

Lehrstuhl für Steuerungs- und Regelungstechnik
Technische Universität München
Prof. Dr.-Ing./Univ. Tokio Martin Buss

Probabilistic Grasping for Mobile Manipulation Systems: Skills, Synthesis and Control

Dong Chen

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender: Prof. Dr.-Ing. Werner Hemmert

Prüfer der Dissertation:

1. Prof. Dr.-Ing./Univ. Tokio Martin Buss
2. Prof. Dr.-Ing. Tamim Asfour

Die Dissertation wurde am 11.06.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 21.04.2020 angenommen.

Foreword

This dissertation summarizes the work that I conducted at *Robotics, Autonomous System and Control* (RAC) group of Siemens Corporate Technology (CT). I gratefully acknowledge the generous financial support of Siemens AG.

First of all, I would like to thank my doctoral supervisor, Dr. Georg von Wichert, for your supervision and tremendous support during my entire Ph.D. work. You let me find my research direction and makes me believe in myself. Several times, you generously sacrifice your private time to help me revise my paper until the midnight. I really feel so lucky to be your student. I also would like to thank Prof. Dr. -Ing./Univ. Martin Buss, who gives me the opportunity to finish and defend my thesis at Institute of Automatic Control Engineering of TU München. I also sincerely thank Dr. Gisbert Lawitzky, who provides me the chance to work in this team and ‘SIR’, an amazing mobile manipulation robot which accompanies me during the past joyful years.

I would like to thank every member in the RAC team. Their specializations in each domain of robotics provide me a broad view of the fantastic world of robotics as well as enlighten me in many aspects of my research work. I enjoy every lunch, every coffee round and every ‘Mannerabend’ with them. Special thanks go to Ziyuan Liu, Vincent Dietrich, Bernd Kast, Florian Wirnshofer, Philip Koehler, Philipp Schmitt and Konstantin Ritt, for their fruitful discussions, critical reviews on each paper and great contributions to this dissertation. I am so happy to meet these amazing people in my life and enjoy every moment of solving challenging technical problems with them. Great thanks also go to Dr. Thomas Wösch, Dr. Wendelin Feiten and Dr. Kai Wurm for their immense experience, invaluable advice, and technical support.

Finally, I would like to thank my wife Zijia Bai and my parents, who take care of my little son, give me unconditional love, endless patience and enormous encouragement. Any word can not be enough to express my gratitude to them.

I wish all the best to all of them.

Munich, December 2016

Dong Chen

This work is accomplished with the support from Research group of *Robotics, Autonomous Systems and Control*, funded by Siemens AG

SIEMENS

to Zijia and Yibo

...

Abstract

Grasping is one of the most fundamental skills of manipulation. Mastering grasping is a precondition of performing complex manipulation tasks for robots. Although robotic grasping has been studied over a long period, there still remains a big gap when compared to the capability of human grasping. Especially in an unstructured environment, the robot is confronted with many challenges. First, grasping is usually related to a task intention. For example, how to select a suitable grasp configuration to satisfy a task constraint is not trivial. In addition, due to the variety of objects existing in the world, it is unfeasible and impracticable to model every individual object a-priori. Thereafter, grasping objects which are presented to robots for the first time is another task that must be tackled. Moreover, the imperfection of sensors used for grasping generate inaccurate measurements. Focus must be undertaken with regards to dealing with the uncertainty of a perceptual system for grasping. People may ask: why is grasping such a difficult task for robots when humans can master the grasping skill at such an early age? One must accept that grasping is comprised of an interdisciplinary set of skills. These include the requirement to perceive the surrounding environment plus objects, decide upon a feasible grasping strategy and finally execute the control of the grasping process.

In this dissertation, an integrated system for grasping is proposed. The system regards grasping as a dynamic process, from the state an object is perceived to the state the object is grasped. A mobile manipulator which uses the proposed system is able to perform robust grasping under various levels of prior knowledge and conditions. The prior knowledge and conditions determine the major problem to be solved in a specific grasp scenario.

I propose to represent the diversity of various grasping scenarios under a unified model which is defined by 'Bayesian Network'. In this way, the grasping problem is approached in a probabilistic fashion. The model defines a joint distribution of grasping relevant factors, while the uncertainty of these factors is quantified by probabilistic distributions. Three challenging grasping scenarios under predefined conditions are formulated by the proposed model. The difference between the scenarios is encoded by conditioning respective factors in this model.

In the first scenario, grasping is addressed in the context of assembly tasks. The major problem is to select grasp configurations which maximize the probability of satisfying task constraints. A skill framework is proposed to generate a complete sequence for a mobile manipulator to perform an assembly. This ranges from object perception, to the reasoning of grasp strategy to the control of robot motions. The framework provides an intuitive way for a non-expert user to parameterize an assembly problem. Within the framework, high level skills are proposed to perform pick-and-place actions to alter the orientation of objects as well as insertion actions for assembling two objects. Successful execution of both tasks requires that the objects should be grasped in an assignment preferred orientation. Different from previous works which only consider gripper orientation to fulfill this requirement, my method is able to generate complete robot configurations for the purpose

of reducing both the total traveling distance and the task execution time. This combined with modeling arm-platform coordination in the framework, further increase the efficiency and robustness of executing the given task.

In the second scenario, grasping of unknown objects is addressed. Here the challenge lies in synthesizing a grasp configuration which maximizes the probability of force closure. For this purpose, a new probabilistic object representation plus sensor fusion method is applied to reconstruct the shape of objects prior to grasping. This approach is especially suitable for modeling objects with irregular forms. Different from previous works which use other representations, my proposal can model the reconstruction uncertainty based on the noise model of sensors. In this way, this representation has a correct interpretation of the modeling uncertainty and can be used to penalize grasp points on uncertain areas. Utilizing this form of representation, a grasp synthesis method based on simulated annealing is proposed to search for an appropriate grasp configuration. Compared with the state-of-the-art method that uses a similar representation, my method works on 3D objects and is more computationally efficient.

In the third scenario, a sensor with a large uncertainty is used to study grasping performance. The major challenge is how to perform robust grasping even when given a large perceptual uncertainty. Different from previous approaches which usually execute grasp in open loop, an adaptive control architecture is proposed. It allows a robot to perform grasping in a closed-loop fashion. Using the proposed architecture, the perceptual result is continuously fed back to the grasp synthesis module in which current grasp configurations are generated. Motion adaptation to the new grasp configuration is achieved by Dynamic Movement Primitives (DMP) such that during the approaching phase, uncertainty of perception is actively reduced. Actuators of a mobile manipulator can be combined gracefully and used purposefully in different phases to improve the overall grasping process.

This research lays the foundation of how to approach the grasping problem in a probabilistic and systematic fashion. When compared to previous works which often consider one grasping phase, this work provides an end-to-end solution and closes the perception-action loop for the entire grasping process. My work provides the infrastructure for future research directions and paves the way toward realizing human-level dexterous grasping.

