unknown objects. The grasped object was the topmost object in each case. Each table top scene for each object was tested 10 *times in different poses in combination with* 2 *to* 5 *other objects.*

Figure 5.17: Laser data: Detected grasping points and poses of unknown objects in a box. There is a lower detection rate, because of possible collisions of the hand with the box.

Then a very complex case of objects in different poses in a box was evaluated. Fig. 5.17 illustrates that there is a lower detection rate because of possible collisions of the gripper with the box. Tab. 5.4 shows the detection of 77.50% of the objects and grasping rate of 82.70% overall. However, with a grasping rate of 82.70%, the developed method has a high reliability to detect necessary features to successfully grasp the objects.

Finally, a pile of touching objects was evaluated. This happens if the purchase is simply emptied on the kitchen table at home. With a detection rate of 82.50% and a grasping rate

object no.	object	detection rate [%]	grasping rate [%]
1	Salt Shaker (cylinder)	80	87.50
2	Sweets (cylinder)	30	100
3	Spread (cylinder)	60	100
4	Sweets (half cylinder)	90	66.67
5	Melba Toast (cuboid)	80	100
6	Dextrose (cuboid)	70	100
7	Salt Shaker (cuboid)	90	100
8	Sweets (pentagon object)	80	87.50
9	Coffee Cup	100	60
10	Shower Bath (paraboloid)	80	25
11	Cleaner (paraboloid)	100	80
12	Sweets (prism)	70	85.71
Overall		77.50	82.70

Table 5.4: Laser data: Detection and grasping rate of touching unknown objects in a box. The grasped object was the topmost object in each case. Each scene for each object was tested 10 *times in different poses and combinations with all other objects.*

of 79.40% presented in Tab. 5.5, the presented method shows a high level of robustness.

object no.	object	detection rate [%]	grasping rate [%]
1	Salt Shaker (cylinder)	100	50
2	Sweets (cylinder)	30	66.67
3	Spread (cylinder)	90	77.78
4	Sweets (half cylinder)	60	50
5	Melba Toast (cuboid)	80	100
6	Dextrose (cuboid)	80	75
7	Salt Shaker (cuboid)	100	100
8	Sweets (pentagon object)	100	100
9	Coffee Cup	80	100
10	Shower Bath (paraboloid)	90	55.56
11	Cleaner (paraboloid)	90	77.78
12	Sweets (prism)	90	100
Overall		82.50	79.40

Table 5.5: Laser data: Detection and grasping rate of a pile of touching unknown objects on the table. The grasped object was the topmost object in each case. Each scene for each object was tested 10 *times in different poses and combinations with all other objects.*

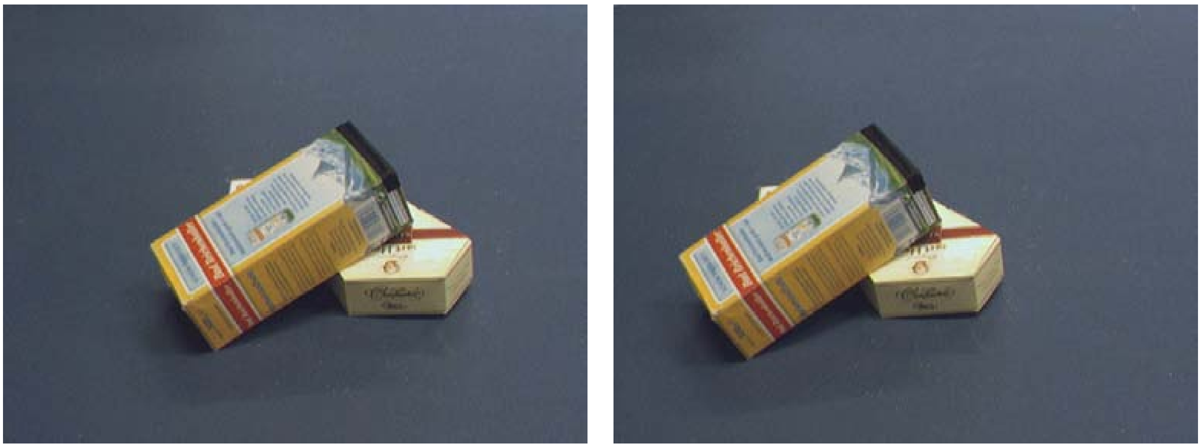
5.3.2 Results Based on Dense Stereo Data

The last case was evaluated again based on dense stereo data, recorded with the stereo system presented in Section 5.3. With the dense stereo data only a potential power grasp was calculated because of more noise and outliers in these input data.

The accuracy of a stereo reconstruction depends on the accuracy achieved in four important steps of the process: camera calibration, rectification of stereo images, disparity calculation and triangulation of 3D points. Poor performance in untextured regions is a common problem of correlation methods based on area block matching.

Below a short overview of a 3D stereo reconstruction is given[4]:

"Stereo reconstruction is using two images of an object to compute its 3D coordinates in the real world. The camera is observing the object from a different point of view and the images appear shifted to a certain extent. The amount of the object shift is called disparity, which is the key information to determine the physical position of an object in the real world.



(a) (b)

Figure 5.18: **(a)** *Left and* **(b)** *right image of the camera at two defined angularities of the pan/tilt unit to overlook the tabletop scene.*

Camera parameters are required for the stereo reconstruction, which are obtained by a camera calibration. The aim in this process is to find parameters belonging to two categories: internal (consisting of focal length, principal point coordinates and distortion parameters data) and external (rotation matrix $\mathbf{R}$ *and translation vector* $\mathbf{T}$ *with respect to the reference coordinate frame). While internal parameters describe properties of the image projection through the lens, external parameters give the information about the relative position of the camera coordinate frame regarding a fixed reference coordinate frame. For*

[4]I would like to thank Dipl.-Ing. Dzemaludin Efendic. His master thesis [Efendic, 2008] presents the basis for the following short overview in 3D reconstruction.

this task the "Camera Calibration Toolbox for Matlab[5]" was used.

Inaccuracy during the lens production process lead to deformations of the image projection, commonly appearing as barrel and pincushion distortion. The image distortions are modelled as radial and tangential distortions. Radial distortion causes bending of a straight line on the image, with the bending centre in the principal point. Tangential distortion is a pixel shift in the tangential direction on a circle centred in the principal point.

The distortion of a particular image pixel is related to the second and fourth power of its distance from the principal point[6], quantified by two radial (k_1 and k_2) and two tangential (p_1 and p_2) distortion parameters. If x and y describe the physical coordinates of an undistorted image pixel, the physical coordinates x_d and y_d of the corresponding pixel in the initial, distorted image are given by:

$$x_d = x + x \cdot [k_1 \cdot r^2 + k_2 \cdot r^4] + [2 \cdot p_1 \cdot x \cdot y + p_2 \cdot (r^2 + 2 \cdot x^2)], \tag{5.10}$$

$$y_d = y + y \cdot [k_1 \cdot r^2 + k_2 \cdot r^4] + [2 \cdot p_1 \cdot x \cdot y + p_2 \cdot (r^2 + 2 \cdot y^2)] \tag{5.11}$$

with

$$r^2 = x^2 + y^2. \tag{5.12}$$

Stereo parameters can be computed by using the extrinsic parameters of the stereo cameras [Trucco and Verri, 1998]. Since they are related to the same reference coordinate frame, the rotation matrices of the left $\mathbf{R_l}$ and right $\mathbf{R_r}$ camera produce immediately:

$$\mathbf{R} = \mathbf{R_r} \cdot \mathbf{R_l}^T. \tag{5.13}$$

Similarly, the translation vector $\mathbf{T}$ follows from the translation vector of the left $\mathbf{T_l}$ and right $\mathbf{T_r}$ camera as:

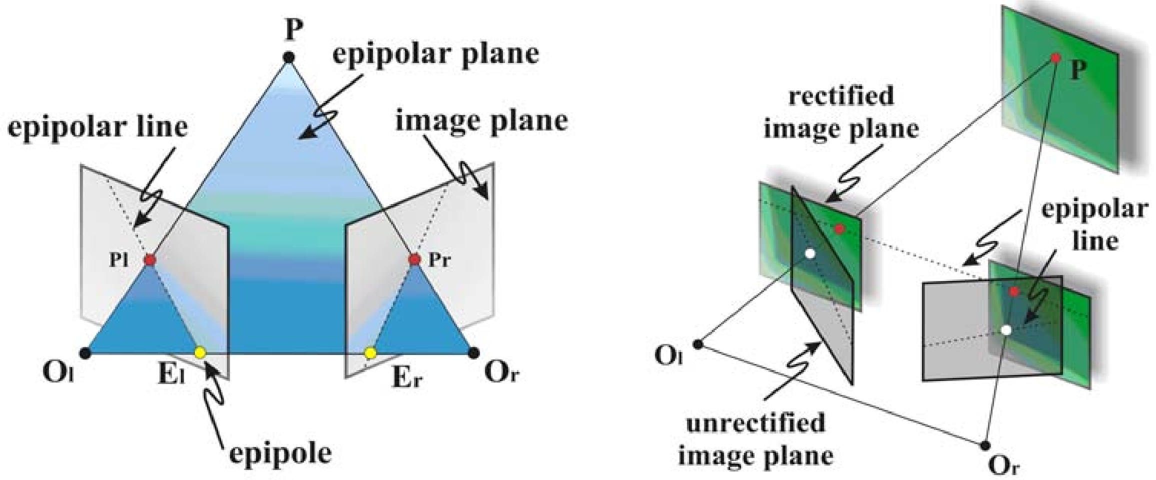
$$\mathbf{T} = \mathbf{T_l} - (\mathbf{R_l} \cdot \mathbf{R_r}^T) \cdot \mathbf{T_r}. \tag{5.14}$$

Correction of the image distortion, requires to start with a pixel P_u of the undistorted image and to "jump back" to a certain point P_d on the distorted image and the rectification of the stereo images is closely tied to the epipolar geometry of the stereo setup [Trucco and Verri, 1998] as illustrated in Fig. 5.19. Fig. 5.20 shows the result of the correction of distortion and rectification of the images.

A point P is projected on the image planes along the lines of sight connecting it with the camera's focal points O_l and O_r. Together with the direction linking O_l and O_r, the lines of sight define a triangle surface representing a part of the epipolar plane. The intersections of the epipolar plane with the image planes are called epipolar lines. The adjective "epipolar" originates from the term "epipole" used to name points of the image planes (E_l and E_r) that intersect with the connecting line between O_l and O_r. All epipolar lines of an image are passing through its epipole.

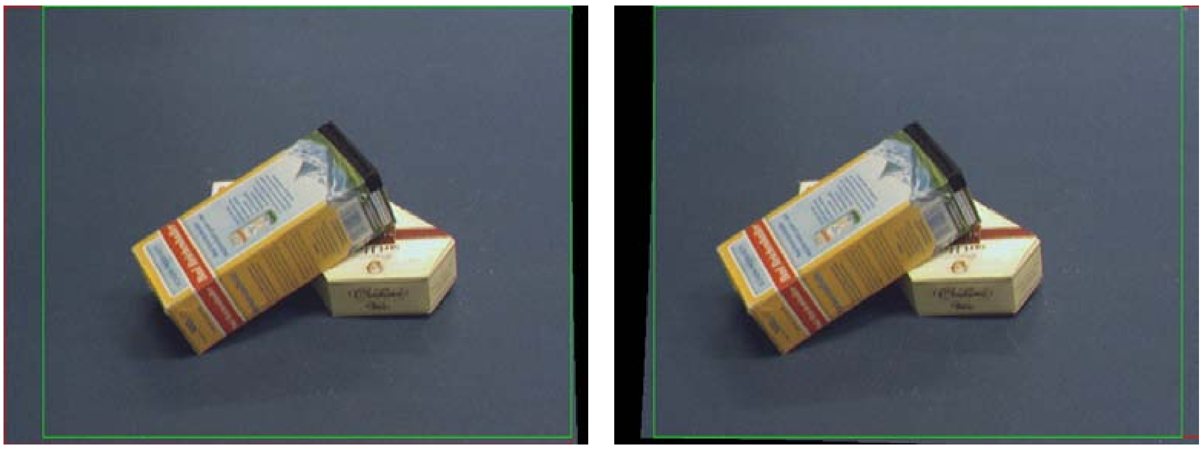
[5]Camera Calibration Toolbox for Matlab, Jean-Yves Bouguet, http://www.vision.caltech.edu/bouguetj/index.html

[6]Open Source Computer Vision Library, Reference Manual, Issued in USA, Order Number: 123456-001.



(a) (b)

Figure 5.19: Principle of epipolar geometry and rectification: **(a)** *The intersections of the epipolar plane with the image planes are called epipolar lines.* **(b)** *The aim is to produce images with horizontal and collinear epipolar lines.*



(a) (b)

Figure 5.20: Result of the correction of the image distortion and rectification on the **(a)** *left and* **(b)** *right image.*

The point's projections P_l and P_r must lie on the corresponding epipolar lines. This is the motivation for performing rectification of the stereo images, with the aim to produce images with horizontal and collinear epipolar lines [Trucco and Verri, 1998]. It means, that the corresponding epipolar lines (thus the corresponding projections P_l and P_r of the point P as well) of the rectified images will be placed along the same row of the image pixel matrix.

Two pixels of the left and right stereo image showing the same object point are called corresponding pixels. Corresponding pixels of the rectified stereo images with horizontal *epipolar lines vary only in their horizontal coordinate values. The difference d of the corresponding pixel coordinates x_r (for the right image) and x_l (for the left image):*

$$d = x_r - x_l. \tag{5.15}$$

is referred to as disparity.

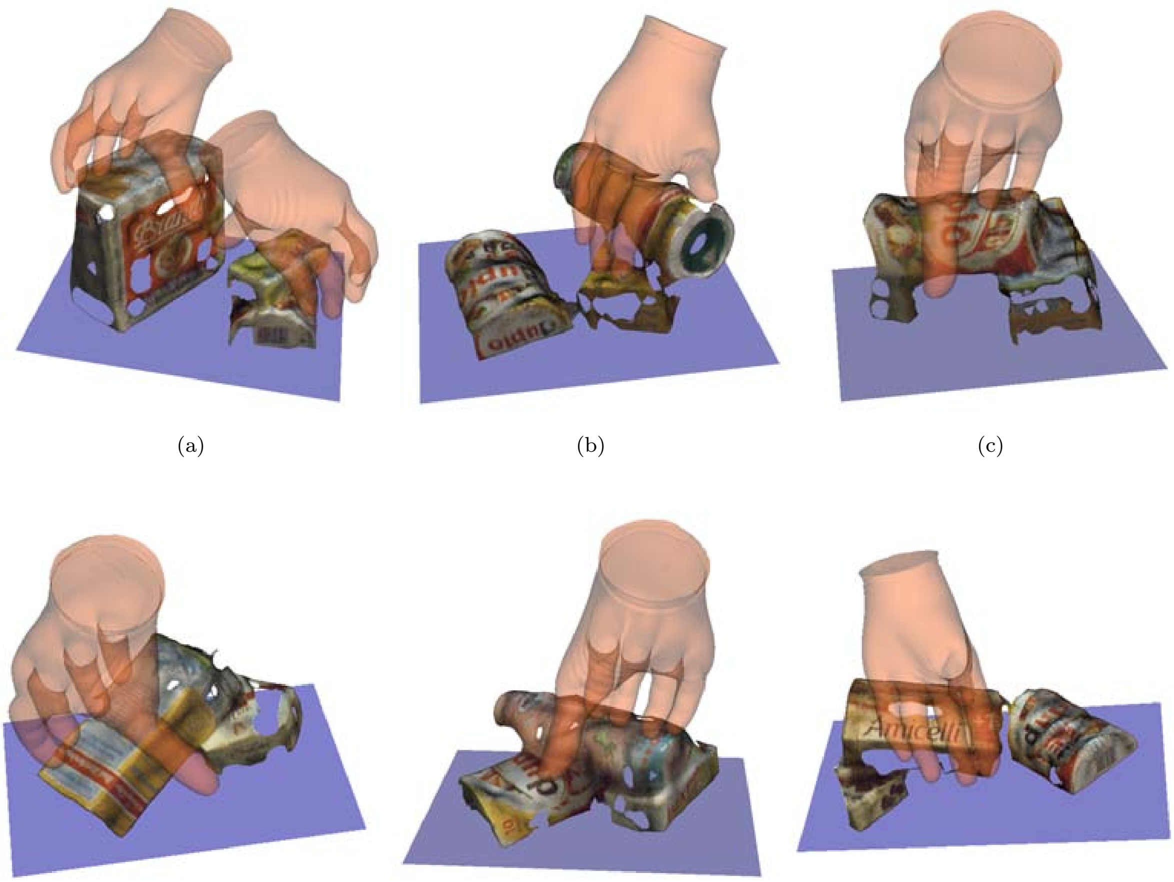
Computing disparities for all image pixels is the key step in reconstructing object's 3D geometry. All the information needed for making the object's 3D model is contained in its disparity values. There is a direct link between the position of an object with respect to the stereo cameras and its disparity value, which allows to compute the 3D coordinates of objects by a triangulation. Results of the disparity search are stored in a lookup-table (LUT) called disparity map. In a disparity map every image pixel receives a disparity value assigned. There exist different methods to calculate the disparity map [Scharstein and Szeliski, 2002, Trucco and Verri, 1998].

The used method for finding image disparities is area block matching with a "sliding window" [Mühlmann et al., 2002]. This method relies on using a local cost function to evaluate the similarity level of potentially matching pixels. In practice, pixel colour intensities from a window-shaped area around the correspondence candidates are compared by computing a correlation function, in order to assess them. Commonly used correlation functions are sum of absolute differences (SAD) and sum of squared differences (SSD)." Fig. 5.21 shows the result of the 3D dense stereo reconstruction and Fig. 5.22(d) the detected grasping pose.

Figure 5.21: Result of the 3D stereo reconstruction.