Zusammenfassung

Greifen ist eine der grundlegenden Fähigkeiten der Manipulation. Das Beherrschen des Greifens ist eine Voraussetzung für die Durchführung komplexer Manipulationsaufgaben für Robotern. Obwohl das Greifen von Robotern über einen langen Zeitraum hinweg untersucht wurde, bleibt dennoch eine große Lücke im Vergleich zu der Fähigkeit des menschlichen Greifens bestehen. Gerade in einer unstrukturierten Umgebung ist der Roboter mit vielen Herausforderungen konfrontiert. Erstens, Greifen bezieht sich normalerweise auf eine Aufgabenabsicht. Zum Beispiel ist es nicht trivial, eine geeignete Griffkonfiguration auszuwählen, um eine Aufgabenbeschränkung zu erfüllen. Aufgrund der Vielzahl von Objekten, die auf der Welt existieren, ist es außerdem unmöglich und unpraktikabel, jedes einzelne Objekt a priori zu modellieren. Danach ist das Ergreifen von Objekten, die zum ersten Mal einem Roboter präsentiert werden, eine weitere Aufgabe, die angegangen werden muss. Darüber hinaus erzeugt die Unvollkommenheit von Sensoren, die zum Erfassen verwendet werden, ungenaue Messungen. Der Fokus muss auf den Umgang mit der Unsicherheit eines Wahrnehmungssystems für das Greifen gelegt werden. Die Leute mögen fragen: Warum fällt Roboter beim Greifen schwer, wenn die Menschen in einem so frühen Alter die Fähigkeit des Greifens beherrschen? Man muss akzeptieren, dass das Greifen interdisziplinär ist. Dazu gehört die Anforderung, die Umgebung und Objekte wahrzunehmen, über eine durchführbare Greifstrategie zu entscheiden und schließlich die Steuerung des Greifprozesses durchzuführen.

In dieser Dissertation wird ein integriertes Greifsystem vorgeschlagen. Das System betrachtet das Greifen als einen dynamischen Prozess, vom Zustand, in dem ein Objekt wahrgenommen wird, bis zum Zustand, in dem das Objekt erfasst wird. Ein mobiler Manipulator, der das vorgeschlagene System verwendet, ist in der Lage, robustes Greifen unter verschiedenen Ebenen von Vorwissen und Bedingungen durchzuführen. Das Vorwissen und die Vorbedingungen bestimmen das Hauptproblem, das in einem spezifischen Griffszenario gelöst werden muss.

Ich schlage vor, die Vielfalt verschiedener Erfassungsszenarien in einem vereinheitlichten Modell darzustellen, das durch das Bayes'sche Netzwerk definiert wird. Auf diese Weise wird das Erfassungsproblem probabilistisch angegangen. Das Modell definiert eine gemeinsame Verteilung von greifrelevanten Faktoren, während die Unsicherheit dieser Faktoren durch probabilistische Verteilungen quantifiziert wird. Mit dem vorgeschlagenen Modell werden drei herausfordernde Erfassungsszenarien unter vordefinierten Bedingungen formuliert. Der Unterschied zwischen den Szenarien wird durch Konditionieren entsprechender Faktoren in diesem Modell codiert.

Im ersten Szenario wird das Greifen im Zusammenhang mit Montageaufgaben behandelt. Das Hauptproblem besteht darin, Greifkonfigurationen so auszuwählen, dass die Wahrscheinlichkeit, Aufgabenbeschränkungen zu erfüllen, maximiert wird. Ein Qualifika-

tionsrahmen wird vorgeschlagen, um eine vollständige Sequenz für einen mobilen Manipulator zu erzeugen, um eine Assembly durchzuführen. Dies reicht von der Objektwahrnehmung über die Begründung der Greifstrategie bis hin zur Steuerung von Roboterbewegungen. Das Framework bietet einem nicht sachkundigen Benutzer eine intuitive Möglichkeit, ein Montageproblem zu parametrisieren. Innerhalb des Rahmens werden Fähigkeiten auf hohem Niveau vorgeschlagen, um Pick-and-Place-Aktionen durchzuführen, um die Ausrichtung von Objekten sowie Einfügungsaktionen zum Zusammensetzen von zwei Objekten zu ändern. Die erfolgreiche Ausführung beider Aufgaben erfordert, dass die Objekte in einer bevorzugten Zuweisungsausrichtung erfasst werden. Anders als bei früheren Arbeiten, bei denen nur die Ausrichtung der Greifer berücksichtigt wird, um diese Anforderung zu erfüllen, ist meine Methode in der Lage, komplette Roboterkonfigurationen zu generieren, um sowohl die Gesamtfahrstrecke als auch die Ausführungszeit der Aufgabe zu reduzieren. Dies kombiniert mit einer Modellierung der Arm-Plattform-Koordination in dem Rahmenwerk, erhöht weiter die Effizienz und Robustheit der Ausführung der gegebenen Aufgabe.

Im zweiten Szenario wird das Erfassen unbekannter Objekte angesprochen. Hier besteht die Herausforderung darin, eine Griffkonfiguration zu synthetisieren, die die Wahrscheinlichkeit des Schließens der Kraft maximiert. Zu diesem Zweck wird eine neue probabilistische Objektrepräsentation plus Sensorfusionsmethode angewendet, um die Form von Objekten vor dem Ergreifen zu rekonstruieren. Dieser Ansatz eignet sich besonders für die Modellierung von Objekten mit unregelmäßigen Formen. Anders als bei früheren Arbeiten, die andere Darstellungen verwenden, kann meine Methode die Rekonstruktionsunsicherheit basierend auf dem Rauschmodell von Sensoren modellieren. Auf diese Weise hat diese Darstellung eine korrekte Interpretation der Modellierungsunsicherheit und kann verwendet werden, um Griffpunkte in unsicheren Bereichen zu bestrafen. Unter Verwendung dieser Darstellungsform wird ein auf synthetischem Tempern beruhendes Greifsyntheseverfahren vorgeschlagen, um nach einer geeigneten Griffkonfiguration zu suchen. Verglichen mit der Methode nach dem Stand der Technik, die eine ähnliche Darstellung verwendet, arbeitet meine Methode an 3D-Objekten und ist recheneffizienter.

Im dritten Szenario wird ein Sensor mit einer großen Unsicherheit verwendet, um die Greifleistung zu untersuchen. Die große Herausforderung besteht darin, robustes Greifen auch bei großen Wahrnehmungsunsicherheiten durchzuführen. Anders als bei früheren Ansätzen, bei denen normalerweise ein Zugriff in offener Schleife ausgeführt wird, wird eine adaptive Steuerungsarchitektur vorgeschlagen. Es ermöglicht einem Roboter, das Greifen in einer geschlossenen Schleife durchzuführen. Unter Verwendung der vorgeschlagenen Architektur wird das Wahrnehmungsergebnis kontinuierlich an das Griffsynthesemodul zurückgegeben und die aktuelle Griffkonfiguration angepasst werden. Die Anpassung der Bewegung an die neue Griffkonfiguration wird durch Dynamic Movement Primitives (DMP) generiert, so dass während der Annäherungsphase die Unsicherheit der Wahrnehmung aktiv reduziert wird. Aktuatoren eines mobilen Manipulators können sinnvoll kombiniert und gezielt in verschiedenen Phasen eingesetzt werden, um den gesamten

Greifprozess zu verbessern.

Diese Forschung legt den Grundstein dafür, wie das Greifenproblem probabilistisch und systematisch behandelt werden kann. Im Vergleich zu früheren Arbeiten, die oft nur eine Greifphase betrachtet, bietet diese Arbeit eine End-to-End-Lösung und schließt den Wahrnehmungs-Aktions-Loop für den gesamten Greifprozess. Meine Arbeit stellt die Infrastruktur für zukünftige Forschungsrichtungen bereit und ebnet den Weg für die Verwirklichung von Geschicklichkeit auf der Hand des Menschen.