To create a dense stereo calibration onto the laser range coordinate system as precisely as possible the laser range scanner was used to scan the same chessboard as used for the camera calibration. At the obtained point cloud, a marker was set as reference point to the camera coordinate system. The blue table surface was filtered directly in the images and all pixels with the dominant colour component blue were removed before the 3D reconstruction started.

In Fig. 5.22, it is clearly visible that the resulting mesh has a higher inaccuracy due to noise and outliers than the segmented mesh in Fig. 5.17 based on a range scanner system. A detailed analysis and comparative study is given in [Efendic, 2008]. Fig. 5.21 shows some example results of the presented method to calculate a power grasp.



(a) (b) (c)

(d) (e) (f)

Figure 5.22: Detected grasping poses of unknown objects based on dense stereo data.

Tab. 5.6 shows that the quality of the input data has a strong influence on the detection rate followed by the grasping rate. But the presented method reaches a practical grasping rate of 66.62%, even in this difficult and complex case.

object no.	object	detection rate [%]	grasping rate [%]
1	Salt Shaker (cylinder)	100	70
2	Sweets (cylinder)	60	66.67
3	Spread (cylinder)	70	71.43
4	Sweets (half cylinder)	40	75
5	Melba Toast (cuboid)	80	100
6	Dextrose (cuboid)	40	75
7	Salt Shaker (cuboid)	90	88.89
8	Sweets (pentagon object)	90	88.89
9	Coffee Cup	40	50
10	Shower Bath (paraboloid)	100	60
11	Cleaner (paraboloid)	70	28.57
12	Sweets (prism)	50	100
Overall		69.17	66.62

Table 5.6: Dense stereo data: Detection and grasping rate of a pile of touching unknown objects on the table. The grasped object was the topmost object in each case. Each scene for each object was tested 5 *times in different poses and combinations with* 2 *to* 5 *other objects.*

5.4 Discussion

The presented method for automatic grasping of unknown objects with a 3D model of the used gripper demonstrates a high reliability. In addition, the presented method was also tested on dense stereo data. The approach for object grasping is well suited to use in related applications under difficult conditions and can be applied to a reasonable set of objects.

The algorithm calculates potential grasping points and poses based on the top surface of the objects. The method was developed for arbitrary objects in different poses, on top of each other or side by side with a special focus on rotationally symmetric objects. If objects cannot be separated because they are stacked on top of each other, they are considered as one object. If the algorithm detects clipped rotationally symmetric parts, these fragments are merged because this object class can be robustly identified using the symmetry assumption, allowing a cylindrical grasp as well as a tip grasp along the top rim. If objects cannot be separated because they are stacked on top of each other they are considered as one object. A grasp will disambiguate the situation. For this objects, the algorithm calculates a power grasp based on the top surface.

With the dense stereo data only a potential power grasp was calculated, due to a lot of noise and outliers of these data.

The evaluation takes 12 objects in different daily life situations as categories:

- several objects in different poses,
- 3 - 5 touching objects,
- objects in a box,
- and a pile of touching objects (same as scattered objects without a box).

After testing each object in 10 different poses for different categories, the algorithm shows the following results. The algorithm reaches for laser range data a grasping rate of 80.65% for several objects, 77.57% for touching objects, 82.70% for objects in a box, and 79.40% for a pile of objects.

Based on dense stereo data a pile of touching objects was tested and the algorithm reached a grasping rate of 66.62%.

Problems remain when important parts of shiny objects are not visible to the laser range scanner, hence the presented algorithm is neither able to calculate correct grasping points nor the pose of the object. Furthermore, the quality of the point cloud may not be good enough to guarantee a successful grasp. The success of the grasping algorithm depends on ambient light, object surface properties, laser-beam reflectance, and absorption of the objects.

This work points out the general feasibility to realise stable grasps from only one single view. That means that occluded objects cannot be analysed or grasped. It was assumed that the top surface of all objects on the table are clearly visible.

Chapter 6

Object Part Segmentation

"Clouds are not spheres,
mountains are not cones,
coastlines are not circles,
and bark is not smooth,
nor does lightning travel
in a straight line."
– *Benoit Mandelbrot*

The human vision segments an object into different parts and analyses their spatial and functional relationships [Shipley and Kellman, 2001]. A part-based description allows one to detect the different parts and functional properties of an object.

This chapter introduces a 3D segmentation algorithm, based on spherical mirroring, which works directly on point clouds and meshes to address the problem of partitioning a 3D object into useful sub-parts.

Segmentation is a crucial task of partitioning range images into useful parts. It is obvious from the results that no perfect segmentation algorithm exists. Each algorithm has its own benefits depending on the task. In general, a good segmentation result is achieved if regions can be approximated by a given set of surface functions as described in 2.4.

An ideal shape descriptor finds the main features of an object and segments it into useful parts, which can be used for automatic processes such as fitting, registration, feature extraction [Gumhold et al., 2001] or comparison of shapes. Dividing a point cloud into simpler sub-parts has also several benefits for modelling [Funkhouser et al., 2004], robotics [Huebner et al., 2008] or collision detection [Li et al., 2001]. The object should be segmented into parts that correspond to relevant features, e.g. handles or the core part of the object.

In the last few decades, many different algorithms have been proposed in this growing field, e.g. feature point and core extraction [Katz et al., 2005], Hierarchical Fitting Primitives (HFP) [Attene et al., 2006a], spectral methods [Zhang et al., 2007], K-Means [Shlafman et al., 2002], random walks [Lai et al., 2008], and the shape diameter function (SDF) [Shapira et al., 2008]. Most of them can only segment a watertight mesh and not a point cloud. In robotics especially, it is not always possible to generate a watertight mesh.

The motivation was to develop an algorithm which segments a point cloud into useful sub-parts without any model assumptions and fixed thresholds. This is significant because the input data can be deducted from different systems. So the algorithm also segments a mesh and is expandable to solve more complex segmentation tasks. Additionally, the algorithm is invariant under rotation, translation, and scaling.

An experimental evaluation of a number of complex objects demonstrates the robustness and the efficiency of the proposed algorithm and the results prove that it compares well with a number of state-of-the-art 3D object segmentation algorithms. Additionally, the metrics "cut discrepancy" and the "rand index" of the proposed method have been evaluated with the help of 10 different object classes [Chen et al., 2009].

This comparison study has been done with the K-Means clustering method, the shape diameter function (SDF) and especially with the feature point and core extraction method.

The developed algorithm is based on spherical mirroring as the core extraction method. This allows different changes of the main influence factors, like the position of the centre for spherical mirroring or the change of the topology of the object. Thus such a method is expandable to solve more complex segmentation tasks. To realise a fair comparison, the study is based on the original meshes because the mesh segmentation results of all 3 other algorithms are freely available [Chen et al., 2009] and are also achieved without changing the topology of the object.

The main difference to the existing core extraction algorithm is the calculation of an additional convex hull to get a hole-free core part, used to segment a point cloud. In [Katz et al., 2005], the algorithm segments the shape of the object hierarchically and stops automatically when the current segment $\mathcal{S}^i$ has no feature points or when the fraction of vertices contained in the convex hull is above a fixed threshold. In comparison, the proposed algorithm works without any fixed thresholds or parameters.

6.1 Description Overview of the Method

This section presents the method for 3D object part segmentation. Fig. 6.1[1,2] outlines the segmentation approach in detail. It illustrates the main processing steps taking the 3D point cloud of the object as input data (see Fig. 6.1(a)) to the segmented parts of the object based on spherical mirroring, as illustrated in Fig. 6.1(h).

Firstly, the algorithm calculates the internal centre and the radius of the bounding sphere by computing the smallest enclosing sphere of points [Gärtner, 1999], see Fig. 6.1(b). Then all points are spherically mirrored (radially reflected) outside in the reverse direction to the centre. Fig. 6.1(c) illustrates that all points inside the original point cloud are farthest away after this step. Fig. 6.1(d) shows the convex hull [O'Rourke, 1998] (coloured yellow) calculated with the reflected point cloud to detect all points farthest away inside the original point cloud. This step allows the automatic detection of the core

[1] All images are best viewed in colour. In each case the core part is coloured red.

[2] Freely available at the Aim@Shape repository, http://shapes.aim-at-shape.net/index.php

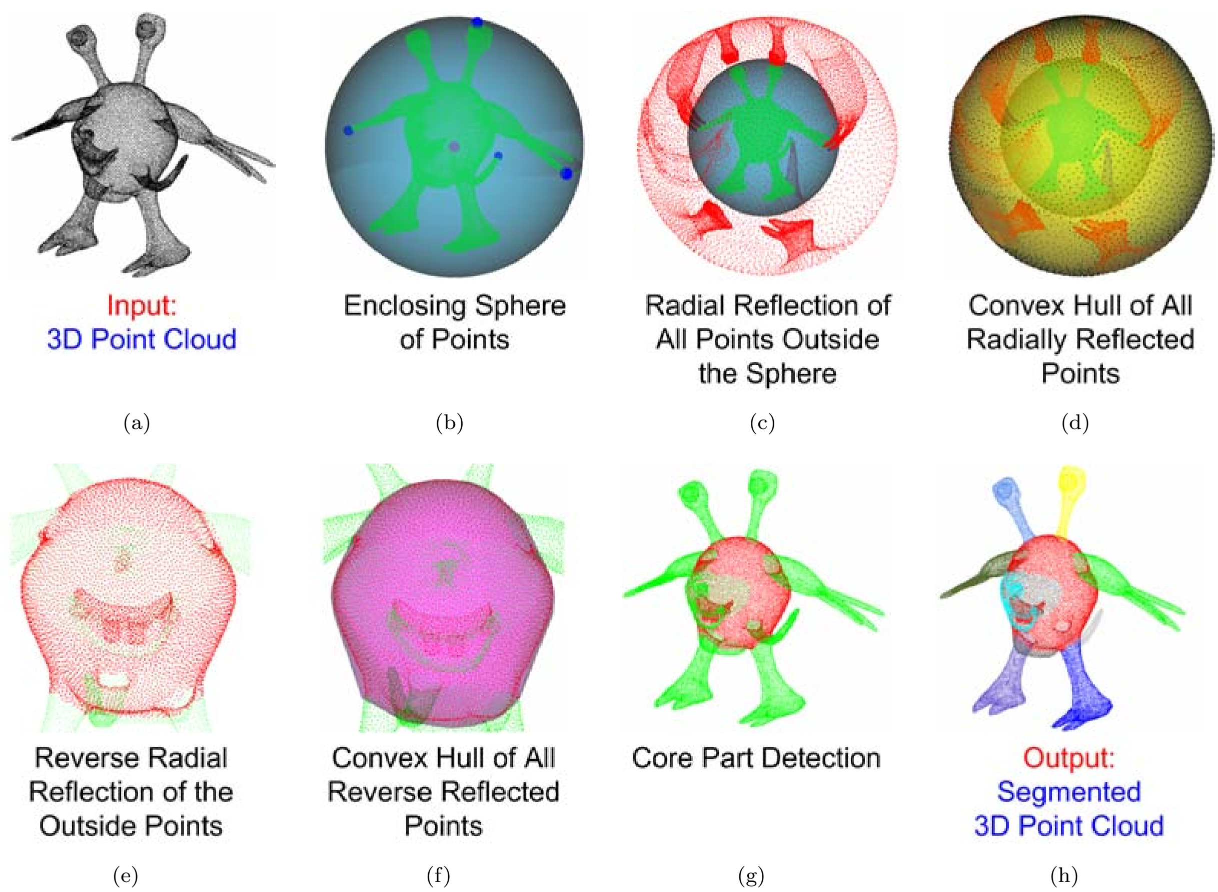


Figure 6.1: Overview of the segmentation algorithm.

part of the object. To carry out a hole-free segmentation of the core part, all mirrored points lying on the convex hull move towards the centre depending on the distances of the neighbouring points [Arya et al., 1998], see Fig. 6.1(e). Based on these points an inner convex hull is calculated. Fig. 6.1(f) shows that this red inner convex hull surrounds the core part of the object, whereby all adhering parts of the core part will be automatically cut off. Fig. 6.1(g) shows the detected core part of the object. Then the algorithm automatically segments the remaining 3D point cloud into a set of sub-parts by recursive flood-filling [Burger and Burge, 2007], see Fig. 6.1(h).

6.2 Segmentation Method

The proposed algorithm consists of two main steps: *"Core Part Extraction"* and *"Cut Refinement"*. Section 6.2.1 presents all the calculation steps for detecting the core part of the object. Section 6.2.2 clarifies all the steps of segmenting the 3D point cloud into a set of sub-parts by recursive flood-filling. Section 6.2.3 presents possible options of solving more complex segmentation tasks.

6.2.1 Core Part Extraction

This section describes every stage of the proposed segmentation algorithm to detect the core part of the object based on the principle of spherical mirroring (radial reflection). The advantage of the algorithm means, that the presented method also segments a mesh with only trivial changes.

The location of the chosen internal centre $\mathcal{C}$ has a strong influence on the segmentation result. There are different ways of computing an internal centre: The way that usually shows the best results is the calculation of the internal centre $\mathcal{C}$ by computing the smallest enclosing sphere of points [Gärtner, 1999]. [Katz et al., 2005] presented a method of approximating the internal centre based on level of hierarchy.

The bounding sphere is defined by the maximum distance $\mathcal{R}$ between the centre $\mathcal{C}$ and all points p_i:

$$\mathcal{R} = max\|p_i - \mathcal{C}\|. \tag{6.1}$$

Fig. 6.1(b) illustrates the calculated bounding sphere (coloured blue), the centre of the sphere (coloured cyan), and the points of the 3D model lying on the sphere (coloured dark blue) with the maximum distance $\mathcal{R}$ to the centre.

Every point p_i of the point cloud $\mathbf{P} = \{p_0, \ldots, p_{n-1}\}$ with n points is spherically mirrored outside the calculated sphere, as illustrated in Fig. 6.1(c) and Fig. 6.2:

$$p_i^{,} = p_i + 2(\mathcal{R} - \parallel p_i - \mathcal{C} \parallel)\frac{(p_i - \mathcal{C})}{\parallel p_i - \mathcal{C} \parallel}. \tag{6.2}$$

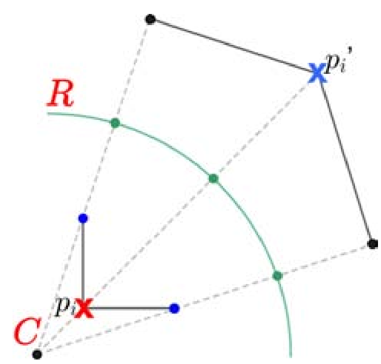


Figure 6.2: Spherical mirroring of all points p_i outside the sphere towards the position $p_i^{,}$ with the calculated radius $\mathcal{R}$ and the centre $\mathcal{C}$ of the sphere.

Thus all points farthest inside the original point cloud become farthest away after this step (see Fig. 6.2). By calculating the convex hull $\mathcal{H}_{out}$ [O'Rourke, 1998] (see Eq. (6.3)) of the reflected point cloud, all k points which are farthest away are automatically detected, as illustrated in Fig. 6.1(d). Each of these k points has a corresponding point on the original point cloud, and these points are fragments of the core part.

$$\mathcal{H}_{out} = ConvexHull\left(\bigcup_{i=0}^{n-1} p_i^{,}\right) \tag{6.3}$$

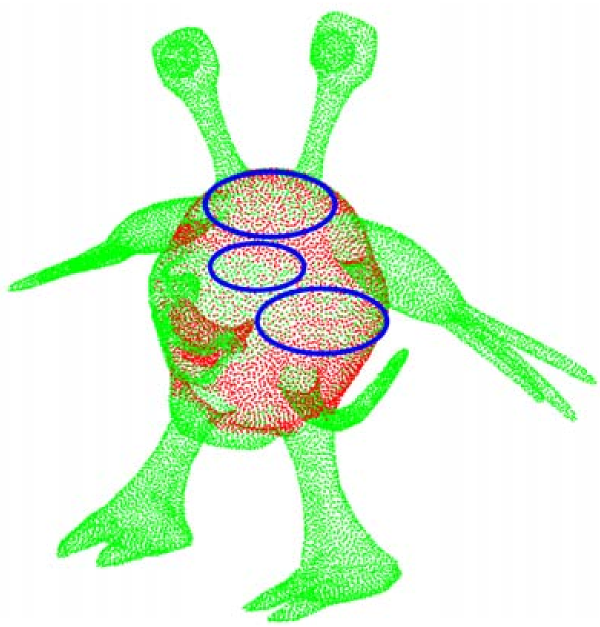


Figure 6.3: Segmented core part with holes. The holes inside the core part are encircled blue.

Fig. 6.3 demonstrates that the detected core part includes a lot of holes. To solve this problem, each point $p_i^{'}$ residing on the convex hull $\mathcal{H}_{out}$ with a corresponding point p_i of the original point cloud will more towards the centre inside the sphere with an offset.

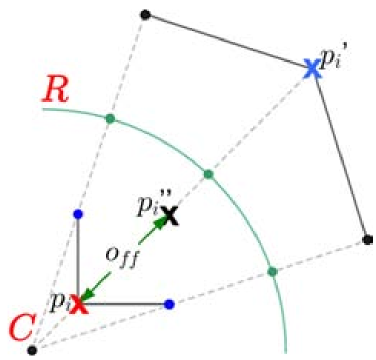


Figure 6.4: All k points, which lie on the convex hull are radially reflected inside the sphere with an offset o_{ff}. The offset o_{ff} is calculated by the neighbouring points of the considered point p_i.