Contents

1	Introduction	1
1.1	A brief history of research on robotic grasping	1
1.2	Challenges	2
1.3	Main contributions and outline of this thesis	4
2	A Probabilistic Graphical Model for the Grasping Problem	9
2.1	Introduction	9
2.2	Related work	9
2.3	Probabilistic modeling of grasping using Bayesian networks	10
2.3.1	Variable definition and graph structure	10
2.3.2	Query of conditional probability distribution	12
2.4	Inference in different grasping scenarios	13
2.4.1	Grasping known objects under task constraints	13
2.4.2	Grasping unknown objects	14
2.4.3	Grasping objects under limited sensing capability	16
2.5	Summary	19
3	A Skill Framework for Assembly Tasks Using Mobile Manipulator	21
3.1	Introduction	21
3.2	Related Work	22
3.2.1	Methods of grasp synthesis considering task information	23
3.2.2	Methods of solving assembly problem using skill concept	25
3.3	Contributions	27
3.4	Framework description	28
3.5	High-level skills	29
3.5.1	Pick & Reconfigure	31
3.5.2	Pick & Insert	34
3.6	Low-level skills	37
3.7	Grasp planing under task constraints	43
3.7.1	Grasp and task parameterization	43
3.7.2	Probabilistic model of satisfying task constraints	44
3.7.3	Model verification	45
3.7.4	Grasp realization	49
3.8	Sensing and actuation coordination	49

3.9	Experiments	51
3.9.1	Evaluation of the skill framework	51
3.9.2	Reachability analysis	59
3.9.3	Benchmarking manipulation scenarios	60
3.10	Summary	66
4	A Probabilistic Approach to Grasp Synthesis	69
4.1	Introduction	69
4.2	Related work	70
4.2.1	Methods considering pose uncertainty	71
4.2.2	Methods considering shape uncertainty	71
4.2.3	Methods using learning based approach	72
4.3	Contributions	76
4.4	Probabilistic modeling of grasping success	77
4.4.1	Mathematic model	77
4.4.2	Illustration of the concept	77
4.4.3	Modelling of conditional grasp success probability	78
4.5	Modelling perception uncertainty	82
4.5.1	Surface based probability distribution estimation based on probabilistic fusion	82
4.5.2	Surface representation: p-SDF	82
4.5.3	Surface reconstruction with sensor fusion	82
4.6	Searching for feasible grasp configuration	84
4.6.1	Reducing configuration space of grippers using synergy	85
4.6.2	Calculation of surface normals	86
4.6.3	A sampling-based approach to grasp synthesis	88
4.7	Sensitivity analysis of p-SDF	88
4.8	Grasping experiments	91
4.8.1	Experimental setup	92
4.8.2	Performance evaluation on grasp synthesis algorithm	94
4.8.3	Performance on grasping accuracy	96
4.8.4	Comparison with one baseline approach	99
4.8.5	Discussion	100
4.9	Summary	101
5	An Adaptive Grasping Control Architecture for Whole Body Mobile Manipulation	103
5.1	Introduction	103
5.2	Related work	104
5.2.1	Methods using visual servoing	104
5.2.2	Methods using force sensing	106

5.2.3	Methods considering whole body coordination	108
5.3	Contributions	110
5.4	System architecture	110
5.4.1	Motion adaptation	111
5.4.2	Arm platform redundancy resolution	113
5.4.3	Whole-body coordinated control	115
5.5	Use case: grasping cylindrical objects with unknown dimension	116
5.5.1	Measurement model for object state estimation	117
5.5.2	Grasp synthesis for circular objects	118
5.5.3	Experimental evaluation	120
5.6	Summary	123
6	Conclusion and Future Work	127
6.1	Concluding remarks	127
6.2	Future directions	128
	Bibliography	137

1 Introduction

Grasping is one of the fundamental skills of human beings. A three months old infant already possesses a palmar grasp reflex. A baby closes his fingers when an object is placed in the palm. In the sixth month, the ability of active grasping is further developed. They can track an object with their eyes and grasp them actively. Gradually, they gain how to hand over an object between the left and the right hand. From eighth to ninth month, they can already perform precision grasping by only two fingers. A ten months old infant already has some ability to perform task oriented grasping. They know how to grasp a nipple or a nursing bottle correctly so that they can suck them after grasping. As we can see, a human develops its grasping ability in the very early age, from the passive grasping reflex to the active grasping skill with task intention and accuracy.

By contrast, the most advanced robots can not reach the dexterity of grasping compared to a human. In the factory, robots are designed to work in a structured environment. They have perfect knowledge of what type of objects they are working with, how the objects should look like, when and where they should appear. Therefore, they can perform a range of manipulation tasks with high speed. These include material handling, assembly, etc. Without this prior knowledge while operating robots in unstructured environment, grasping is a very challenging task for robots. People may ask why grasping is fairly a simple task even for little children but difficult to robots. In unstructured environment with little prior knowledge, robotic grasping is an extremely complex problem, which requires mastering of several tasks. These include object perception, reasoning about grasp configuration and robust control of grasp execution.

1.1 A brief history of research on robotic grasping

The research on robotic grasping has a long history. Most researchers in the earliest time address the problem through analytic approaches. Previous work regarding contact models ([120],[121]), form/force closure ([26],[100]), force optimizations([12],[84])etc. lay the theoretical framework of determining a firm grasp. The stability of a grasp is usually quantified by a quality metric ([115],[9]). The quality metric describes how much external wrench a grasp can resist. Bicchi and Kumar [4] provides a comprehensive survey of these methods.

Using a simulator can, in general, accelerate algorithms designed to work on real robots. In the grasping community, there was no simulator until the emergence of ‘*GraspIt*’ [90]. The ‘*GraspIt*’ simulator provides a fast collision detection and contact determination in-

terface. It is designed for grasp planning with arbitrary objects and gripper types. Since then, a lot of grasp synthesis approaches ([91],[71],[92]) have been developed based on this tool. However, it remains a challenge to apply these approaches for grasping real objects, because most of them rely on prior knowledge of a given object model. Obviously, it is not feasible to create models for every particular item in the world. In addition, due to the imperfection of real sensors, the perception of a robot usually contains estimation error, which further influence the grasping success. These approaches can not handle the problem of this kind.

Recently, with the development of new sensing technologies, low-cost sensors for acquiring rich 3D information of the environment are available. Data-driven methods became familiar and put more emphasis on perception for grasping. Different from analytic approaches, they focus on extracting information from the sensor input, which is relevant for the synthesis of a grasp. These include object representations, extraction of grasp relevant features and reactive grasp executions. These methods can better handle the perception uncertainty which is essential for the real grasping problem. A comprehensive review of data-driven methods is given by Bohg et al. [7].

1.2 Challenges

Although robotic grasping has been studied intensively in the past, it remains a gap between the theoretical findings and technical realization in a real environment. Various models and methods have been proposed to find feasible grasp configurations. However, there are only a few methods which both consider how to generate task orientated grasp configuration and exploit the robot kinematics simultaneously. In addition, most object representations designed for grasp synthesis simply use a mesh model for grasp synthesis, which is not able to integrate perceptual uncertainty. Methods that able to handle the perceptual uncertainty probabilistically and model the object efficiently are missing. Furthermore, an integrated system solution for taking advantage of a robot's structure while considering the perceptual uncertainty, and allowing a robust adaptive grasp execution is not available. The various aspects of challenges for robotic grasping are summarized in this section.

How to design a task-oriented grasping skill which is reusable for mobile manipulation tasks?