The offset for every point $p_i^{'}$ to eliminate possible holes depends on the distances of the neighbouring points [Arya et al., 1998], as illustrated in Fig. 6.4. Hence, the algorithm calculates the distance between the nearest neighbour for each point of the original point cloud and searches for the maximum distance d_{max} between two closest points. Then the algorithm finds all z neighbouring points p_n with the maximum distance d_{max} for every point p_i with a corresponding point $p_i^{'}$ of the reflected point cloud, and calculates the offset

o_{ff}. This step is important to achieve a hole-free core part:

$$o_{ff} = \frac{\sum_{n=0}^{z-1} ||p_n - \mathcal{C}| - |p_i - \mathcal{C}||}{z}. \tag{6.4}$$

Fig. 6.1(e) shows that all k points $p_i^{,}$ which lie on the outer convex hull $\mathcal{H}_{out}$ will lead to a reverse radial reflected with an own offset:

$$p_i^{,,} = p_i^{,} - 2(\mathcal{R} + o_{ff} - \parallel p_i^{,} - \mathcal{C} \parallel) \frac{(p_i^{,} - \mathcal{C})}{\parallel p_i^{,} - \mathcal{C} \parallel}. \tag{6.5}$$

These k reversely reflected points $p_i^{,,}$ are used to calculate an inner convex hull $\mathcal{H}_{in}$, as illustrated in Fig. 6.1(f) (red convex hull), surrounding the core part:

$$\mathcal{H}_{in} = ConvexHull\left(\bigcup_{i=0}^{k-1} p_i^{,,}\right). \tag{6.6}$$

The resulting inner convex hull $\mathcal{H}_{in}$ surrounds the core part of the object and is used to separate the point cloud into a core part and a remaining part, as illustrated in Fig. 6.1(g).

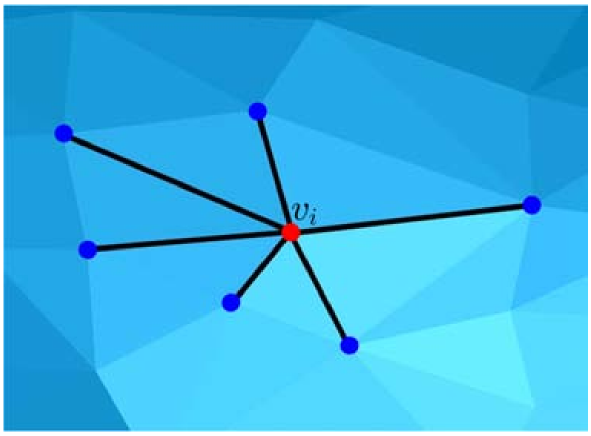
At a mesh $\mathcal{M}$, all points p_i are vertices v_i, which are special kinds of points that describe the corners or intersections of geometric shapes. In this case the offset o_{ff} of a given vertex $v_i^{,}$ on the convex hull $\mathcal{H}_{out}$ is calculated with all z connected vertices v_n of the corresponding vertex v_i of the original mesh (see Eq. (6.7)). At a mesh no closest point and neighbouring search is needed. The neighbouring vertices are the vertices of the adjacent faces of the mesh $\mathcal{M}$, as illustrated in Fig. 6.5.

$$o_{ff} = \frac{\sum_{n=0}^{z-1} ||v_n - C| - |v_i - C||}{z} \tag{6.7}$$

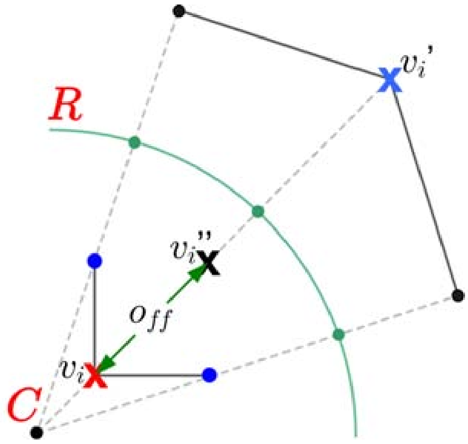
Then $v_i^{,,}$ is calculated with Eq. (6.5) in the same way as with the point cloud. The resulting inner convex hull $\mathcal{H}_{in}$ is used to detect all faces (triangles) inside.

6.2.2 Cut Refinement

This section describes every stage of the proposed segmentation algorithm, which segments the remaining point cloud into different sub-parts. If the core part of the point cloud is found, all the other segments of the remaining point cloud are extracted by recursive flood-filling [Burger and Burge, 2007] as illustrated in Fig. 6.1(h) and Fig. 6.6. An object-part is defined as a set of points with distances between neighbours below a threshold d_a. A kd-tree [Bentley, 1975] finds point neighbours and a region growing method (the recursive flood-filling function [Burger and Burge, 2007]) is used to identify connected point sets. d_a is the average distance between the neighbouring points, calculated by nearest neighbour searching [Arya et al., 1998] of the remaining point cloud.



(a)



(b)

Figure 6.5: Reverse radial reflection of all outside vertices inside the sphere with an offset o_{ff}: **(a)** *All z connected vertices (coloured dark blue) of the adjacent faces are used to calculate an additional offset for the considered vertex (red coloured). This offset o_{ff} is used to avoid potential holes.* **(b)** *The vertex v_i is spherically mirrored outside the sphere to $v_i^{'}$ and if this vertex lies on the convex hull $\mathcal{H}_{out}$, the vertex is transformed inside the sphere with an offset o_{ff} to $v_i^{''}$.*

To belong to a fragment of the object, the distance d between a point p_a and the given point p_i must be smaller than the average distance d_a:

$$d < d_a. \tag{6.8}$$

This region growing method segments the remaining point cloud into different sub-parts.

It is possible to improve the segmentation result with the help of a substantially curvature-based filter [Trucco and Fisher, 1995], a mean shift, Gaussian/mean curvature, additional weighting factors [Richtsfeld and Vincze, 2009a], or a feature point based approach [Katz et al., 2005].

If the segmented object is a mesh $\mathcal{M}$, all sub-parts of the mesh are extracted by a face (triangle) connectivity filter. This filter segments the remaining mesh automatically into different sub-parts.

6.2.3 Options to Solve More Complex Segmentation Tasks

The main characteristic of the presented algorithm is its expandability to solve more complex segmentation tasks. In Fig. 6.7(a)[3], a more complex shape is presented, consisting of alternating convex and concave parts. Fig. 6.7(b) illustrates the segmentation result based on the proposed algorithm. Fig. 6.7(b) shows that the core part is not completely detected and all sub-parts like the legs and the head are not successfully segmented.

To successfully segment the horse, it is possible to expand the algorithm to obtain a pose-invariant model representation. Hence, the algorithm generates a 3D mesh based on

[3]The 3D model is freely available at the Aim@Shape repository.

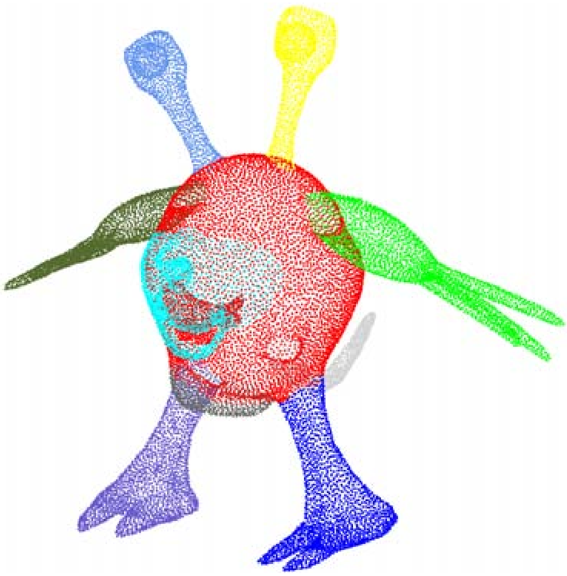
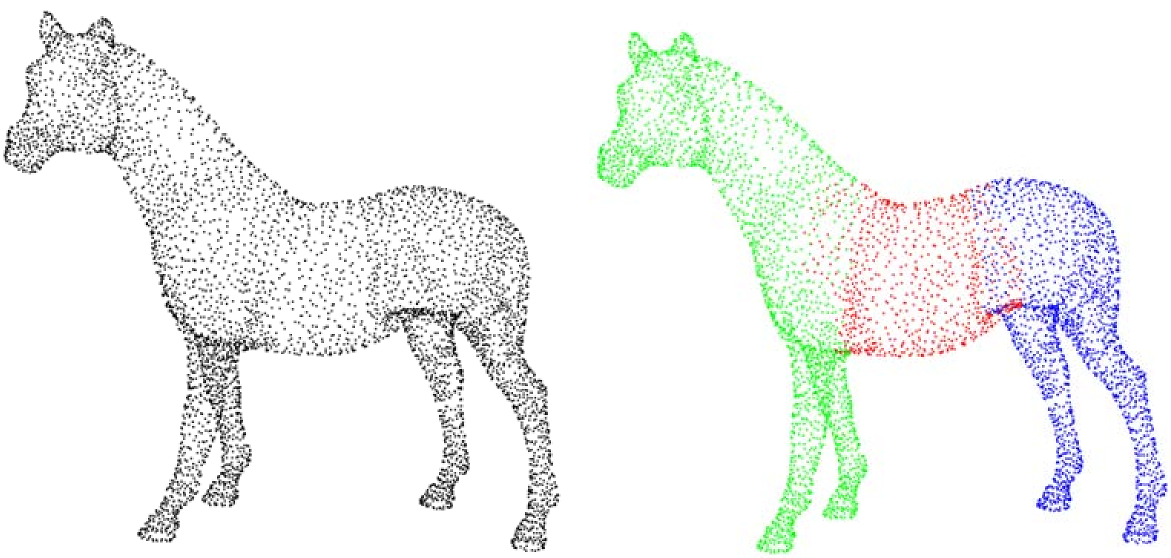


Figure 6.6: Segmentation result based on region growing with the recursive flood filling function.

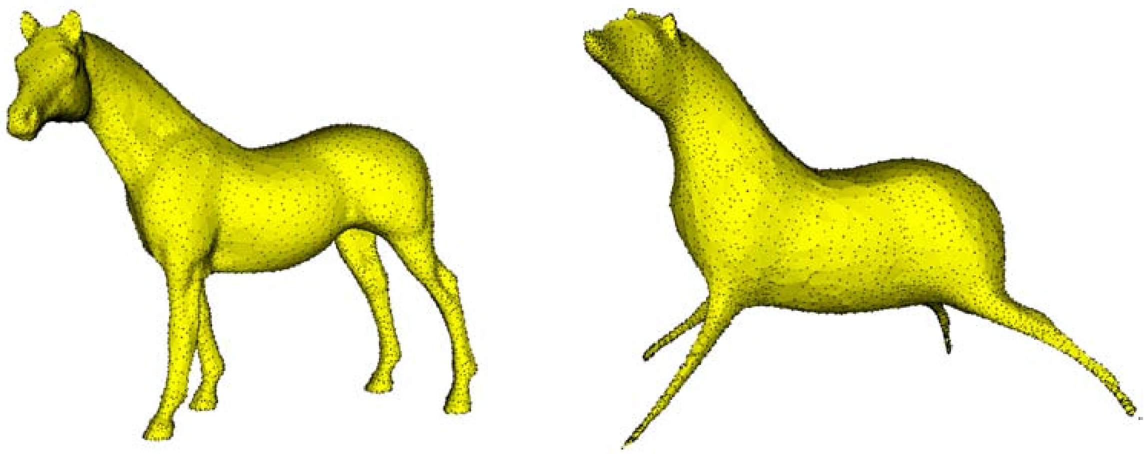


(a) (b)

Figure 6.7: Segmentation of complex shapes: **(a)** *Complex shape with* 5, 360 *points, consisting of alternating convex and concave parts.* **(b)** *Segmentation result based on the proposed algorithm.*

the power crust algorithm [Amenta et al., 2001] (see Fig. 6.8(a)) and multi-dimensional scaling (MDS) is used to get a pose-invariant model representation, see Fig. 6.8(b).

MDS is a generic name for a family of algorithms that construct a configuration of points in a target metric space with information about inter-point distances (dissimilar-



(a) (b)

Figure 6.8: Pose-invariant mesh representation: **(a)** *Triangle mesh calculated with the power crust algorithm, which consists of* 58, 441 *vertices.* **(b)** *Pose-invariant model representation based on MDS.*

ities) measured in some other metric space [Bronstein et al., 2006]. Dissimilarities are defined as geodesic distances δ_{ij} between all vertices v_i on the mesh $\mathcal{M}$ in a symmetrical dissimilarities matrix $\Delta = \mathcal{N} \times \mathcal{N}$ between N points on a Riemannian manifold $\mathcal{S}$. One should distinguish between metric and non-metric MDS (Shephard-Kruskal). Metric MDS preserves the intervals and the ratios between the dissimilarities, whereas non-metric MDS only preserves the order of the dissimilarities.

The scaled gradient-descent algorithm (SMACOF) is used for MDS as published by [Bronstein et al., 2006]. A detailed description of this method can be found in Section 2.6.3. The result of the SMACOF algorithm is a monotonous non-increasing sequence of stress values, which is equivalent to a scaled steepest descent iteration with constant step size.

The aim is to minimise the embedding error, i.e. minimising the sum of distances between the optimally scaled data $f(\delta_{ij})$ and the euclidean distances d_{ij}, where f is an optimal monotonic function (in order to obtain optimally scaled similarities):

$$min_x \sum_{i<j} (d_{ij}(X) - \delta_{ij})^2. \tag{6.9}$$

A stress function $\mathcal{F}_s$ is used to measure the degree of correspondence of the distances between vertices.

Each vertex in MDS space corresponds to a vertex in euclidean space and every point of the original point cloud corresponds to a vertex in euclidean space. In order to speed up the calculation time, the geodesic distances are calculated only on a reduced set of landmark points. Approximately 5, 360 vertices of the generated mesh with a corresponding point of the input point cloud as landmark points lead to an optimal balance between accuracy of representation and time.

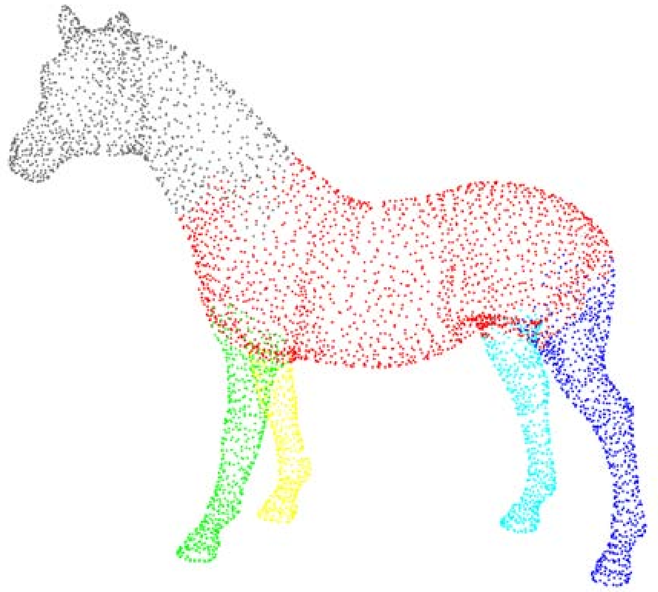


Figure 6.9: Segmentation result based on MDS.

Fig. 6.9 shows the segmentation result based on MDS.

Fig. 6.10[4] illustrates the segmentation results based on pose-invariant model representation of a point cloud and Fig. 6.11 of a mesh. In every case the algorithm successfully segments the arms, the legs, the tail, and the head.

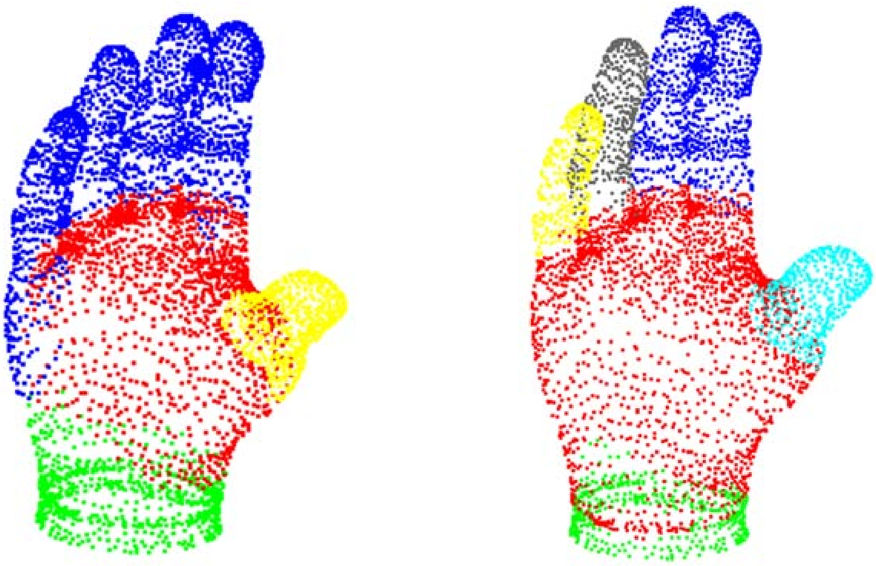
Figure 6.10: Pose-invariance: each model-based on a point cloud is segmented separately.

Fig. 6.12[5] illustrates the influence of MDS. Without MDS the algorithm segments only the palm, whereas the thumb and the fingers are one part. However, with MDS the algorithm also segments the specific fingers. The index and the middle finger are in reality connected, thus the segmentation result is correct.

[4]All models are freely available at the Aim@Shape repository.

[5]Own model, created with the 3D Scanner VIVID 700 of MINOLTA.

Figure 6.11: Pose-invariance: each model-based on a mesh is segmented separately.



(a) (b)

Figure 6.12: Influence of MDS: **(a)** *Result without MDS.* **(b)** *Result with MDS.*

6.3 Segmentation Results

This section presents several segmentation results without MDS on point clouds (see Fig. 6.13) and meshes (see Fig. 6.14). All 3D models are freely available at the Aim@Shape repository.

The presented algorithm fails if the object consists of only one main part because the method always aims to segment the object as illustrated in Fig. 6.15(a). Another drawback is the sensitivity to the location of the internal centre $\mathcal{C}$ of the sphere, which has a strong influence on the segmentation result. If the centre of the bounding sphere is outside the shape of the object (see Fig. 6.15(b)), the algorithm fails (see Fig. 6.15(c)).