In a mobile manipulation scenario such as an assembly task, a robot is required to assemble several objects together. The robot usually has to perform a sequence of manipulation actions, which include grasping, placing, insertion and so on. According to a given assembly plan, one object may be picked and placed multiple times, so that the orientation of the object meets the conditions of the following manipulation steps. Therefore, grasping serves as one of the core skills for the task. It is preferred to have a reusable grasping skill which

can be easily used throughout the entire manipulation process. In addition, grasping has also to be task-oriented. For an object and a given gripper, it may exist thousands of grasp configurations which are feasible, but only a sub-set of these configurations meet both task constraints and the workspace constraints. How to calculate a ‘good’ robot configuration for grasping is an issue. Another problem regarding this scenario is that the objects to be assembled may not directly locate in the workspace of the manipulator. The robot must use its mobility to move to positions so that the objects are reachable. Determining these intermediate base positions which minimize the movement of the robot while combining the base motion and manipulator motion can increase the overall task performance. It remains a big challenge how to address the above-mentioned issues in an integrated framework.

How to model a grasping favorable object representation and use it for precision grasp synthesis?

A large portion of objects daily life have irregular shapes. To handle the variety of object forms is a challenge. One naïve approach for representing such objects is by approximating the geometry by shape primitives. This representation may be feasible for power grasping such as caging, but is not suitable for precision grasping with fingertips. For tasks which require precision, choosing of an appropriate representation for grasp synthesis is an issue. A good representation should be able to model the surface as accurate as possible. As an accurate representation lies the foundation for grasp synthesis algorithm to generate precision grasps. If the object model is not known a priori, the representation should allow fast reconstruction online. In addition, as the object model can never be perfect, the representation should model the uncertainty of the surface, so that the robot is able to estimate the probability of success. Furthermore, for grasp synthesis, it is often required to evaluate a large number of grasp hypotheses, a model which allows fast query of object surface normals can accelerate the speed of finding a feasible grasp configuration. It is a big challenge to design such a representation which can include all the mentioned features and a grasp synthesis method that is tailored for the representation.

How to increase the robustness of grasp execution in spite of perceptual uncertainty?

Grasp execution refers to the actual motion of performing a grasp. Traditional approaches usually execute pre-computed grasp movements in open loop. However, due to perceptual uncertainty, the planned grasping points may differ from the actual contact points, performing of the scheduled grasp trajectory may result in failure. One approach to increasing the robustness of grasping is by closing the perception-action loop during the grasp execution. The robot is required to adapt its grasp motion to the perceptual result. For a mobile manipulator, how to coordinate and control the required component to achieve such adaptive grasp behavior is a big challenge.

1.3 Main contributions and outline of this thesis

In this dissertation, a general and integrated system is developed to tackle the challenges as mentioned above for robotic grasping. Three grasping scenarios which represent different aspects of grasping are explored. Fig. 1.1 illustrates the outline of this thesis. In chapter 2, we propose to unify various aspects of challenges using a Bayesian network. The variables defined in the model represent factors which are relevant to the overall grasping success. The difference between the scenarios only exist in which variables are observable. In chapter 3, we focus on how to generate reusable task-oriented grasping skills for assembly task. For this purpose, a skill-based framework is proposed so that a mobile manipulator can perform the assembly task very efficient and flexible. Different from the objects that are known in prior in an assembly tasks, we address the problem of unknown object grasping in chapter 4. A grasp synthesis method based on a novel probabilistic object representation is proposed in this chapter. The method improves the accuracy of grasping significantly. In chapter 5, we tackle another grasping scenario that the accuracy of sensor used for grasping is not high. To face the challenge, an adaptive grasping control architecture is proposed so that the robot is able to adapt its grasping motion online during execution. In this way, robust grasp execution is achieved despite sensing uncertainty.

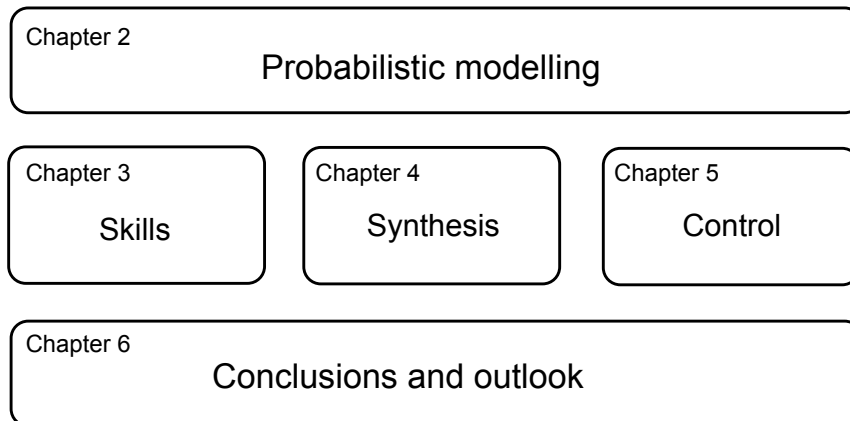


Fig. 1.1: Thesis outline















A skill framework for the assembly problem exploiting arm-base coordination

Mobile manipulators are the robots which equipped with a mobile base platform for navigation and arm(s) for manipulation. For reach-and-grasp tasks, state-of-the-art approaches consider it as two isolated problems: a navigation problem followed by a motion planning problem. Using these approaches a robot should first move its mobile platform to a location so that the object is reachable, then use its manipulator to accomplish the object grasping task. Although this is a feasible solution, the time used to perform this task is not efficient. In contrast, a human performs the same task with all his actuation capability (arm, hand,

body, locomotion) in a coordinated manner. In chapter 3, inspired by the human, we propose to bridge the gap between human reach-and-grasping and robotic reach-and-grasping by arm-platform coordination. An analysis of the reachability of a mobile manipulator is first conducted. The result revealed that the overlap between the workspace of a manipulator with the observation space of a fixed mounted camera is very limited. As a result, if the design of a mobile manipulator does not consider this aspect, it may have to face the following dilemma, the target object is located in the field of view of the robot, but it is not reachable from the robot manipulator, or the object is reachable from the manipulator but lies out of the field of view of the robot. This situation implies that arm-platform coordination is the only promising approach to solve the reach-and-grasping problem efficiently. To enable a mobile manipulator to move with its full degree of freedom (DOF), we propose a concept of virtual joints as well as a virtual kinematic chain. The virtual kinematic chain is composed of both the manipulators' joints and virtual joints. In this way, the planning of the arm-platform coordinated motion can be seamlessly integrated into an existing motion planning framework. We demonstrate the benefits of this concept in an assembly problem and propose a skill-based framework to solve the problem in a generalizable fashion. In this framework, reusable high-level skills can be parametrized to perform pick, place and insert operations. The high-level skills can reason about where to grasp the object so that the constraints for place and insert operation are satisfied. Fig. 1.2 illustrates the input and output of the system.