Thus, it is essential that the approximated centre is inside the shape of the object. Fig. 6.16 shows that, in this case, a possible solution is the use of the centre of gravity of the object. If no centre inside the shape can be found, the algorithm stops.

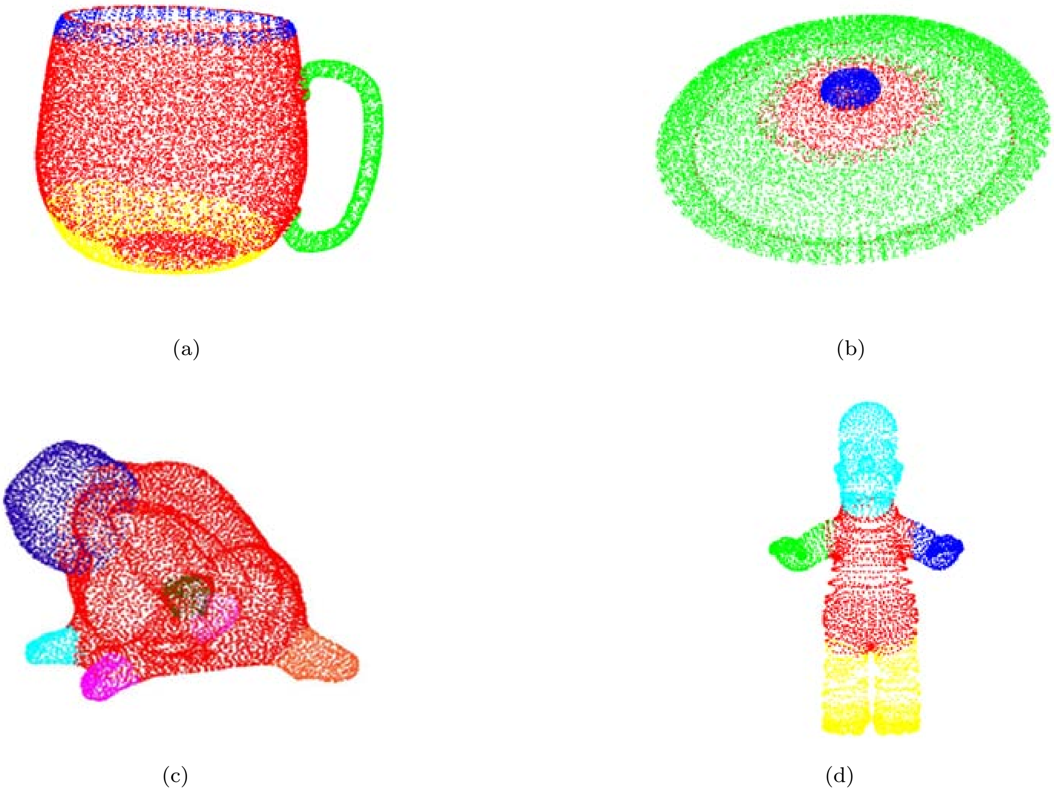


Figure 6.13: Segmentation results based on a point cloud: **(a)** *cup,* **(b)** *cap,* **(c)** *frog, and* **(d)** *man.*

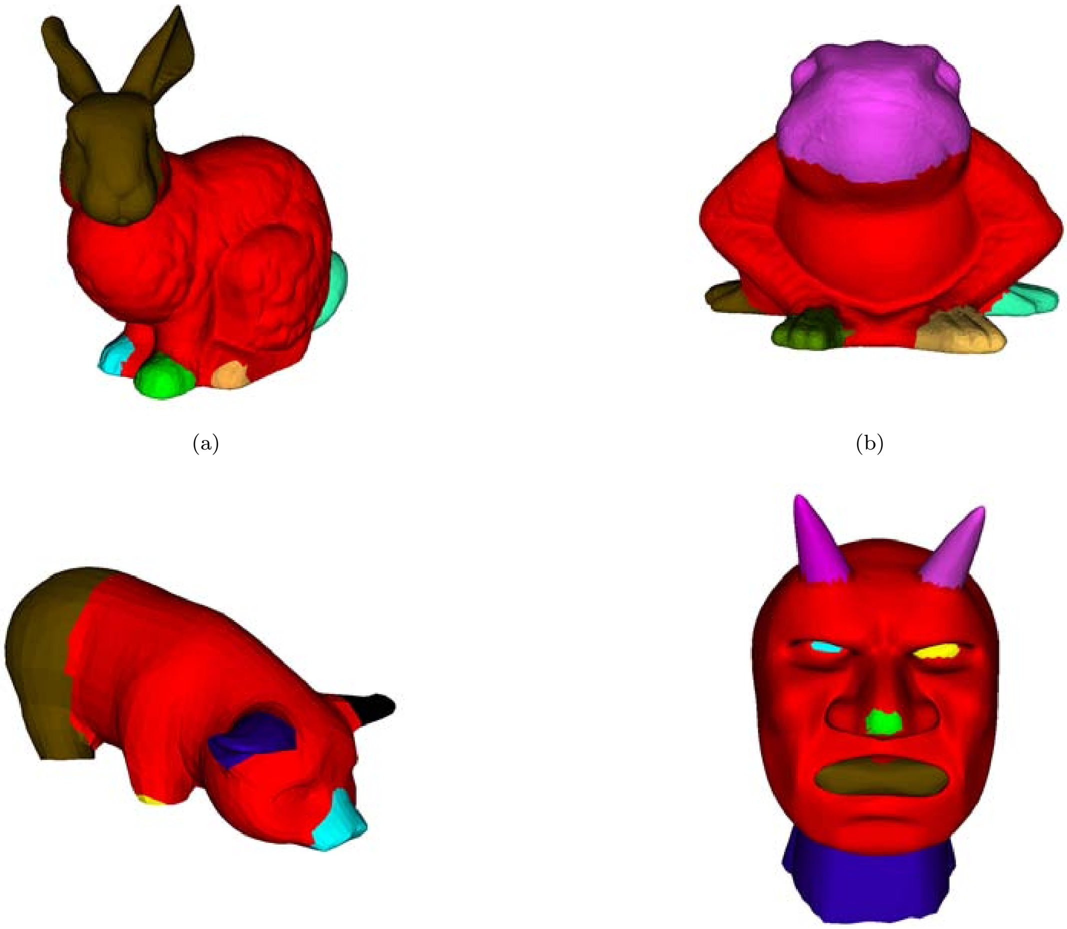
On a 3.2GHz machine with 2GB RAM, the algorithm needs on average $20min$ to generate a pose-invariant mesh representation for a point cloud with $3,000$ points. The time-consuming part is the calculation of the symmetrical dissimilarities matrix $\Delta = \mathcal{N} \times \mathcal{N}$ with all geodesic distances δ_{ij}. Core extraction needs less than 10sec, including the segmentation of all sub-parts of the 3D model. However, the calculation time has a quadric dependence to the number of points. The algorithm is implemented in C++ using the Visualization Tool Kit (VTK)[6].

6.4 Evaluation of the Algorithm

This section compares and evaluates the presented method with the K-Means clustering method [Shlafman et al., 2002], the shape diameter function (SDF) [Shapira et al., 2008] and, in particular the core extraction method [Katz et al., 2005].

The developed algorithm is based on spherical mirroring as the core extraction method, which allows different changes of main influence factors, like the position of the centre for spherical mirroring or the change of the objects topology. The main difference to the

[6]Freely available open source software, http://public.kitware.com/vtk



(a) (b)

(c) (d)

Figure 6.14: Segmentation results based on a mesh: **(a)** *bunny,* **(b)** *frog,* **(c)** *pig, and* **(d)** *grimace.*

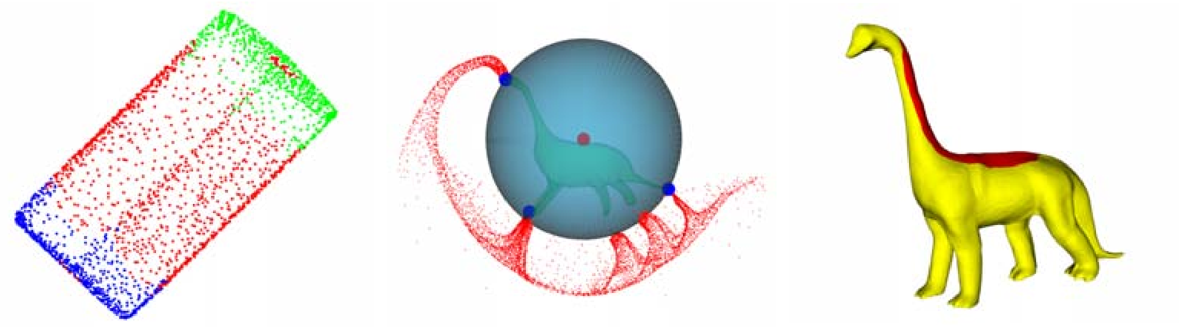
core extraction algorithm is the possibility to segment a point cloud without any fixed thresholds or parameters.

The evaluation has been conducted without MDS based on meshes for a fair comparison because the mesh segmentation results of all 3 other algorithms are also achieved without changing the topology of the object.

Chen et al. [Chen et al., 2009] published a benchmark[7] with different object classes, segmentation results of different methods, and hand segmented models to compare the segmentation results of an own algorithm with others. This work [Chen et al., 2009] is the basis for the realised evaluation with hand segmented models, which are used as ground truth. Additionally, it also summarises fundamental knowledge of the following segmentation methods:

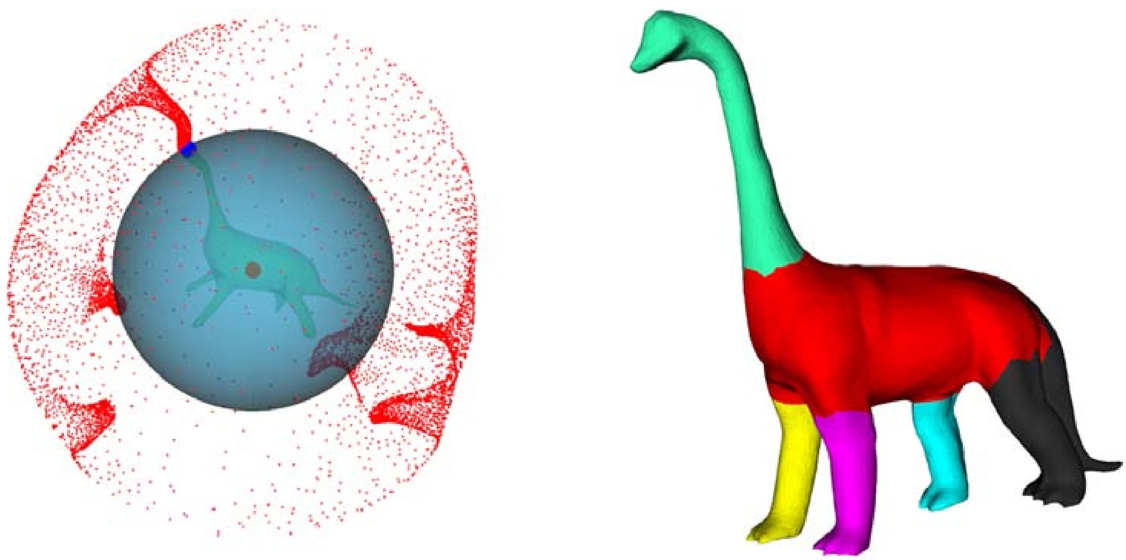
*"**K-Means:** [Shlafman et al., 2002] describe an algorithm based on K-Means*

[7]Freely available benchmark, http://segeval.cs.princeton.edu



(a) (b) (c)

Figure 6.15: Segmentation results: **(a)** *The object consists of only one part.* **(b)** *The centre of the bounding sphere is outside the shape of the object.* **(c)** *Segmentation result if the centre of the bounding sphere is outside the shape of the object.*



(a) (b)

Figure 6.16: Possible solution to get a practical segmentation result: **(a)** *The centre of the bounding sphere is the centre of gravity of the object.* **(b)** *Segmentation result with the centre of gravity of the object as centre of the bounding sphere.*

clustering of faces. Given a user-specified number of segments, k, the algorithm first selects a set of k seed faces to represent clusters by continually selecting the further face from any previously selected. Then, it iterates between: 1. *assigning all faces to the cluster with the closest representative seed, and* 2. *adjusting the seed of each cluster to lie at the centre of the faces assigned to it. This iteration continues until the assignment of faces to clusters converges. The implementation of [Chen et al., 2009] differs from the origi-*

nal [Shlafman et al., 2002], the distances on the dual graph of the mesh are computed with a penalty related to the dihedral angle of each traversed edge using the method in [Funkhouser et al., 2004]."

*"**Shape Diameter Function:** [Shapira et al., 2008] describe an algorithm, which measures the diameter of an objects volume in the neighbourhood of a point on the surface. The SDF is computed for the centroid of every face, and then the segmentation proceeds in two steps. First, a Gaussian Mixture Model is used to fit k Gaussians to the histogram of all SDF values in order to produce a vector of length k for each face indicating its probability to be assigned to each of the SDF clusters. Second, the alpha expansion graph-cut algorithm is used to refine the segmentation to minimise an energy function that combines the probabilistic vectors from step one along with boundary smoothness and concaveness. The algorithm proceeds hierarchically for a given number of "partitioning candidates", which determines the output number of segments. Followed by the authors' advice, the number of partitioning candidates was set to 5, and the algorithm determines the number of segments automatically."*

*"**Core extraction:** [Katz et al., 2005] propose a hierarchical decomposition algorithm that performs four main steps at each stage: transformation of the mesh vertices into a pose insensitive representation using multi-dimensional scaling, extraction of prominent feature points, extraction of core components using spherical mirroring, and refinement of boundaries to follow the natural seams of the mesh. The algorithm partitions segments hierarchically, stopping automatically when the current segment $\mathcal{S}^i$ has no feature points or when the fraction of vertices contained in the convex hull is above a threshold."*

Two metrics have been used to evaluate how well the segmentation results match the human-generated segmentations. The metrics "cut discrepancy" and "rand index" are applied to evaluate the proposed algorithm based on 10 object classes and each object class consists of 20 different models. Fig. 6.17 gives an overview of the 10 object classes.

The first metric "Cut Discrepancy" sums up the distances from points along the cuts to the closest cuts in the ground truth segmentation. The cut discrepancy provides an intuitive measure of how well boundaries align. The disadvantage is its sensitivity to the segmentation granularity. Thus, it is undefined when a model has zero cuts. It decreases to zero if more cuts are added to the ground truth segmentation.

The second metric "Rand Index" measures the likelihood that a pair of faces are either in the same segment in two segmentations or in different segments in both segmentations [Rand, 1971]. A detailed description of the analysed metrics can be found in [Chen et al., 2009].

In the case of the object classes *Bird* and *Cup*, the centre of the sphere is in most cases outside the shape (e.g. cup: 19 of 20 cases), whereas the object class *Mech* the shape usually consists of only one part (in most cases), see Fig. 6.18. Normally, the algorithm stops if the centre is outside the shape, but for this evaluation these object classes are considered for a fair comparison.

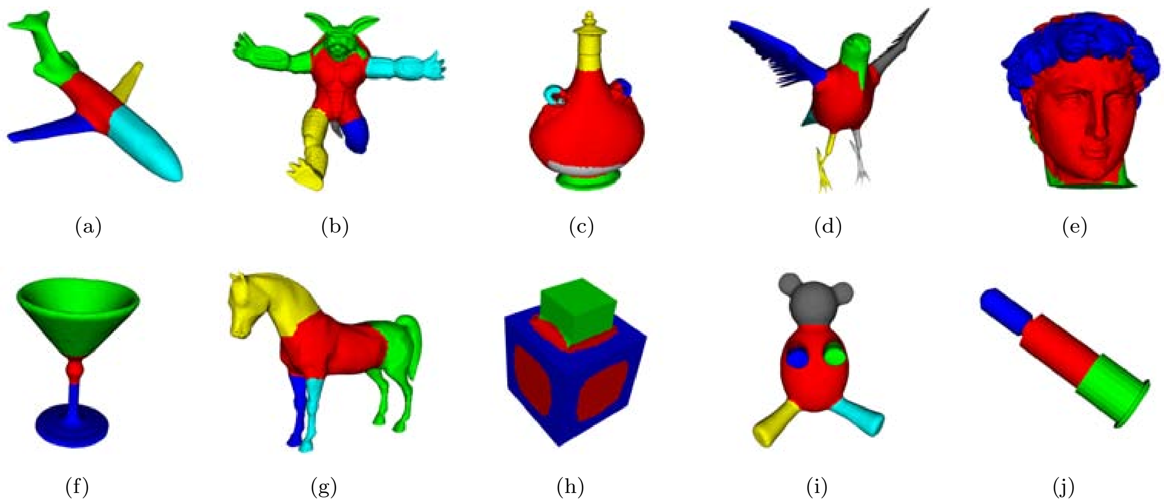


Figure 6.17: Overview of the 10 *different object classes to evaluate the presented algorithm:* **(a)** *Airplane,* **(b)** *Armadillo,* **(c)** *Vase,* **(d)** *Bird,* **(e)** *Bust,* **(f)** *Cup,* **(g)** *Fourleg,* **(h)** *Mech,* **(i)** *Teddy, and* **(j)** *Bearing. The segmentation results are based on the proposed algorithm.*

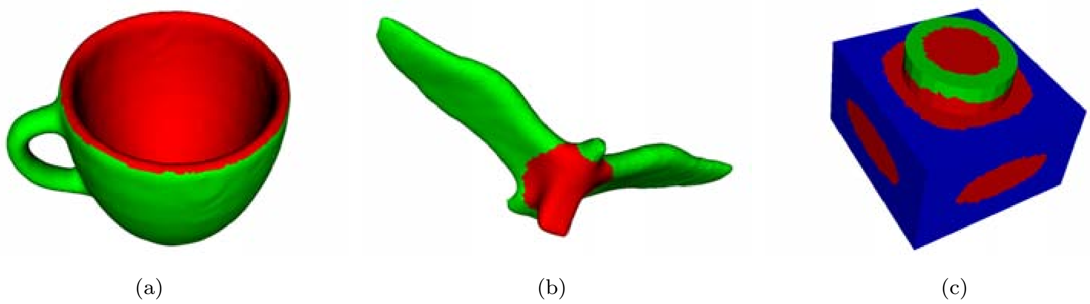


Figure 6.18: The segmentation result fails when the centre is outside the shape or the object consists of only one part: **(a)** *Cup,* **(b)** *Bird, and* **(c)** *Mech.*

The advantage of analysing different object classes with different metrics is discovering which segmentation algorithm attains significant results for which object class.

Fig. 6.19 shows that especially for the object class *Cup*, the cut discrepancy of the presented algorithm reaches the highest value. Thus, the developed method is not suitable for the object class *Cup*, since the computed centre of the sphere is nearly always (in 19 of 20 cases) outside the shape of the object. In comparison with the results of the other methods, the proposed method achieves good results for the object classes *Teddy*, *Mech*, *Bust*, and *Bird*.

Fig. 6.20 illustrates the evaluation result of all object classes for the cut discrepancy. The cut discrepancy of the presented method is better than the cut discrepancy of the core extraction method as a consequence of the detected centre and the consideration of neighbouring points or faces. Fig. 6.20 shows that the shape diameter function obtains the best result over all object classes and the presented algorithm is more suitable than

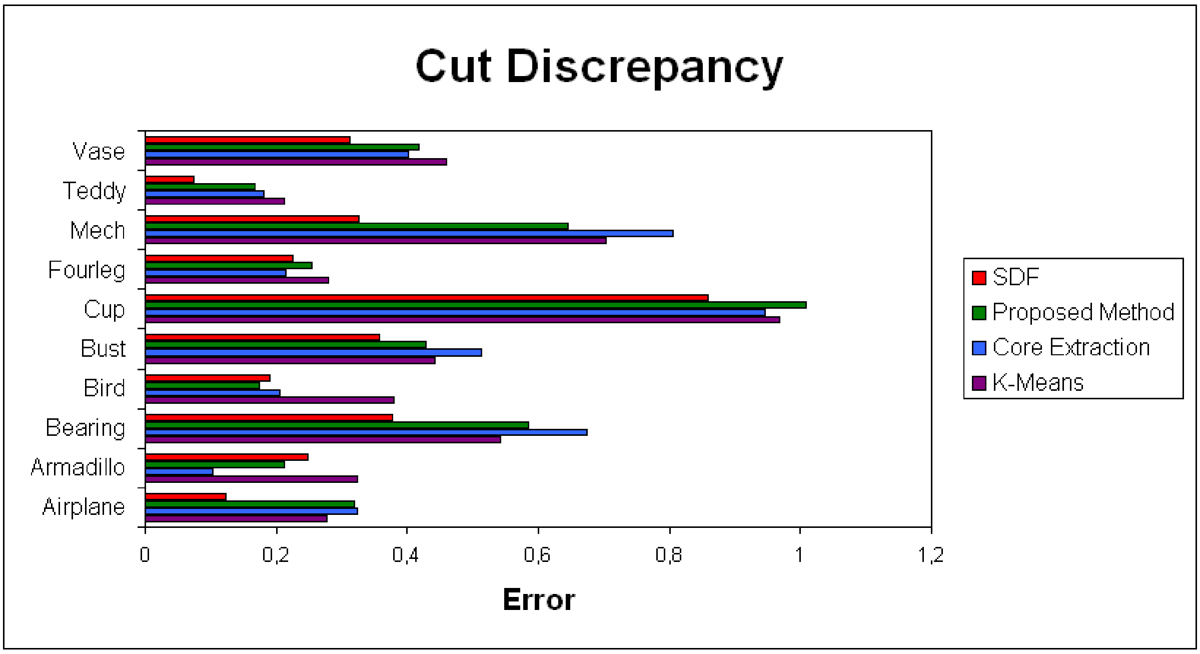


Figure 6.19: Comparison of the cut discrepancy of different segmentation algorithms for all specific object classes.

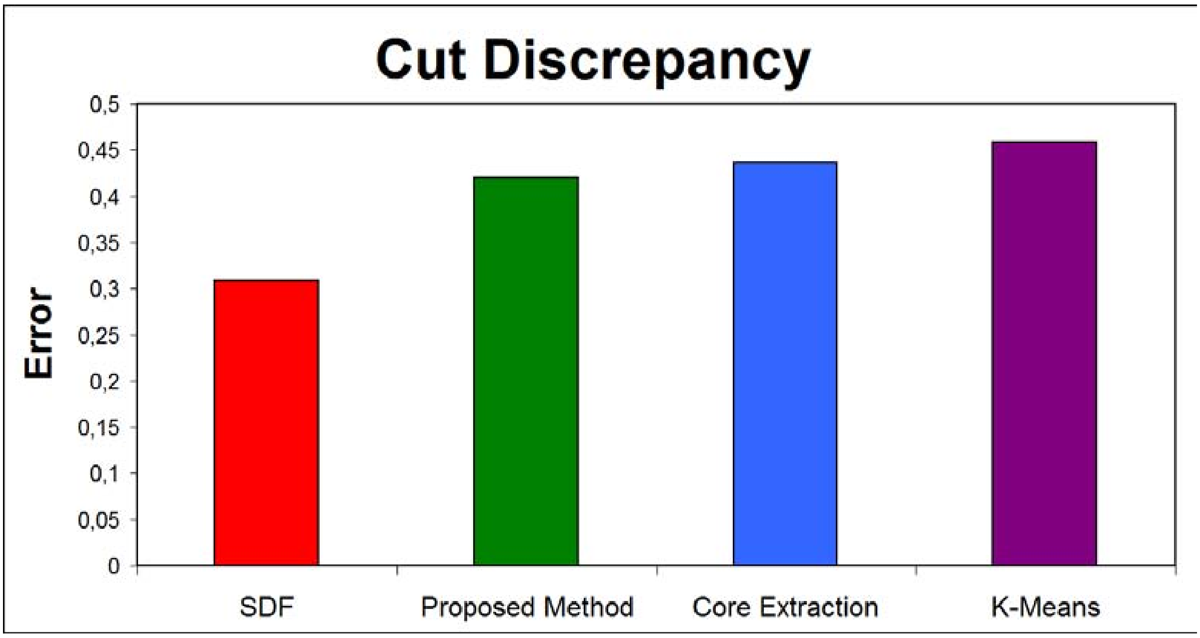


Figure 6.20: Comparison of the cut discrepancy of different segmentation algorithms of all object classes.

the K-Means clustering method.

Fig. 6.21 illustrates the comparison of the rand index of different segmentation algorithms for all specific object classes and Fig. 6.22 illustrates the result of all object classes.

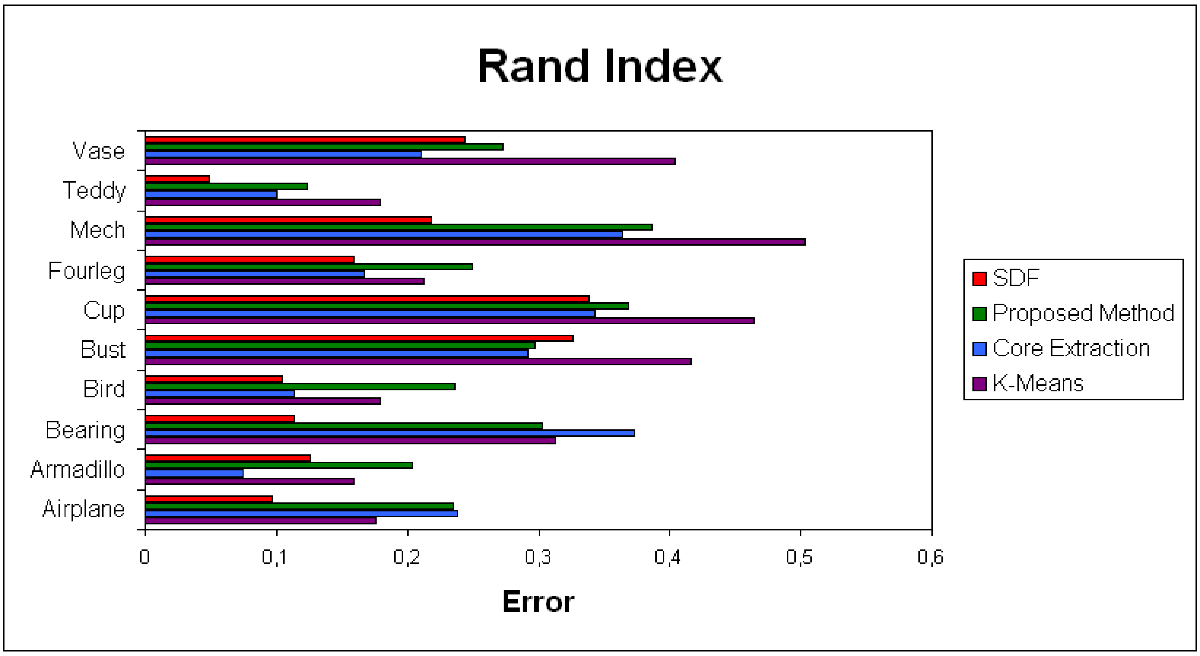


Figure 6.21: Comparison of the rand index of different segmentation algorithms for all specific object classes.

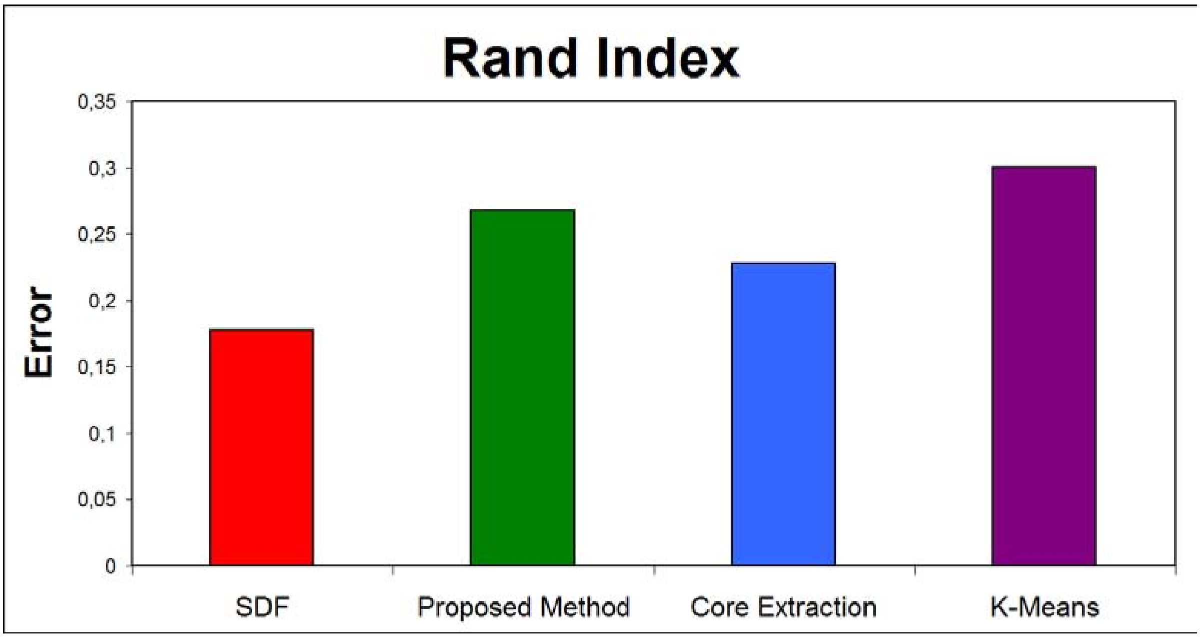


Figure 6.22: Comparison of the rand index of different segmentation algorithms of all object classes.

For nearly every object class the presented method reaches higher values than the core extraction method [Katz et al., 2005]. This result arises from the influence of the iterative process of the core extraction method [Katz et al., 2005] to close the core part based on the detected feature points and the fixed threshold. This method [Katz et al., 2005] has the following drawbacks: This iterative function operates only on meshes and is very time-consuming. The shape diameter function also achieves the best result for the rand index; it only operates on a mesh. However, the rand index of the presented method is more suitable than the rand index of the K-Means clustering method.

Tab. 6.1 shows the average computation time of every segmentation algorithm. In comparison with the other tested methods the proposed method based on radial reflection shows best time performance. The average computation time of the other segmentation methods is based on the values presented in [Chen et al., 2009].

Segmentation Algorithm	t [s]
SDF	8.9
Proposed Method	4.2
Core Extraction	19.5
K-Means	2.5

Table 6.1: Average computation time of every segmentation method.

The result of the presented comparison study reflects the published evaluation result of [Chen et al., 2009] completely.

6.5 Discussion

To begin with the algorithm calculates the internal centre and the radius of the bounding sphere by computing the smallest enclosing sphere of points. This sphere is used to mirror all points outside. Then a convex hull is used to detect all points farthest outside of the reflected point cloud. These outside points correspond with the core part of the input object. To realise a hole-free segmentation of the core part all mirrored points, which lie on the convex hull, are moved towards the centre depending on the distances of the neighbouring points. Based on these points, an inner convex hull which surrounds the core part of the object is calculated. Then, the algorithm automatically segments the remaining 3D point cloud into a set of sub-parts by a region growing method.

An experimental evaluation of 10 different object classes demonstrates the robustness and efficiency of the proposed algorithm and the results prove that it compares well with a number of state-of-the-art 3D object segmentation algorithms. A comparison study has been done with the K-Means clustering method, the shape diameter function (SDF), and the core extraction method. Additionally, the metrics "cut discrepancy" and "rand index" of the proposed method have been evaluated. To realise a fair comparison, the evaluation is based on the original meshes since the mesh segmentation results of all three other algorithms are also achieved without changing the topology of the object.

The result for the cut discrepancy of the presented method is more exact than the result of the core extraction and the K-Means clustering method as a consequence of the detected centre and the consideration of the neighbouring points or faces. The core extraction algorithm obtains better results for the rand index as it uses an iterative process to close the core part. However, better results are achieved with the proposed algorithm than with the K-Means clustering method. The shape diameter function (SDF) achieves best results for both metrics. The presented comparison study reflects the published evaluation result of [Chen et al., 2009] completely.

The proposed segmentation algorithm represents a flexible and completely automatic way of segmenting a 3D point cloud or a mesh. In comparison with other methods the proposed algorithm based on radial reflection shows best time performance. The algorithm works directly on point clouds and meshes, and shows a high reliability of segmenting an object into parts that correspond to relevant features. Additionally, the presented algorithm works without any model assumptions and fixed thresholds, making it possible to segment models deducted from different systems. A further significant characteristic of the presented algorithm is its expandability to solve more complex segmentation tasks. Applying MDS to change the topology of the shape to create a pose-invariant model representation is only one possibility.

Chapter 7

Conclusion and Future Work

"Science is a wonderful thing if one does not have to earn ones living at it."
– *Albert Einstein*

This dissertation presents robust feature detection approaches in range images to realise stable object detection, segmentation, and grasping tasks in variable environments with a restricted field of view. In particular, novel model-based methods are proposed to achieve robust and reliable detection results of object features related to specific handling, and manipulation tasks in industrial as well as in service robotics. Due to the high amount of point data, another issue of the presented methods is to keep the processing time low to rapidly detecting features and objects for industrial and robotic applications.

One major contribution of this thesis is the development of methods to detect relevant features for following robotic tasks as edge tracking and grasping. Additionally, a novel method to segment a 3D point cloud as well as a mesh into useful sub-parts was developed. The effectiveness of these contributions is demonstrated by showing results from a variety of different approaches to take a step towards full automation of industrial and robotic applications.

7.1 Summary and Contributions

The presented project "REDUX" in Chapter 3 illustrates the feasibility of an autonomously guided robot stitching system to realise a lot-size-one production. To achieve a better solidity and durability of two overlapping carbon fibre layers, an automated fibre mat stitching system is needed. At the moment, the stitching path is predefined and inflexible to minor changes. A fully automated sewing process is developed to realise a more flexible stitching system and to increase the efficiency of the whole fabrication process.

Contributions in this field have been made by introducing a real-time approach. Existing and newly developed edge detection methods are combined in a voting scheme to increase the edge tracking robustness. The novelty is based on different edge detection methods and their combination to carry out robust edge tracking in the presence of outliers and artefacts in noisy range data. The individually developed methods show very high reliability. The experiments illustrate that a two out of three voting of three methods achieves a better detection result than the individual methods. Thus, the voting scheme for edge detection and localisation is suitable for the use in related industrial applications under difficult conditions. The experiments show that for the “Base Sensor” a two out of three voting over the three developed methods achieves a detection result of 99.3% and the edges are located within $1.0mm$. The “3 Lines Sensor” achieves a detection result of 86.4% by replacing the algorithm results with sum of weight functions and historic edge positions. For both sensors the achieved detection results are totally sufficient for the sewing process, where draping is only accurate to a few centimeters.

Chapter 4 presents a system of detecting and grasping given cylindrical objects with cluttered adjacent objects on a table in soft real-time. The method is optimised for fitting cylinders in sparse and noisy range data under difficult and changing light conditions recorded from a single view. The contributions focus on the treatment of different objects on the table. The system must distinguish between them and detect the defined cylindrical objects. The entire system exhibited its practical behaviour at a live demo presentation.

The experimental evaluation shows that the developed fitting method for cylindrical objects achieves reliable results in comparison to standard least-squares cylinder fitting methods. The radius deviation and also the angle deviation are essentially smaller than the results of the least-squares cylinder fitting method. The processing time depends on the number of raw data points of the whole point cloud.