A probabilistic approach to the grasp synthesis for unknown objects

Because of the uncertainty of the real world, dynamics of objects and error in perception, it is difficult for the robot to ensure that the grasp action will always succeed. To manage the complexity and uncertainty in the real world, modeling the success probability of a grasping action is a promising method to handle these problems. In chapter 4, the success probability is calculated based on a model of conditional grasping success and a model of object posterior. This method reduces the complexity of directly modeling the grasp success probability. It splits a complex model into two independent models which are not correlated with each other. The conditional grasping success model describes the probability of success by taken an action, under the condition that the object state is given. It is modelled using four criteria, which maps the principle of underlying grasp physics, actuation error and representation uncertainty to the model. Object posterior models the object state distribution after a series of observations are obtained. Depending on the choice of the representation, for some real world objects whose shape can be approximated by a group of shape primitives, the dimension of the object state space is small. The object posterior of these objects can be computed by e.g. Bayes filtering. For most real world objects, the dimension of state space is large because of their individual shape. We propose a new surface representation and a fusion method to compute the object posterior. The underlying structure of the surface representation is a variance augmented signed distance

Step	Skill	Parameters	Subproduct
1.	P.R	 Back	
2.	P.I	 	
3.	P.R	 Bottom	
4.	P.I	 	
5.	P.I	 	
6.	P.I	 	

(a)



(b)

Fig. 1.2: (a) A high-level assembly sequence provided to the system. P.R: The skill for picking and placing an object in another orientation. P.I: The skill for picking an object and inserting in another one. (b) A mobile manipulation robot performs the assembly task autonomously.

function. This representation allows temporal and spatial fusion by multiple depth cameras with individual noise characteristics. The result of the sensor fusion is a uncertainty-aware surface distribution which approximates the object posterior. After modeling the grasp success probability, the aim is to search for a grasp action that maximizes the success probability. For this purpose, we propose a simulated annealing method to find a feasible grasp. This approach seamlessly connects grasp perception and grasp control in a systematic way, while serves as a foundation for grasping unknown objects. The computational time of our method performs 30 times faster than a state-of-the-art method which uses a similar representation. Experimental results verify a significant grasping accuracy improvement in the scenario of grasping real world unknown objects by using the proposed method. Fig. 1.3 elaborates some detail of proposed method.

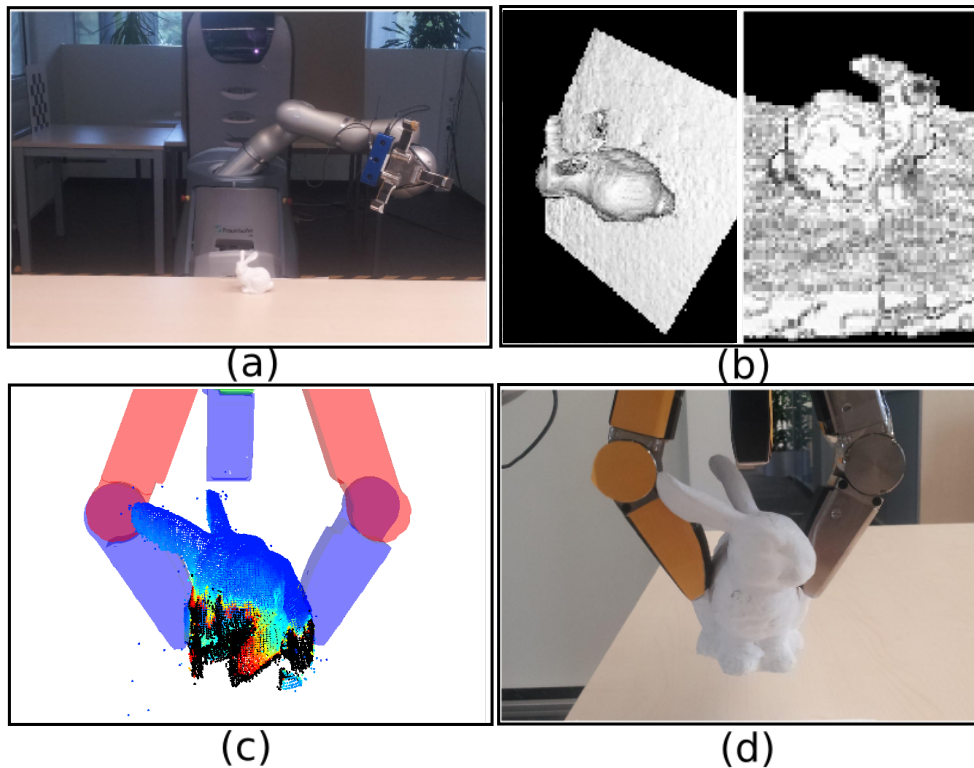


Fig. 1.3: (a) The object is presented to the robot for the first time. (b) The robot reconstructs the object surface by fusing sensor measurement from two depth cameras. (c) An uncertainty-aware object model is constructed and a feasible grasp is found for grasping the object. (d) Successful execution of the grasp on a real robot.

An adaptive control architecture for closed-loop grasp execution

Chapter 5 addresses the control aspect of the grasping process. In general, individual actuators of a mobile manipulator are usually controlled separately. However, arm-platform coordination requires control commands to be sent to each actuator simultaneously. To control the individual actuators in synchrony, we propose a system control architecture,

which enables a plug-in mechanism to combine arbitrary independent actuators into a standard control loop in a flexible manner. This allows a mobile manipulator to easily switch and combine the actuators according to the requirements of a grasping phase. For example, a mobile manipulator can use arm-platform coordination in the pre-grasp phase for moving to a reachable position and for interacting with an object. Motions executed by the arm alone can be employed in the grasp-motion phase for approaching a grasp configuration. Arm-gripper coordination can be utilized in the force-closure phase for stabilizing the object within the gripper. In most previous approaches, motions of the grasping are usually planned beforehand. Thus, the adaptation of the grasp motion to the change of the object state is not possible. We propose to solve this problem by an online trajectory generation algorithm and feed the state of the object back to the trajectory generator. In this way, a large control loop is closed by the system. Using the proposed method, a mobile manipulator is not only able to reason about the grasp configuration but also recognize the change of the environment and adapt to the change. As a result, the robustness of the grasping accuracy is significantly improved. In Fig. 1.4, the robot uses the proposed method to grasp light objects.

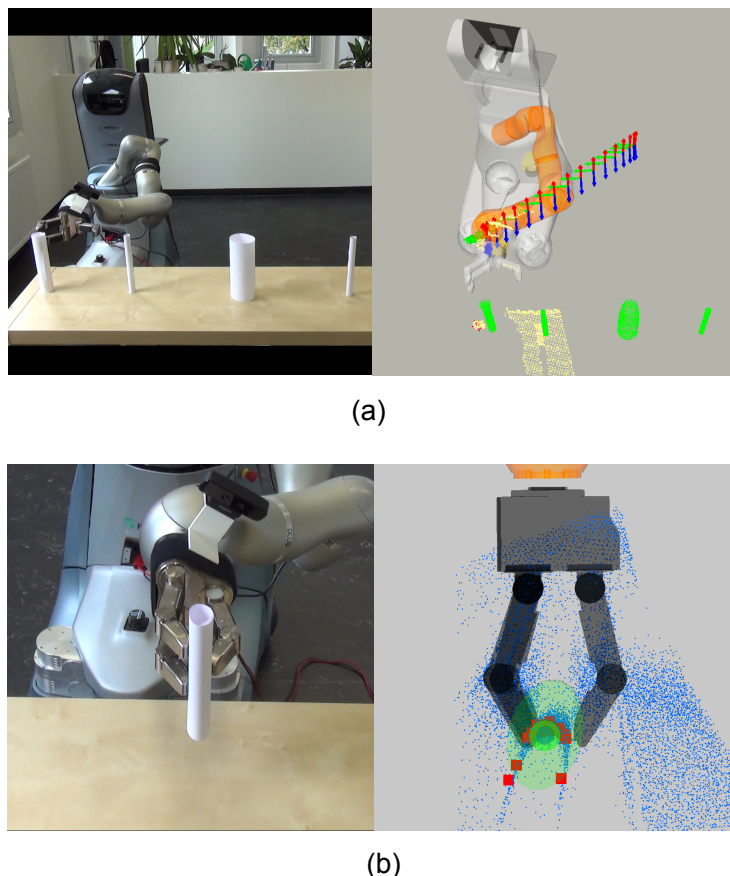


Fig. 1.4: (a) The robot is about to grasp light objects. The radius and position of the objects are estimated online during the reach-to-grasp phase. (b) The robot successfully pick up the object despite of large sensing uncertainty.

2 A Probabilistic Graphical Model for the Grasping Problem

2.1 Introduction

As mentioned in the previous chapter, the essence of the robotic grasping problem is to control the actuators to achieve grasping success. Due to the process uncertainty, the same measurement and the same actuator control may not result in the same outcome. A lot of uncertain factors are involved in the grasping process. These include imperfect sensor input, the shape of an object, precision of control and satisfaction of task constraints. These factors determine the final outcome of the grasp process. Therefore, it is more reasonable to use a probabilistic model to estimate the probability of grasp success rather than using a deterministic model. As the success probability of grasping depends on many factors, directly modeling the problem from the factors to success probability is complex and hard. One way to handle the complexity is to model the grasping problem using probabilistic graphic models [67].

Probabilistic graphic models use a graph structure to represent a probabilistic distribution over high dimensional space. The major advantage of using a graph-based representation is one can break the modeling of complex probability distribution over every possible variable into simple small factors. The joint distribution of every possible variable is simply the multiplication of these small factors. Specifically, we can break up the modeling of the joint distribution over all the factors which may influence the success into smaller manageable probability models. We use one family of the probabilistic graphic models called ‘Bayesian Networks’ to model the grasping problem. The ‘Bayesian Networks’ uses a directed graph to represent the probability distribution. A directed graph is composed of a set of nodes and directed edges. The nodes specify random variables that represent influence factors. The edges define the direction of dependency between factors. The reader can refer to [69] for a more formal description of ‘Bayesian Networks.’

2.2 Related work

In previous research work, grasping under uncertainty can be handled using different assumptions and solutions. One assumption is that the robot may use its fingertip sensor to sense and interact with the target object. Based on this assumption, Hsiao et al [52] models the grasping problem as a partially observable Markov decision process (POMDP), and

demonstrates in a 2-D planer simulation how the grasping policy is generated by solving the POMDP. Another direction of handling uncertainty focuses on grasp planning. Kim et al. [66] proposed a grasp quality under pose uncertainty by capturing the dynamics of the object in a physical simulation. Roa et al [116] proposed to plan independent contact regions for finger placement by considering the kinematic of the gripper. Uncertainty can also be handled by generating reach-to-grasp motion. Stulp et al [125] proposed to use dynamic motion primitives (DMP) to plan reach-to-grasp motion rather than a single grasp for handling uncertainty.

As the uncertainty can be handled in different ways, the major difference of the proposed formulation is that it is not a specific solution to solve a specific scenario of grasping under uncertainty but provides an overall description of which part of the uncertainty in the system must be considered. This formulation can be used to model grasping under different pre-conditions. These include grasping selection under task specification, grasping synthesis of known/unknown objects, and reducing of the uncertainty during grasp execution.

2.3 Probabilistic modeling of grasping using Bayesian networks

In this section, we introduce a general description for grasping problems using a Bayesian network. We first define a set of variables for the problem. A graph is then proposed to model the dependency between the variables. Then we explain that solving grasping problems is equivalent to solving inference tasks under the condition that a subset of the defined variables is observed.

2.3.1 Variable definition and graph structure

The first variable that we define for the grasping problem is the outcome of a grasping process S . The outcome is a discrete random variable which takes two values to specify whether a grasping process succeeds or not. One standard definition of the success is whether the gripper of a robot achieves force closure on the object. However, this definition only considers the result of grasp acquisition itself. As grasping of an object may also have a purpose, the task intention of a grasp should also be considered as a part of grasp success. Therefore, we define the success of a grasping process not only based on force closure property but also on whether the task constraint is fulfilled. Thus, we define another two discrete variables F for whether the gripper achieves force closure condition and C for whether constraints defined by the grasping task is fulfilled. Consequently, the outcome of a grasping process S only depends on F and C . Now let's consider which factors can influence the force closure condition F . It is reasonable to think that object state X and robot controls U jointly affect the force closure condition. As the object state X is

normally not observable and derived from the sensor measurement, we define Z to be the sensor measurement that can influence the estimation of object state. The task constraint satisfaction C represents whether a grasp configuration satisfies the condition regarding a specified task. Therefore, a task specification T naturally influences the task constraint. Besides that without the grasp configuration, it is not possible to evaluate whether task constraint is fulfilled or not. Therefore, C is also dependent on object state X and robot control U .

Based on the analysis of the factors and their dependencies we can model the relationships of the factors into a Bayesian network. Figure. 2.1 depicts the proposed network that we define for the grasping problem. All the variables surrounded by a box are binary-valued discrete variables, while the variables surrounded by an ellipse indicate their domain can be either discrete or continuous. All the defined variables are summed up in Tab. 2.1.

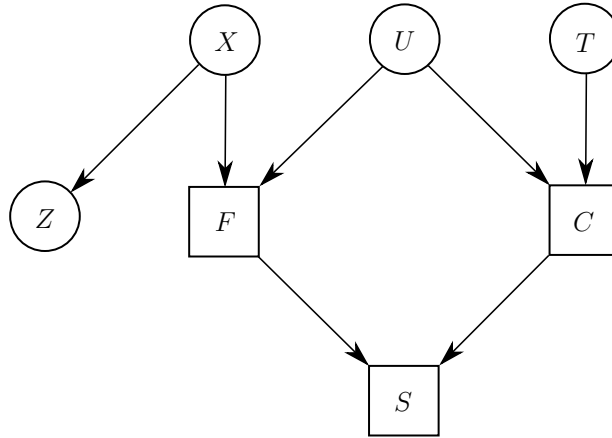


Fig. 2.1: A Bayesian network for the grasping problem. Variables surrounded by a box are binary-valued discrete random variables. Variables surrounded by a circle can be either continuous or discrete.

Variable	Description
S	outcome of a grasping process
F	satisfaction of force closure condition
U	robot control
Z	sensor measurement
X	object state
T	task specification
C	satisfaction of task constraints

Tab. 2.1: Definition of variables

Based on the proposed Bayesian network, the joint distribution of all the variables can be written according to the graph structure. The joint distribution of the grasping problem

is given by

$$P(S, F, C, X, Z, U, T) = P(S|F, C) \cdot P(F|X, U) \cdot P(C|U, T) \cdot P(Z|X) \cdot P(X) \cdot P(U) \cdot P(T) \quad (2.1)$$

where $P(S|F, C)$, $P(F|X, U)$, $P(C|U, T)$ and $P(Z|X)$ are conditional probability distributions (CPD). Among them, the CPD of $P(S|F, C)$ can be expressed using a CPD table, because all the variables involved are discrete. The outcome S of a grasping process is success, if and only if F is success and C is success. Therefore, the CPD of $P(S|F, C)$ is directly given by Tab. 2.2. The CPD of $P(F|X, U)$ specifies the probability of force closure. The modeling of this CPD depends on which representation of X and of U are used. $P(C|U, T)$ models the probability of task constraint satisfaction. $P(Z|X)$ models the probability distribution of Z given the object X , which is per definition the measurement model of a sensor. $P(X)$, $P(U)$ and $P(T)$ are the prior distributions respectively.

	s^0	s^1
f^0, c^0	1	0
f^0, c^1	1	0
f^1, c^0	1	0
f^1, c^1	0	1

Tab. 2.2: The conditional probability distribution for $P(S|F, C)$. The superscript 1 and 0 indicate the variable equals success or failure respectively.