Chapter 5 introduces a method of automatic grasping of unknown objects with a 3D model of the used gripper. The approach to object grasping is well suited for the use in related applications under different conditions and can be applied to a reasonable set of objects. From a single view, the rear side of an object is not visible due to self occlusions and the front side may be occluded by other objects. The algorithm is developed for arbitrary objects in different poses, on top of each other or side by side, with a special focus on rotationally symmetric objects. If objects cannot be separated because they are stacked on top of each other, they are considered as one object. If the algorithm detects clipped rotationally symmetric parts, these parts are merged because this object class can be robustly identified and allows a cylindrical grasp as well as a tip grasp along the top rim. For all other objects, the algorithm calculates a power grasp based on the detected top surface of the object. The algorithm reaches a grasping rate of 80.65% for several objects, 77.57% for touching objects, 82.70% for objects in a box, and 79.40% for a bunch of objects. With dense stereo data, a grasping rate of 66.62% was reached. The success of the grasping algorithm depends on ambient light, object surface properties, laser-beam reflectance, and absorption of the objects. For point data based on dense stereo reconstruction the texture of the objects must be sufficient.

Chapter 6 presents a flexible and completely automatic way of segmenting a 3D point cloud. The algorithm works directly on point clouds as well as meshes and shows a high reliability for segmenting an object into sub-parts corresponding to the relevant features. Additionally, the presented algorithm works without any model assumptions and fixed thresholds, which makes it possible to segment models deducted from different systems.

An experimental evaluation of 10 different object classes demonstrates the robustness and the efficiency of the proposed algorithm. A comparative study has been done with the K-Means clustering method, the shape diameter function (SDF) and the core extraction method. Additionally, the metrics "cut discrepancy" and "rand index" of the proposed method have been evaluated. In comparison with the other segmentation methods the proposed algorithm based on radial reflection shows best time performance. The results prove that it compares well with a number of state-of-the-art 3D object segmentation algorithms.

7.2 Open Research Work

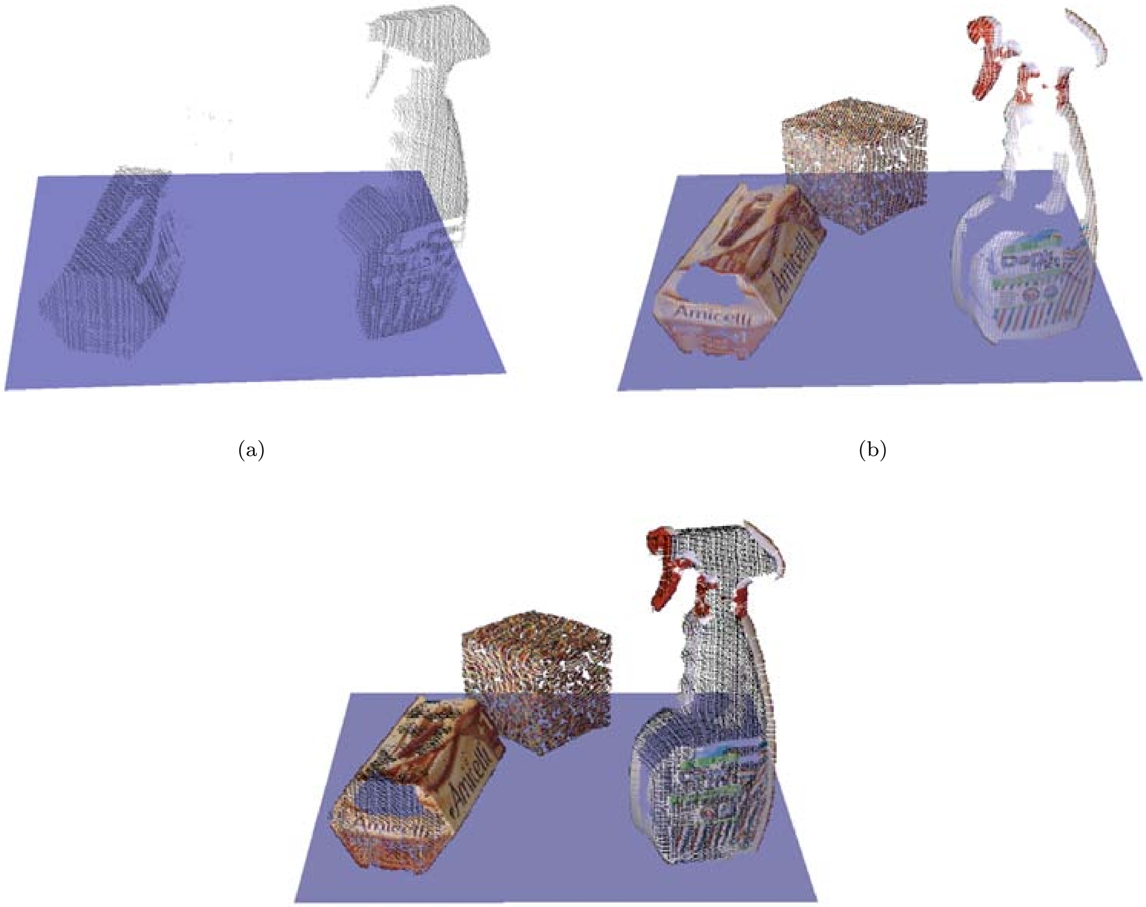
Today, a lot of different systems to grasp and manipulate known objects in well-defined poses exist. A challenging task which should be tackled in the future is to develop a fully automatic system to realise function-based grasping and manipulation of arbitrary objects. The presented methods in Chapter 4, Chapter 5, and Chapter 6 represent a basis for solving this challenging task.

This section illustrates a way for a totally independent mobile robot platform which processes given commands like "James, please bring me my cup!" and behaves autonomously to solve the given tasks. A mobile robot should be able to perform actions in an unknown, variable environment. It should grasp, manipulate and place objects, open and close doors and cupboards, and assist elderly and handicapped people.

Laser range data offer high accuracy, whereas dense stereo systems are more flexible. With the assistance of a laser range scanner, the typical problem is missing sensor data because of absorption, as illustrated in Fig. 7.1(a). Fig. 7.1(b) shows that a dense stereo system needs textured objects to realise a 3D reconstruction. Laser range scanners are essential to obtain accurate point clouds. To filter the noisy stereo point cloud, the described geometrical filter in Section 4.2.2 can be used. The combination reflects the advantages of both systems, as shown in Fig. 7.1(c). Hence, both systems should be mounted and calibrated on the robot arm. The system should attempt together different views of the interesting object by either moving the robot arm or the object.

All point clouds of the different views can be merged into one point cloud with the well-known trajectory of the robot to obtain more object information. Afterwards, all objects in this point cloud will be segmented for the following steps, as illustrated in Fig. 7.2(a). For the segmentation step, the presented methods in Chapter 4 and Chapter 5 can be used.

Fig. 7.2(b) shows the detection of the pose of the *Amicelli box*, which can be reached with a registration [Pottmann et al., 2004] or iterative closest point (ICP) [Besl and McKay, 1992] method. The use of the colour information of the dense stereo data would be an extension of existing registration methods. Function-based grasping poses should be predefined for known objects and the algorithm will automatically find the best gripper pose in regard to all the objects in clutter. For unknown objects

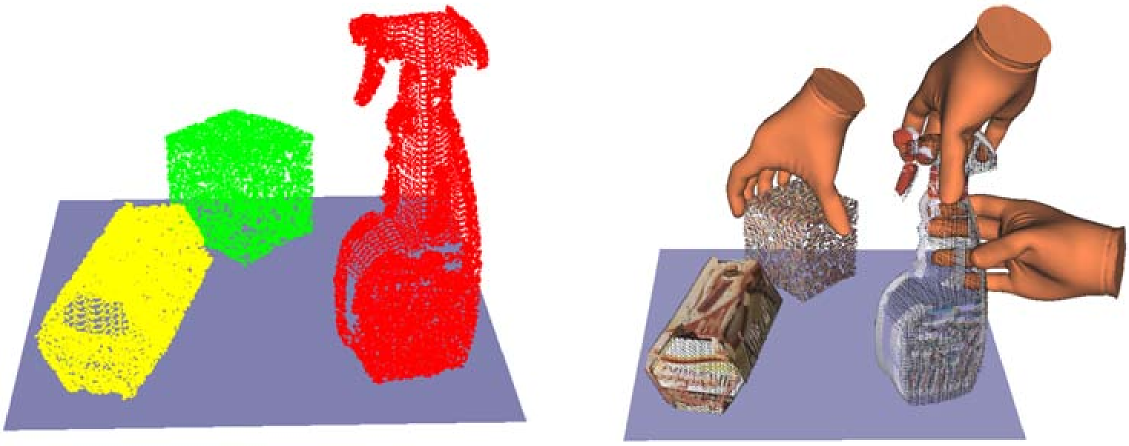


(a) (b) (c)

Figure 7.1: Combination of laser range and dense stereo data: **(a)** *Point cloud based on a laser range scanner. It is clearly visible that the range scanner is not able to scan the cube, because of absorption.* **(b)** *Point cloud based on dense stereo data, including a lot of noise and outliers.* **(c)** *Combination of laser range and dense stereo data to utilise the advantages of both systems.*

potential grasping poses should be detected based on the detected features, such as top surfaces, handles, and parallel surfaces, as described in Chapter 5.

The detected power grasp for the *cube* and the top power grasp for the *cleaning aerosol can* are based on the presented method in Chapter 5. The lateral grasping pose for the *cleaning aerosol can* represents a possible expansion of the grasping features. These features should be extended for more object classes in the future.



(a) (b)

Figure 7.2: Segmentation and grasping pose detection: **(a)** *Segmentation result of the combined point cloud.* **(b)** *Detection of the correct object pose of the known object (Amicelli box) and detection of possible grasping poses based on detected features.*

Glossary

3D – Three Dimensional.

CAD – **C**omputer **A**ided **D**esign.
CCD – **C**harge **C**oupled **D**evice – camera's sensor chip technology.
CFK – **C**arbon**F**aserverstärkter **K**unststoff.
CFRP – **C**arbon **F**ibre **R**einforced **P**olymer.

DOF – **D**egrees **O**f **F**reedom – in most cases the degrees of freedom are related to translation and rotation in an Euclidean 3D space.

GC – **G**eneralised **C**ylinder.
GEONS – **GEO**metric ico**NS**:
In 1987, Biederman proposed the "Recognition by Components" theory which offers an explanation of human visual object recognition [Biederman, 1987]. The fundamental assumption of this theory is that there exists a small number (i.e. 36) of fundamental part primitives, whose boolean combinations can represent more complicated objects for the purpose of "primal access".
GUI – **G**raphical **U**ser **I**nterface.

HFP – **H**ierarchical **F**itting **P**rimitives.

ICP – **I**terative **C**losest **P**oint.
ILGA – **I**nfant **L**earning **G**rasping and **A**ffordances.
ILGM – **I**nfant **L**earning to **G**rasp **M**odel.

LASER – **L**ight **A**mplification by **S**timulated **E**mission of **R**adiation.
LSCF – **L**east-**S**quares **C**ylinder **F**itting.
LSF – **L**east-**S**quares **F**itting.
LUT – **L**ookup-**T**able.

MAT – **M**edial **A**xis **T**ransform.
MDS – **M**ulti-**D**imensional **S**caling.

NURBs – **N**on **U**niform **R**ational **B**-spline surfaces.

OLP – **O**ff-**L**ine **P**rogramming – this term is used in robot programming.

Origin – Define the coordinate frame centre.

PCA – **P**rincipal **C**omponent **A**nalysis – image processing methodology [Xu et al., 1992].

POSE – Position and Orientation – if nothing else is noted the pose is defined with 6 DOF (3 for the position and 3 for the orientation in a 3D space).

Power Grasp – A power grasp is commonly defined as a grasp type, which uses many contact points, which are not only on the fingertips but also on the inner side of the fingers and even the palm. The advantage of a power grasp is the passive resistance of the hand against external forces, which provides robustness against external disturbances [Zhang et al., 1994].

ppt – **p**re-**p**rocessing time.

px – **Pic**ture **El**ement or **P**i**X**el is the smallest item of information in an image.

QA – **Q**uality **A**ssurance.

Range Data – Three dimensional point cloud measured from a laser range scanner.

Range Image – Three dimensional point cloud measured from a laser range scanner.

RANSAC – **Ran**dom **S**ample **a**nd **C**onsensus – algorithm proposed by [Fischler and Bolles, 1981].

RCF – **R**andom **C**ircle **F**itting.

Real-Time – Definition given in the German industry standards, DIN 44300:
The operating mode of a computer system in which the programs for the processing of data arriving from the outside are permanently ready, so that their results will be available within predetermined periods of time; the arrival times of the data can be randomly distributed or be already a priori determined depending on the different applications.

RGC – **R**ight **G**eneralised **C**ylinder.

RTM – **R**esin **T**ransfer **M**oulding technique.

SDF – **S**hape **D**iameter **F**unction.

SMACOF – **S**caled **G**radient **D**escent **A**lgorithm.

STL – **St**ereo **L**ithography - file format for a 3D mesh.

SUVA sensor – **S**chweizerische **U**nfallversicherungsanstalt sensor, Luzern/Schweiz.

TCP – **T**ool **C**entre **P**oint – tool reference point of a robot arm.

THOR – **T**ool **H**andling the **O**perations of **R**obots.

UDP – **U**ser **D**ata **P**rotocol.

VAP – **V**acuum **A**ssisted **P**rocess technique.

VTK – **V**isualisation **T**ool **K**it – freely available open source software, http://public.kitware.com/vtk.

XML – e**X**tensible **M**arkup **L**anguage.

Bibliography

[Aarno et al., 2007] Aarno, D., Sommerfeld, J., Kragic, D., Pugeault, N., Kalkan, S., Wörgötter, F., Kraft, D., and Krüger, N. (2007). Early Reactive Grasping with Second Order 3D Feature Relations. In *ICRA Workshop: From Features to Actions*, pages 319–325.

[Altrichter, 2008] Altrichter, M. (2008). *Verhaltensbasierte Steuerung für einen Robotermanipulator*. Master Thesis, Vienna University of Technology.

[Amenta et al., 2001] Amenta, N., Choi, S., and Kolluri, R. (2001). The Power Crust. In *Proceedings of the 6^{th} ACM Symposium on Solid Modeling and Applications*, pages 249–260.

[Arya et al., 1998] Arya, S., Mount, D., Netanyahu, N., Silverman, R., and Wu, A. (1998). An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions. *Journal of the ACM*, 45(6):891–923.

[Attene et al., 2006a] Attene, M., Falcidieno, B., and Spagnuolo, M. (2006a). Hierarchical mesh segmentation based on fitting primitives. *Springer; The Visual Computer: International Journal of Computer Graphics*, 22(3):181–193.

[Attene et al., 2006b] Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., and Tal, A. (2006b). Mesh Segmentation - A Comparative Study. In *Proceedings of the IEEE Conference on Shape Modeling and Applications*, pages 7–18.

[Bab-Hadiashar and Gheissari, 2006] Bab-Hadiashar, A. and Gheissari, N. (2006). Range Image Segmentation Using Surface Selection Criterion. *IEEE Transactions on Image Processing*, 15(7):2006–2018.

[Bahlmann and Heidemann, 1999] Bahlmann, C. and Heidemann, G. (1999). Artificial Neural Networks for Automated Quality Control of Textile Seams. *Pattern Recognition*, 32(6):1049–1060.

[Barr, 1981] Barr, A. (1981). Superqadrics and Angle-Preserving Transformations. *IEEE Computer Graphics and Applications*, 1(1):11–23.

[Belmonte et al., 2004] Belmonte, Ó., Remolar, I., Ribelles, J., Chover, M., and Fernández, M. (2004). Efficiently using connectivity information between triangles in a mesh for real-time rendering. *Elsevier Science*, 20(8):1263–1273.

[Bentley, 1975] Bentley, J. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(19):509–517.

[Bernardini et al., 1999] Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., and Taubin, G. (1999). The Ball-Pivoting Algorithm for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359.

[Besl and Jain, 1985] Besl, P. and Jain, R. (1985). Three-Dimensional Object Recognition. *ACM; Computing Survey*, 17(1):75–145.

[Besl and McKay, 1992] Besl, P. and McKay, H. (1992). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligenc*, 14(2):239–256.

[Biederman, 1987] Biederman, I. (1987). Recognition-by-Components: A Theory of Human Image Understanding. *APA Journal; Psychological Review*, 94(2):115–147.

[Biegelbauer et al., 2007] Biegelbauer, G., Richtsfeld, M., Wohlkinger, W., Vincze, M., and Herkt, M. (2007). Optical Seam Following for Automated Robot Sewing. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 4758–4763.

[Biegelbauer and Vincze, 2000] Biegelbauer, G. and Vincze, M. (2000). Fast and Robust Bore Detection in Range Image Data for Industrial Automation. In *Proceedings of the 2^{nd} International Symposium on 3D Data Processing*, pages 526–533.

[Biegstraaten et al., 2003] Biegstraaten, M., Smeets, J., and Brenner, E. (2003). The influence of obstacles on the speed of grasping. *Springer; Experimental Brain Research*, 149(4):530–534.

[Boissonnat, 1984] Boissonnat, J. (1984). Geometric Structures for Three-Dimensional Shape Representation. *ACM Transactions on Graphics*, 3(4):266–286.

[Bone et al., 2008] Bone, G., Lambert, A., and Edwards, M. (2008). Automated Modelling and Robotic Grasping of Unknown Three-Dimensional Objects. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 292–298.

[Borst et al., 2003] Borst, C., Fischer, M., and Hirzinger, G. (2003). Grasping the Dice by Dicing the Grasp. In *Proceedings of the IEEE/RSJ Conference on Robotics and Systems*, pages 3692–3697.

[Boughorbel and Zhang, 2007] Boughorbel, F. and Zhang, Y. (2007). Laser ranging and video imaging for bin picking. *Assembly Automation*, 23(1):53–59.