2.3.2 Query of conditional probability distribution

One of the advantages of using probabilistic graphical models is we can perform inference tasks. The goal of an inference task is to query the probability of a set of variables based on a set of observed evidence. Mathematically, let Y be a set of variables for the query, $E = e$ be a set of evidence. The inference task is to compute the conditional probability distribution of Y given E . The conditional probability distribution is given by

$$P(Y|E = e) = \frac{P(Y, E = e)}{P(E = e)} \quad (2.2)$$

In the context of our grasping problem, the evidence variables are measurement Z , robot controls U and task specifications T . The probability distribution that most interests us is $P(S|Z, U, T)$, the outcome of a grasping process given the evidence variables. The query of the probability gives how likely a grasp will succeed. Then, the goal of grasping is to maximize the probability of success $P(S = 1|Z, U, T)$. Among the evidence in this probability, Z is measurement given by the sensor and T is given by the task, the only variable that can be changed to maximize the success probability is the robot control U . Therefore, the grasping problem is defined by find the best robot control $U = u^*$, so that

the conditional probability $P(S = 1|Z = z', U = u^*, T = t')$ is maximized:

$$u^* = \arg \max_u P(S = 1|Z, U, T) \quad (2.3)$$

2.4 Inference in different grasping scenarios

So far we propose a coherent probabilistic model to describe the grasping problem. In reality, the objects to be grasped, the type of sensor used for perception and the type of gripper jointly determine the hardness of a grasping problem. Hence, the representation of the random variables and the modeling of the conditional probability distribution can vary with the condition of a grasping problem. In the following sections, we elaborate how we configure the model in different grasping scenarios.

2.4.1 Grasping known objects under task constraints

In chapter 3, we consider the assembly as our target application scenario. In this scenario, since object models are given prior to grasping, the robot can use the object models for grasp planning and perception. Specific task constraints are defined for the objects, so grasping of an object is always related to a purpose. The robot has to infer the control to ensure that a specific manipulation task can be performed after grasping. Since grasping of known objects has been well studied in the previous research, we manually specify a set of force closure grasps to the object. In this way, we are allowed to condition the random variable $F = 1$ to assume that the force closure is already fulfilled. The conditional probability of success in this scenario can be reformulated by

$$P(S = 1|Z, U, T) = P(S = 1|Z = z, U = u, T = t, F = 1). \quad (2.4)$$

Similar to the previous derivation, the conditional probability of success can be expanded using Bayes rule and is expressed by

$$P(S = 1|Z, U, T, F = 1) = \frac{P(S = 1, z, u, t, f)}{P(z, u, t, f)}. \quad (2.5)$$

We can further compute the nominator of the Equ. 2.5 by marginalizing C and X given by

$$\begin{aligned} P(S = 1, z, u, t, f) &= \sum_{C, X} P(S = 1, C, X, z, u, t, f) \\ &= \sum_X P(S = 1, C = 0, X, z, u, t, f) + \sum_X P(S = 1, C = 1, X, z, u, t, f). \end{aligned} \quad (2.6)$$

Similar as in the Equ. 2.14, the first term of Equ. 2.6 also equals zero, because the conditional probability of success given the evidence $P(S|C = 0)$ is zero according to Tab. 2.2. The second term can be further expressed by

$$\begin{aligned}
 P(S = 1, z, u, t, f) &= \sum_X P(S = 1, C = 1, X, z, u, t, f) \\
 &= \sum_X P(S = 1|F = 1, C = 1) \cdot P(F = 1|X, u) \cdot P(z|X) \\
 &\quad \cdot P(C = 1|u, t) \cdot P(X) \cdot P(u) \cdot P(t) \\
 &= \sum_X P(C = 1|u, t) \cdot P(z|X) \cdot P(X) \cdot P(u) \cdot P(t).
 \end{aligned} \tag{2.7}$$

Now we can replace the term $P(z|X) \cdot P(X)$ by $P(X|z) \cdot P(z)$ using Bayes rule and moves the sum product before $P(X|z)$. The joint probability is then given by

$$P(S = 1, z, u, t, f) = P(C = 1|u, t) \cdot \sum_X P(X|z) \cdot P(z) \cdot P(u) \cdot P(t). \tag{2.8}$$

With $\sum_X P(X|z) = 1$ and $P(z) \cdot P(u) \cdot P(t)$ is equal to a normalization constant η''' . We can write the joint probability further by

$$P(S = 1, z, u, t, f) = \eta''' \cdot P(C = 1|u, t). \tag{2.9}$$

Combining the denominator which is also a normalization constant, the final conditional probability which we want to query is simplified to

$$\begin{aligned}
 P(S = 1|Z, U, T, F = 1) &= \frac{\eta''' \cdot P(C = 1|u, t)}{P(z, u, t, f)} \\
 &= \eta'''' \cdot P(C = 1|u, t).
 \end{aligned} \tag{2.10}$$

Therefore, the problem of grasping known objects with task constraints is defined by finding the best robot control $U = u^*$, so that the conditional probability of satisfying the task constraints is maximized.

$$u^* = \arg \max_u P(C = 1|u, t) \tag{2.11}$$

2.4.2 Grasping unknown objects

In chapter 4, we consider the scenario of grasping unknown objects. In this case, no information about the objects is available in prior. The robot neither knows the geometry nor the pose of an object. In this scenario only ‘Pick and place’ is one of the reasonable tasks. Since there is no information about object’s geometry, the orientation of an object can be chosen arbitrarily for a placing task. It is reasonable to assume that the task constraint is already fulfilled. Therefore, we can condition an additional variable in the

set of evidence by setting $C = 1$. Based on the graphical model, conditioning of variable C blocks the reasoning pattern from task specification T to the outcome S , T and S are therefore conditional independent. The probability of success specified in Equ. 2.3 can be reformulated by

$$P(S = 1|Z, U, T) = P(S = 1|Z = z, U = u, T = t, C = 1) \quad (2.12)$$

where the variable C now becomes an evidence. Based on the Bayes rule, the conditional probability can be further expanded by

$$P(S = 1|Z = z, U = u, T = t, C = 1) = \frac{P(S = 1, z, u, c, t)}{P(z, u, c, t)}. \quad (2.13)$$

Here we replace $C = 1$ by c to keep the equation concise. The nominator of Equ. 2.13 can be computed by marginalizing the joint distribution of all the variables specified in Equ. 2.1. The variables to be marginalized are neither in the set of query nor the set of evidence. In this case, the variables to be marginalized are X and F . Without loss of generality, we use the sum operation for marginalization by assuming all the variables are in discrete space. Later we can freely replace the sum operation by an integral if the actual representation of the variable is in continuous space. First, we eliminate the variable F . Since F can only take 0 or 1, the result of elimination is thus given by

$$\begin{aligned} P(S = 1, z, u, c) &= \sum_{F, X} P(S = 1, F, c, X, z, u, t) \\ &= \sum_X P(S = 1, F = 0, c, X, z, u, t) + \sum_X P(S = 1, F = 1, c, X, z, u, t). \end{aligned} \quad (2.14)$$