[Brechbühler et al., 1992] Brechbühler, C., Gerig, G., and Kübler, O. (1992). Surface Parametrization and Shape Description. In *Proceedings of the SPIE: Visualization in Biomedical Computing*, volume 1808, pages 80–89.

[Bronstein et al., 2008] Bronstein, A., Bronstein, M., and Kimmel, R. (2008). *Numerical Geometry of Non-Rigid Shapes.* Springer.

[Bronstein et al., 2006] Bronstein, M., Bronstein, A., Kimmel, R., and Yavneh, I. (2006). Multigrid Multi-Dimensional Scaling. *Numerical Linear Algebra with Applications (NLAA), Special issue on multigrid methods*, 13(2–3):149–171.

[Brouwer et al., 2009] Brouwer, A., Franz, V., and Gegenfurtner, K. (2009). Differences in fixations between grasping and viewing objects. *Journal of Vision*, 9(1):1–24.

[Burger and Burge, 2007] Burger, W. and Burge, M. (2007). *Digital Image Processing – An Algorithmic Introduction Using Java*. Springer, UK, London, 1^{st} edition.

[Canny, 1986] Canny, J. (1986). A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

[Casals et al., 2000] Casals, A., Cufi, X., Freixenet, J., Marti, J., and Muñoz, X. (2000). Friendly Interface for Objects Selection in a Robotized Kitchen. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 4070–4075.

[Castiello, 2005] Castiello, U. (2005). The neuroscience of grasping. *Nature Reviews Neuroscience*, 6(9):726–736.

[Cazals et al., 2009] Cazals, F., Chazal, F., and Giesen, J. (2009). Spectral Techniques to Explore Point Clouds in Euclidean Space, with Applications to the Inference of Collective Coordinates in Structural Biology. *Nonlinear Computational Geometry*.

[Chen et al., 2009] Chen, X., Golovinskiy, A., and Funkhouser, T. (2009). A Benchmark for 3D Mesh Segmentation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH'09)*, 28(3).

[Coquillart, 1987] Coquillart, S. (1987). A Control-Point-Based Sweeping Technique. *IEEE Computer Graphics and Applications*, 7(11):36–45.

[Cornea et al., 2005] Cornea, N., Silver, D., Yuan, X., and Balasubramanian, R. (2005). Computing Hierarchical Curve-Skeletons of 3D Objects. *The Visual Computer*, 21(11):945–955.

[Detry et al., 2009] Detry, R., E. Başeski, M. P., Touati, Y., Krüger, N., Kroemer, O., Peters, J., and Piater, J. (2009). Learning Object-specific Grasp Affordance Densities. In *Proceedings of the* 8^{th} *IEEE Conference on Development and Learning*, pages 1–7.

[Dewing et al., 1999] Dewing, A., Sarhadi, M., Mitchell, T., Davis, R., and Eastham, J. (1999). Robotic Tacking of High-Quality Composite Preforms. *Proceedings of the I MECH E Part G Journal of Aerospace Engineering*, 213(3(G)):173–185.

[Dior et al., 1997] Dior, D., Laurendeau, D., and Bergevin, R. (1997). Generalized Cylinders Extraction in a Range Image. In *Proceedings of the IEEE Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 141–147.

[Djebali et al., 2002] Djebali, M., Melkemi, M., and Sapidis, N. (2002). Range-Image Segmentation and Model Reconstruction Based on a Fit-and-Merge Strategy. In *Proceedings of the seventh ACM symposium on Solid modeling and applications*, pages 127–138.

[Dorrity, 1995] Dorrity, J. (1995). New Developments for Seam Quality Monitoring in Sewing Applications. *IEEE Transactions on Industry Applications*, 31(6):1371–1375.

[Duda and Hart, 1972] Duda, R. and Hart, P. (1972). Use of the Hough Transformation To Detect Lines and Curves in Pictures. *Comm. Assoc. Comp. Machine*, 15:11–15.

[Edelsbrunner and Mücke, 1994] Edelsbrunner, H. and Mücke, E. (1994). Three-Dimensional Alpha Shapes. *ACM Transactions on Graphics*, 13(1):43–72.

[Efendic, 2008] Efendic, D. (2008). *Vergleich der 3D-Daten mittels Stereo Rekonstruktion und Laser Range Scanner*. Master Thesis, Vienna University of Technology.

[Ekvall and Kragic, 2007] Ekvall, S. and Kragic, D. (2007). Learning and Evaluation of the Approach Vector for Automatic Grasp Generation and Planning. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 4715–4720.

[El-Khoury et al., 2007] El-Khoury, S., Sahbani, A., and Perdereau, V. (2007). Learning the Natural Grasping Component of an Unknown Object. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems*, pages 2957–2962.

[Faber and Fischer, 2001a] Faber, P. and Fischer, R. (2001a). Euclidean Fitting Revisited. In *Proceedings of the Internatinoal Workshop on Visual Forms*, pages 165–175.

[Faber and Fischer, 2001b] Faber, P. and Fischer, R. (2001b). Pros and Cons of Euclidean Fitting. In *Proceedings of the Annual German Symposium for Pattern Recognition*, pages 414–420.

[Fagg and Arbib, 1998] Fagg, A. and Arbib, M. (1998). Modeling Parietal-Premotor Interactions in Primate Control of Grasping. *Neural Networks*, 11:1277–1303.

[Fan et al., 1989] Fan, T., Medioni, G., and Nevatia, R. (1989). Recognizing 3-D Objects Using Surface Descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1140–1157.

[Feddema and Little, 1997] Feddema, J. and Little, C. (1997). Rapid World Modeling: Fitting Range Data to Geometric Primitives. In *Proceedings of the IEEE Conference on Robotics and Automation*, volume 4, pages 2807–2812.

[Filsinger et al., 2003] Filsinger, J., Dittmann, R., and Bischoff, T. (2003). Nähen als textile Preform-Technik zur Herstellung von Faserverbundstrukturen für Luftfahrtanwendungen am Beispiel der A380-Druckkalotte. *DGLR-Kongress*.

[Fischler and Bolles, 1981] Fischler, M. and Bolles, R. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 26(6):381–395.

[Fitzgibbon and Fisher, 1995] Fitzgibbon, A. and Fisher, R. (1995). A Buyer's Guide to Conic Fitting. In *Proceedings of the British Machine Vision Conference*, pages 513–522.

[Funkhouser et al., 2004] Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., and Dobkin, D. (2004). Modeling by Example. *ACM Transactions on Graphics*, 23(3):652–663.

[Gärtner, 1999] Gärtner, B. (1999). Fast and Robust Smallest Enclosing Balls. In *Proceedings of 7^{th} Annual European Symposium on Algorithms (ESA)*, pages 325–338. Lecture Notes in Computer Science, Springer.

[Gershon, 1990] Gershon, D. (1990). Parallel Process Decomposition of a Dynamic Manipulation Task: Robotic Sewing. *IEEE Transactions on Robotics and Automation*, 6(3):357–367.

[Gershon and Porat, 1988] Gershon, D. and Porat, I. (1988). Vision Servo Control of a Robotic Sewing System. *Proceedings of the IEEE Conference on Robotics and Automation*, 3(6):1830–1835.

[Goldfeder et al., 2007] Goldfeder, C., Allen, P., Lackner, C., and Pelossof, R. (2007). Grasp Planning via Decomposition Trees. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 4679–4684.

[Gottschalk et al., 1996] Gottschalk, S., Lin, M., and Manocha, D. (1996). OBB-Tree: A Hierachical Structure for Rapid Interference Detection. In *Proceedings of SIGGRAPH*, pages 171–180.

[Gray et al., 1997] Gray, A., Abbena, E., and Salamon, S. (1997). *Modern Differential Geometry of Curves and Surfaces with Mathematica*. CRC Press, 3^{rd} edition.

[Grisoni and Marchal, 2003] Grisoni, L. and Marchal, D. (2003). High Performance generalized cylinders visualization. In *Proceedings of the Shape Modeling International*, pages 257–263. IEEE Computer Society.

[Gumhold et al., 2001] Gumhold, S., Wang, X., and MacLeod, R. (2001). Feature Extraction from Point Clouds. In *Proceedings of the 10^{th} International Meshing Roundtable*, pages 293–305.

[Hameiri and Shimshoni, 2003] Hameiri, E. and Shimshoni, I. (2003). Estimating the Principal Curvatures and the Darboux Frame from Real 3D Range Data. *IEEE Transactions on Systems, Man and Cybernetics*, 33(4):626–637.

[Haverinen and Röning, 2000] Haverinen, J. and Röning, J. (2000). A 3-D Scanner Capturing Range and Color: Design and Calibration. In *Proceedings of Vision Interfaces*, pages 378–381.

[Hirsch, 1997] Hirsch, M. (1997). *Differential Topology*. Springer Verlag.

[Hoover et al., 1995] Hoover, A., Jean-Baptiste, G., Jiang, X., Flynn, P., Bunke, H., Goldgof, D., and Bowyer, K. (1995). Range Image Segmentation: The Users Dilemma. In *Proceedings of the International Symposium on Computer Vision*, pages 323–328.

[Hoover et al., 1996] Hoover, A., Jean-Baptiste, G., Jiang, X., Flynn, P., Bunke, H., Goldgof, D., Bowyer, K., Eggert, D., Fitzgibbon, A., and Fisher, R. (1996). An Experimental Comparison of Range Image Segmentation Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):673–689.

[Hough, 1962] Hough, P. (1962). Method and means for recognizing complex patterns. *US Patent 3069654*.

[Huebner et al., 2008] Huebner, K., Ruthotto, S., and Kragic, D. (2008). Minimum Volume Bounding Box Decomposition for Shape Approximation in Robot Grasping. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1628–1633.

[Ivlev and Martens, 2005] Ivlev, O. and Martens, C. (2005). Rehabilitation Robots FRIEND-I and FRIEND-II with the dexterous lightweight manipulator. *Technology and Disability*, 17(2):111–123.

[Jackson et al., 1995] Jackson, S., Jackson, G., and Rosicky, J. (1995). Are non-relevant objects represented in working memory? The effect of non-target objects on reach and grasp kinematics. *Experimental Brain Research*, 102:519–530.

[Jaklic et al., 2000] Jaklic, A., Leonardis, A., and Solina, F. (2000). *Segmentation and Recovery of Superquadrics*. Kluwer Academic Publishers.

[Jiang and Bunke, 1999] Jiang, X. and Bunke, H. (1999). Edge Detection in Range Images Based on Scan Line Approximation. *Computer Vision and Image Understanding*, 73(2):183–199.

[Jiang and Cheng, 1999] Jiang, X. and Cheng, D. (1999). Fitting of 3D Circles and Ellipses Using a Parameter Decomposition Approach. pages 103–109.

[Johansson et al., 2001] Johansson, R., Westling, G., Bäckströ, A., and Flanagan, J. (2001). Eye–Hand Coordination in Object Manipulation. *The Journal of Neuroscience*, 21(17):6917–6932.

[Katsoulas and Werber, 2004] Katsoulas, D. and Werber, A. (2004). Edge Detection in Range Images of Piled Box-like Objects. volume 2, pages 80–84.

[Katz et al., 2005] Katz, S., Leifman, S., and Tal, A. (2005). Mesh segmentation using feature point and core extraction. *The Visual Computer (Pacific Graphics)*, 21(8–10):649–658.

[Katz and Tal, 2003] Katz, S. and Tal, A. (2003). Hierarchical Mesh Decomposition using Fuzzy Clustering and Cuts. *ACM Transactions on Graphics*, 22(3):954–961.

[Keren et al., 1994] Keren, D., Cooper, D., and Subrahmonia, J. (1994). Describing Complicated Objects by Implicit Polynomials. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):38–53.

[Kimmel and Sethian, 1998] Kimmel, R. and Sethian, A. (1998). Computing Geodesic Paths on Manifolds. In *Proceedings of National Academy of Sciences*, volume 95, pages 8431–8435.

[Kolluri et al., 2004] Kolluri, R., Shewchuk, J., and O'Brien, J. (2004). Spectral Surface Reconstruction from Noisy Point Clouds. In *Symposium on Geometry Processing*, pages 11–21.

[Kragic and Bjorkman, 2006] Kragic, D. and Bjorkman, M. (2006). Strategies for Object Manipulation using Foveal and Peripheral Vision. In *Proceedings of the IEEE Conference on Computer Vision Systems*, pages 50–55.

[Krockenberger and Nollek, 1991] Krockenberger, O. and Nollek, H. (1991). Handling within an Automatic Sewing Cell for Trouser Legs. *Conference on Advanced Robotics*, 2:1534–1537.

[Kuhtz-Buschbeck et al., 1998] Kuhtz-Buschbeck, J., Stolze, H., Jöhnk, K., Boczek-Funcke, A., and Illert, M. (1998). Development of prehension movements in children: a kinematic study. *Experimental Brain Research*, 122(4):424–432.

[Lai et al., 2008] Lai, Y.-K., Hu, S.-M., Martin, R., and Rosin, P. (2008). Fast Mesh Segmentation using Random Walks. In *Symposium on Solid and Physical Modeling*, pages 183–191.

[Lawson and Hanson, 1974] Lawson, C. and Hanson, R. (1974). *Solving Least Squares Problems*. New York: Prentice-Hall, ISBN: 0-89871-356-0.

[Lee and Park, 1990] Lee, Y. and Park, S. (1990). A Study of Convex/Concave Edges and Edge-Enhancing Operators Based on the Laplacian. *IEEE Transactions on Circuits and Systems*, 37(7):940–946.

[Levenberg, 1944] Levenberg, K. (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. In *Quarterly of Applied Mathematics 2*, pages 164–168.

[Li et al., 2004] Li, W., Krehl, J., Jr., J. G., Heider, D., Endrulat, M., Hochrein, K., Dunham, M., and Dubois, C. (2004). Process and Performance Evaluation of the Vacuum-Assisted Process. *Journal of Composite Materials*, 38(20):1803–1814.

[Li et al., 2001] Li, X., Toon, T., Tan, T., and Huang, Z. (2001). Decomposing Polygon Meshes for Interactive Applications. In *Proceedings of the Symposium on Interactive 3D Graphics*, pages 35–42.

[Li et al., 2007] Li, Y., Fu, J., and Pollard, N. (2007). Data-Driven Grasp Synthesis Using Shape Matching and Task-Based Pruning. *IEEE Transactions on Visualization and Computer Graphics*, 13(4):732–747.

[Lukacs et al., 1998] Lukacs, G., Martin, R., and Marshall, D. (1998). Faithful Least-Squares Fitting of Spheres, Cylinders, Cones and Tori for Reliable Segmentation. In *Proceedings of the European Conference on Computer Vision*, volume 1, pages 671–686.

[Marquardt, 1963] Marquardt, D. (1963). An Algorithm for Least-Squares Estimation of Nonlinear Parameters. In *Journal of the Society for Industrial and Applied Mathematics*, pages 431–441.

[Marr and Hildreth, 1980] Marr, D. and Hildreth, E. (1980). Theory of Edge Detection. *Proceedings of the Royal Society of London, Series B*, 207(1167):187–217.

[Marshall et al., 2001] Marshall, D., Lukacs, G., and Martin, R. (2001). Robust Segmentation of Primitives from Range Data in the Presence of Geometric Degeneracy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):304–314.

[Martens and Ruchel, 2001] Martens, C. and Ruchel, N. (2001). A FRIEND for Assisting Handicapped People. *IEEE Robotics and Automation Magazine*, 8(1):57–65.

[Miller and Allen, 2004] Miller, A. and Allen, P. (2004). GraspIt! A Versatile Simulator for Robotic Grasping. *IEEE Robotics and Automation Magazine*, 11(4):110–112.

[Miller and Knoop, 2003] Miller, A. and Knoop, S. (2003). Automatic Grasp Planning Using Shape Primitives. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1824–1829.

[Mohan and Nevatia, 1989] Mohan, R. and Nevatia, R. (1989). Using Perceptual Organization to Extract 3-D Structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1121–1139.

[Mosby, 2009] Mosby, W. (2009). *Mosby's Medical Dictionary.* Elsevier Health Sciences, 8^{th} edition.

[Mühlmann et al., 2002] Mühlmann, K., Maier, D., Hesser, J., and Männer, R. (2002). Calculating Dense Disparity Maps from Color Stereo Images, an Efficient Implementation. *International Journal of Computer Vision*, 47(1–3):79–88.

[Murphy, 1998] Murphy, J. (1998). *Reinforced Plastics Handbook.* Elsevier Advanced Technology.

[Nakagawa et al., 2008] Nakagawa, T., Baldauf, D., and Deubel, H. (2008). Gaze Behavior in Grasping of Natural Objects. In *NCP research report, LMU.*

[Nocks, 2007] Nocks, L. (2007). *The Robot: The Life Story of a Technology.* Greenwood Publishing Group.

[Nössing, 2004] Nössing, M. (2004). *Tiefenbildaufnahme nach dem Lichtschnittverfahren mittels Dreh/Schwenkeinheit.* Master Thesis, Vienna University of Technology.

[O'Rourke, 1998] O'Rourke, J. (1998). *Computational Geometry in C.* Univ. Press, Cambridge, 2^{nd} edition.

[Pan et al., 2005] Pan, C., Hongping, Y., Medioni, G., and Songde, M. (2005). Parametric Reconstruction of Generalized Cylinders From Limb Edges. *IEEE Transactions on Image Processing*, 14(8):1202–1214.

[Pillow et al., 1994] Pillow, N., Utcke, S., and Zisserman, A. (1994). Viewpoint-Invariant Representation of Generalized Cylinders Using the Symmetry Set. In *Proceedings of the British Machine Vision Conference*, pages 539–548.

[Ponce et al., 1989] Ponce, J., Chelberg, D., and Mann, W. (1989). Invariant Properties of Straight Homogeneous Generalized Cylinders and Their Contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(9):951–966.

[Porteous, 1994] Porteous, I. (1994). *Geometric Differentiation.* Univ. Press, Cambridge.

[Pottmann et al., 2004] Pottmann, H., Leopoldseder, S., and Hofer, M. (2004). Registration without ICP. *Computer Vision and Image Understanding*, 95(1):54–71.

[Pritschow et al., 2002] Pritschow, G., Mueller, S., and Hober, H. (2002). Fast and robust image processing for laser stripe-sensors in arc welding automation. *Symposium on Industrial Electronics (ISIE)*, 2:651–656.

[Rand, 1971] Rand, W. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850.

[Rao and Nevatia, 1988] Rao, K. and Nevatia, R. (1988). Computing Volume Descriptions from Sparse 3-D Data. *Springer; International Journal of Computer Vision*, 2(1):33–50.

[Recatalà et al., 2008] Recatalà, G., Del Pobil, E. C. Á., Mezouar, Y., and Martinet, P. (2008). Biologically-inspired 3D grasp synthesis based on visual exploration. *IEEE Autonomous Robots*, 25(1–2):59–70.

[Richtsfeld et al., 2007a] Richtsfeld, M., Biegelbauer, G., Wohlkinger, W., and Vincze, M. (2007a). Sensor-Based Robotics for Fully Automated Fibre-Mat Stitching. In *Proceedings of the European Control Conference*, pages 2051–2058.

[Richtsfeld et al., 2008] Richtsfeld, M., Ponweiser, W., and Vincze, M. (2008). Real Time Grasping of Freely Placed Cylindrical Objects. In *Proceedings of the* 5^{th} *International Conference on Informatics in Control, Automation and Robotics*, pages 165–170.

[Richtsfeld and Vincze, 2008a] Richtsfeld, M. and Vincze, M. (2008a). Detection of Grasping Points of Unknown Objects Based on $2^1/_2$D Point Clouds. In *Proceedings of the* 19^{th} *International Conference of Pattern Recognition*, pages 1–4.

[Richtsfeld and Vincze, 2008b] Richtsfeld, M. and Vincze, M. (2008b). Grasping of Unknown Objects from a Table Top. In *Proceedings of the ECCV 2008 Workshop on 'Vision in Action: Efficient strategies for cognitive agents in complex environments'*, pages 1–11.

[Richtsfeld and Vincze, 2008c] Richtsfeld, M. and Vincze, M. (2008c). Robotic Grasping of Unknown Objects using $2^1/_2$D Point Clouds. In *Proceedings of the* 17^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 1–6.

[Richtsfeld and Vincze, 2009a] Richtsfeld, M. and Vincze, M. (2009a). Point Cloud Segmentation Based on Radial Reflection. In *Proceedings of the* 13^{th} *International Conference on Computer Analysis of Images and Patterns*, pages 955–962.

[Richtsfeld and Vincze, 2009b] Richtsfeld, M. and Vincze, M. (2009b). Robotic Grasping from a Single View. In *Proceedings of the* 18^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 1–6.

[Richtsfeld et al., 2007b] Richtsfeld, M., Wohlkinger, W., Biegelbauer, G., and Vincze, M. (2007b). Real-Time Laser Profile Edge Detection by Voting. In *Proceedings of the* 31^{st} *Annual Workshop of the Austrian Association for Pattern Recognition*, pages 137–144.

[Richtsfeld et al., 2007c] Richtsfeld, M., Wohlkinger, W., Biegelbauer, G., and Vincze, M. (2007c). Robust Edge Tracking for an Automated Robot Sewing Process Using a Laser Range Scanner. In *Proceedings of the* 16^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 222–227.

[Richtsfeld and Zillich, 2008] Richtsfeld, M. and Zillich, M. (2008). Grasping Unknown Objects Based on $2^1/_2$D Range Data. In *Proceedings of IEEE Conference on Automation Science and Engineering*, pages 691–696.

[Roberts, 1965] Roberts, L. (1965). Machine Perception of Three-Dimensional Solids. In *Optical and Electro-Optical Information Processing*, pages 159–197. MA: MIT Press.

[Rosman et al., 2008] Rosman, G., Bronstein, M., Bronstein, A., and Kimmel, R. (2008). Topologically Constrained Isometric Embedding. In *Human Motion - Understanding, Modeling, Capture and Animation, Computational Imaging and Vision*, volume 36, pages 1–10. Springer.

[Sanz et al., 1999] Sanz, P., Iñesta, J., and Del Pobil, Á. (1999). Planar Grasping Characterization Based on Curvature-Symmetry Fusion. *Applied Intelligence*, 10(1):25–36.

[Saxena et al., 2008] Saxena, A., Driemeyer, J., and Ng, A. (2008). Robotic Grasping of Novel Objects using Vision. *International Journal of Robotics Research*, 27(2):157–173.

[Scharstein and Szeliski, 2002] Scharstein, D. and Szeliski, R. (2002). A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47(1–3):7–42.

[Schneiberg et al., 2002] Schneiberg, S., McFadyen, B., Sveistrup, H., Mckinley, P., and Levin, M. (2002). The development of coordination for reach to grasp movements in children. *Experimental Brain Research*, 146(2):142–154.

[Schneider and Eberly, 2003] Schneider, P. and Eberly, D. (2003). *Geometric Tools for Computer Graphics*. The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, Elsevier Science.

[Schöffmann, 2008] Schöffmann, A. (2008). *Evaluation and Implementation of Improved Edge Detection Methods for an Industrial Robot Sewing Process*. Master Thesis, Vienna University of Technology.

[Schulz et al., 2005] Schulz, S., Pylatiuk, C., Reischl, M., Martin, J., Mikut, R., and Bretthauer, G. (2005). A hydraulically driven multifunctional prosthetic hand. *Robotica*, 23(3):293–299.

[Shamir et al., 2004] Shamir, A., Shapira, L., Cohen-Or, D., and Goldenthal, R. (2004). Geodesic Mean Shift. In *Proceedings of the* 5^{th} *Korea-Israel Conference on Geometric Modeling and Computer Graphics*, pages 51–56.

[Shapira et al., 2008] Shapira, L., Shamir, A., and Cohen-Or, D. (2008). Consistent Mesh Partitioning and Skeletonisation using the Shape Diameter Function. *The Visual Computer: International Journal of Computer Graphics*, 24(4):249–259.

[Shipley and Kellman, 2001] Shipley, T. and Kellman, P. (2001). *From Fragments to Objects: Segmentation and Grouping in Vision*, volume 130. Advances in Psychology Series, New York: Elsevier Science B.V.

[Shlafman et al., 2002] Shlafman, S., Tal, A., and Katz, S. (2002). Metamorphosis of Polyhedral Surfaces using Decomposition. *Computer Graphics Forum*, 21(3):219–229.

[Sickinger and Hermann, 2001] Sickinger, C. and Hermann, A. (2001). Structural Stitching as a Method to design High-Performance Composites in Future. In *Proceedings TechTextil Symposium*.

[Sim et al., 2002] Sim, T., Hong, G., and Lim, K. (2002). A Pragmatic 3D Visual Servoing System. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 4185–4190.

[Smeets and Brenner, 1999] Smeets, J. and Brenner, E. (1999). A New View on Grasping. *Motor Control*, 3:237–271.

[Speth et al., 2008] Speth, J., Morales, A., and Sanz, P. (2008). Vision-based Grasp Planning of 3D Objects by Extending 2D Contour Based Algorithms. In *Proceedings of the IEEE Conference on Intelligent Robots and Systems*, pages 2240–2245.

[Spira and Kimmel, 2004] Spira, A. and Kimmel, R. (2004). An Efficient Solution to the Eikonal Equation on Parametric Manifolds. *Interfaces and Free Boundaries*, 6(3):315–327.

[Stansfield, 1990] Stansfield, S. (1990). Knowledge-Based Robotic Grasping. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1270–1275.

[Steinhilber, 2006] Steinhilber, M. (2006). *Entwicklung eines Prüfverfahrens zur Bewertung von Einflüssen nähtechnischer Parameter auf die mechanischen Eigenschaften konsolidierter CFK-Strukturen*. Master Thesis, Reutlingen University / EADS Deutschland GmbH.

[Stiene et al., 2006] Stiene, S., Lingemann, K., Nüchter, A., and Hertzberg, J. (2006). Contour-based Object Detection in Range Images. In *Proceedings of the 3^{rd} International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 168–175.

[Suetens et al., 1992] Suetens, P., Fua, P., and Hanson, A. (1992). Computational Strategies for Object Recognition. *ACM Computing Surveys*, 24(1):5–61.

[Surazhsky et al., 2003] Surazhsky, T., Magid, E., Soldea, O., Elber, G., and Rivlin, E. (2003). A Comparison of Gaussian and Mean Curvatures Estimation Methods on Triangular Meshes. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 1021–1026.

[Taubin, 1991] Taubin, G. (1991). Estimation of Planar Curves, Surfaces and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1115–1138.

[Taubin, 1993] Taubin, G. (1993). An Improved Algorithm for Algebraic Curve and Surface Fitting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 658–665.

[Taylor and Kleeman, 2002] Taylor, G. and Kleeman, L. (2002). Grasping Unknown Objects with a Humanoid Robot. In *Proceedings of the Australian Conference on Robotic and Automation*, pages 191–196.

[Taylor and Kleeman, 2003] Taylor, G. and Kleeman, L. (2003). Robust Range Data Segmentation Using Geometric Primitives for Robotic Applications. In *Proceedings of the 9^{th} International Conference on Signal and Image Processing*, pages 467–472.

[Tresilian, 1998] Tresilian, J. (1998). Attention in action or obstruction of movement? A kinematic analysis of avoidance behavior in prehension. *Experimental Brain Research*, 120(3):352–368.

[Tresilian, 1999] Tresilian, J. (1999). Selective attention in reaching: when is an object not a distractor? *Trends in Cognitive Sciences*, 3(11):407–408.

[Trucco and Fisher, 1995] Trucco, E. and Fisher, R. (1995). Experiments in Curvature-Based Segmentation of Range Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):177–182.

[Trucco and Verri, 1998] Trucco, E. and Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall, ISBN: 0-13-261108-2.

[von Hofsten, 1982] von Hofsten, C. (1982). Eye-Hand Coordination in the Newborn. *Developmental Psychology*, 18(3):450–461.

[Wang and Jiang, 2005] Wang, B. and Jiang, L. (2005). Grasping Unknown Objects Based on 3D Model Reconstruction. In *Proceedings of International Conference on Advanced Intelligent Mechatronics*, pages 461–466.

[Williams et al., 1997] Williams, J., Johnstone, J., and Wolff, L. (1997). Rational Discrete Generalized Cylinders and their Application to Shape Recovery in Medical Images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 387–392.

[Witting, 2001] Witting, J. (2001). Recent Development in the Robotic Stitching Technonology for Textile Structural Composites. *Journal of Textile and Apparel (JTATM)*, 2(1).

[Wohlkinger, 2007] Wohlkinger, W. (2007). *Robust Edge-Tracking for Industrial Stitching of Carbon Fibre Mats Using a Laser Range Scanner*. Master Thesis, Vienna University of Technology.

[Xu et al., 1992] Xu, L., Oja, E., and Suen, C. (1992). Modified Hebian Learning for Curve and Surface Fitting. *Elsevier; Neural Networks*, 5(3):441–457.

[Xue et al., 2008] Xue, Z., Zoellner, J., and Dillmann, R. (2008). Automatic Optimal Grasp Planning Based On Found Contact Points. In *Proceedings of the IEEE/ASME Conference on Advanced Intelligent Mechatronics*, pages 1053–1058.

[Yacoub and Menard, 1997] Yacoub, S. and Menard, C. (1997). Robust Axis Determination for Rotational Symmetric Objects out of Range Data. In *Proceedings of the* 21^{st} *Workshop of the Austrian Association for Pattern Recognition*, pages 197–201. ISBN: 3-7029-0436-0.

[Yokoya and Levine, 1989] Yokoya, N. and Levine, M. (1989). Range Image Segmentation Based on Differential Geometry: A Hybrid Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):643–649.

[Zerroug, 1999] Zerroug, M. (1999). Part-Based 3D Descriptions of Complex Objects from a Single Image. 21(9):835–848.

[Zerrough and Nevatia, 1999] Zerrough, M. and Nevatia, R. (1999). Part-Based 3D Descriptions of Complex Objects from a Single Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):835–848.

[Zhang et al., 2007] Zhang, H., Kaick, O., and Dyer, R. (2007). Spectral Methods for Mesh Processing and Analysis. In *Proceedings of Eurographics*, pages 1–22.

[Zhang et al., 1994] Zhang, X., Nakamura, Y., Goda, K., and Yoshimoto, K. (1994). Robustness of Power Grasp. In *Proceedings of the IEEE Conference on Robotics and Automation*, pages 2828–2835.

Curriculum Vitae

Personal Information

First name / Surname:	Mario Richtsfeld
Academic degree:	Dipl.-Ing.(FH)
Nationality:	Austrian
Date Of Birth:	16/05/1979, Scheibbs, Austria
Gender:	Male
Address:	Grestnerstrae 26/3/1, 3250 Wieselburg, Austria
Languages:	German (mother tongue), English, Spanish (basics)

Education

Since Oct. 2005:	Doctorate (PhD) studies of Electrical Engineering at Vienna University of Technology, Austria Specialisation: Robotics and Range Image Processing
2003 - 2005:	Post-graduate studies for the Ph.D. at Institute of Automation and Control at Vienna University of Technology, Austria
2003:	Graduation (with distinction) as Dipl.-Ing.(FH) of Automation and Control at the University of Applied Sciences, Wels, Austria Master thesis carried out at Fraunhofer Institute (IPA), Stuttgart
1999 - 2003:	Diploma studies of Automation and Control at University of Applied Sciences (FH), Wels, Austria
1999:	Military service
1994 - 1998:	Technical college (Automation Engineering), St.Pölten Graduation
1993 - 1994:	Technical college (Aircraft Manufacturing), Langenlebarn
1989 - 1993:	Secondary school, Wieselburg
1985 - 1989:	Elementary school, Wieselburg

Additional Experiences and Competencies

- REFA basic course 1 and 2

- QA certificate
- Junior Project Manager (IPMA Level D)

Work Experience

Since Feb. 2006:	Research assistant at the Institute of Automation and Control at Vienna University of Technology
2002 - 2003:	6 months internship at Institute Fraunhofer (IPA), Stuttgart/Germany Area of Research: Intelligent Tools Project: Team@Work
2000:	2 months internship, logistics at Audi, Ingolstadt/Germany
1994 - 2005:	different internships, (Umdasch AG, Doka GmbH, ZIZALA Lichtsysteme GmbH, Buntmetall GesmbH,...)

Project Work

Since Mar. 2008:	Research project GRASP (grant agreement number 215821)
Feb. 2006 - Dec. 2008:	Research project REDUX (FFG, 810568/1553 SCK/SAI)

Publications

Articles in Journals:

Vincze, M., Richtsfeld, M., and Biegelbauer, G. (2008). Lösungen zur Fertigung kleiner Stückzahlen in vielen Varianten. *Elektrotechnik und Informationstechnik (e & i)*, vol. 1, no. 2, pages 33–41.

Proceedings of Conferences:

Richtsfeld, M., and Vincze, M. (2009). Point Cloud Segmentation Based on Radial Reflection. In *Proceedings of the* 13^{th} *International Conference on Computer Analysis of Images and Patterns*, pages 955–962.

Richtsfeld, M., and Vincze, M. (2009). Robotic Grasping from a Single View. In *Proceedings of the* 18^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 1–6.

Richtsfeld, M., and Vincze, M. (2008). Detection of Grasping Points of Unknown Objects Based on 2.5D Point Clouds. In *Proceedings of the* 19^{th} *International Conference of Pattern Recognition*, pages 1–4.

Richtsfeld, M., and Vincze, M. (2008). Grasping of Unknown Objects from a Table Top. In *Proceedings of the ECCV 2008 Workshop on 'Vision in Action: Efficient strategies for cognitive agents in complex environments'*, pages 1–11.

Richtsfeld, M., and Zillich, M. (2008). Grasping Unknown Objects Based on 2.5D Range Data. In *Proceedings of the* 4^{th} *IEEE Conference on Automation Science and Engineering*, pp. 691–696.

Richtsfeld, M., and Vincze, M. (2008). Robotic Grasping of Unknown Objects using 2.5D Point Clouds. In *Proceedings of the* 17^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 1–6.

Richtsfeld, M., Ponweiser, W., and Vincze, M. (2008). Real-Time Grasping of Freely Placed Cylindrical Objects. In *Proceedings of the* 5^{th} *International Conference of Informatics in Control, Automation and Robotics*, pages 165–170.

Richtsfeld, M., Wohlkinger, W., Biegelbauer, G., and Vincze, M. (2007). Robust Edge Tracking for an Automated Robot Sewing Process using a Laser Range Scanner. In *Proceedings of the* 16^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 222–227.

Gemeiner, P., Richtsfeld, M., Ponweiser, W., Einramhof, P., and Vincze, M. (2007). Pros and Cons of a High-Speed CMOS Camera for Slam. In *Proceedings of the* 16^{th} *International Workshop on Robotics in Alpe-Adria-Danube Region*, pages 229-303.

Richtsfeld, M., Biegelbauer, G., Wohlkinger, W., and Vincze, M. (2007). Sensor-Based Robotics for Fully Automated Fibre-Mat Stitching. In *Proceedings of the European Control Conference*, pages 2051–2058.

Richtsfeld, M., Wohlkinger, W., Biegelbauer, G., and Vincze, M. (2007). Real-Time Laser Profile Edge Detection by Voting. In *Proceedings of the* 31^{st} *Annual Workshop of the Austrian Association for Pattern Recognition*, pages 137–144.

Biegelbauer, G., Richtsfeld, M., Wohlkinger, W., Vincze, M., and Herkt, M. (2007). Optical Seam Following for Automated Robot Sewing. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4758–4763.

Chapters in Books:

Richtsfeld, M., and Vincze, M. (2009). Robotic Grasping of Unknown Objects. *Robotics, Automation and Control*, In-Tech Education and Publishing, Vienna, ISBN: 978-953-7619-39-8 (ebook), 17 pages.