The first term $P(S = 1, F = 0, c, X, z, u, t)$ can be expanded by $P(S = 1|F = 0, C = 1)$ multiplying the rest of the factors according to Equ. 2.1. The condition probability $P(S = 1|F = 0, C = 1)$ is equal to zero according to Tab. 2.2, so the first term of Equ. 2.14 equals zero regardless what value the rest of the factors take. So we can simplify the joint probability $P(S = 1, z, u, c)$ by considering only the second term by

$$\begin{aligned} P(S = 1, z, u, c) &= \sum_X P(S = 1, F = 1, c, X, z, u, t) \\ &= \sum_X P(S = 1|F = 1, C = 1) \cdot P(F = 1|X, u) \cdot P(z|X) \\ &\quad \cdot P(C = 1|u, t) \cdot P(X) \cdot P(u) \cdot P(t) \end{aligned} \quad (2.15)$$

According to Tab. 2.2, we know that the CPD of $P(S = 1|F = 1, C = 1)$ equals 1. The probability of task satisfaction $P(C = 1|u, t)$ also equals 1, because we assume the task constraint is always fulfilled in this scenario. The multiplication of $P(u) \cdot P(t)$ can be

summarized by a constant η . The object prior $P(X)$ and the sensor model $P(z|X)$ can be summarized by $\eta' \cdot P(X|z)$. Thus we can simplify the Equ. 2.15 by

$$P(S = 1, z, u, c) = \eta \cdot \eta' \cdot \sum_X P(F = 1|X, u) \cdot P(X|z) \quad (2.16)$$

Now let's look at the denominator of the Equ. 2.13. Since the value of the denominator is also a constant η'' , the conditional probability what we query for can be formulated by

$$\begin{aligned} P(S = 1|Z = z, U = u, T = t, C = 1) &= \frac{\eta \cdot \eta'}{\eta''} \cdot \sum_X P(F = 1|X, u) \cdot P(X|z) \\ &\propto \sum_X P(F = 1|X, u) \cdot P(X|z). \end{aligned} \quad (2.17)$$

Therefore, the problem of grasping unknown objects is defined by finding the best robot control $U = u^*$, so that $P(S = 1|Z = z, U = u, T = t, C = 1)$ is maximized:

$$u^* = \arg \max_u \sum_X P(F = 1|X, u) \cdot P(X|z) \quad (2.18)$$

The challenge in this grasping scenario can be seen from Equ. 2.18. The probability of force closure and the posterior probability of object state along with finding the optimal robot control are the major problems to be addressed in this scenario.

2.4.3 Grasping objects under limited sensing capability

In chapter 5, we consider the grasping scenario in which the sensor measurement used for perception has a significant noise characteristic. Specifically, a low-cost time-of-flight depth camera is used as the primary sensor for grasping. The challenge in this scenario is how to improve the grasp success despite significant sensing uncertainty. We propose to use a camera-in-hand setup to measure the object continuously during the grasp motion phase. The closer the distance from the camera to an object, more object detail can be measured. In this way, the uncertainty of the object can be reduced online which eventually increase the success of grasping.

To describe the problem using a graphical model, we need to extend the above model in a temporal fashion. In Fig. 2.2, we first encapsulate the original graphical model into a block, where three variables are exposed to the outside. Then we chain the small blocks in series to obtain a new model as depicted in Fig. 2.3. We use subscript $0, 1, \dots, i$ to indicate the time dependency of the variables. The grasp process begins with the first measurement in time stamp zero. As a result, a dependency is created between X_0 and Z_0 . Based on the sensing result, a control trajectory $U_0 = u_0$ is inferred to maximize the probability of success S_0 . The robot performs one step of the trajectory u_0 and the grasping process moves on to the next time stamp. Since the camera also moves towards the object, the

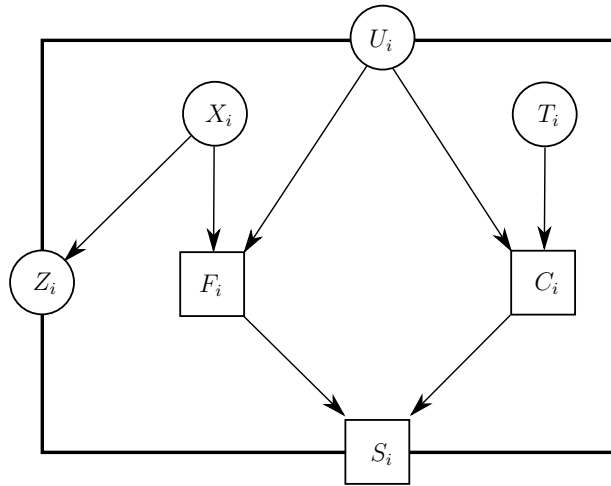


Fig. 2.2: Encapsulation of the original graphical model.

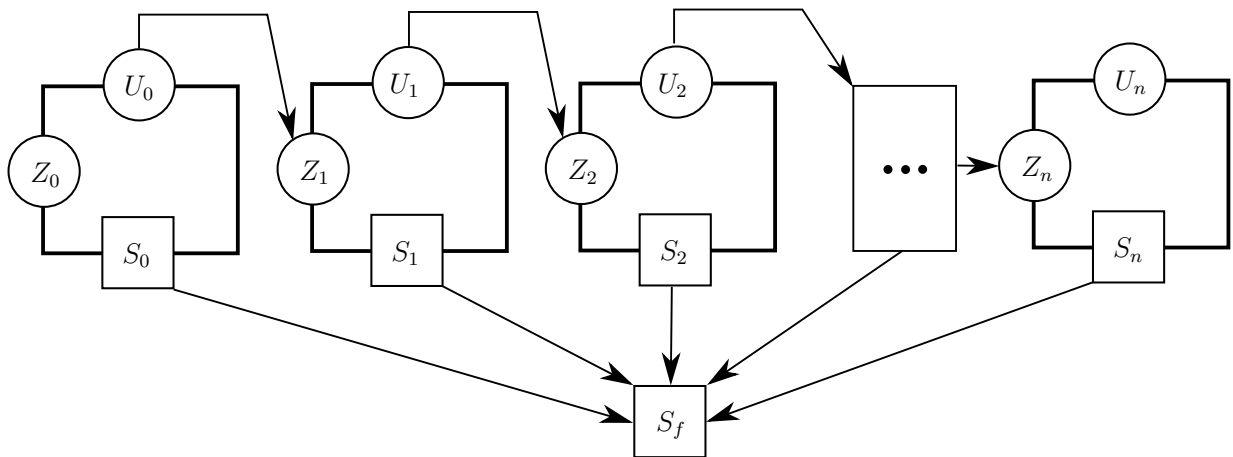


Fig. 2.3: Chaining of the original graphical models.

measurement at the next time stamp Z_1 depends on the last trajectory U_0 and the object state X_1 . The sensing and control loop repeats until the last measurement Z_i and control trajectory U_i are taken. The conditional probability of success is formulated by

$$P(S_f = 1 | \underline{Z} = \underline{z}, \underline{U} = \underline{u}, \underline{T} = \underline{t}) = \frac{\sum_{\substack{F_0, \dots, F_n, \\ C_0, \dots, C_n \\ X_0, \dots, X_n \\ S_0, \dots, S_n}} P(S = 1, \underline{F}, \underline{C}, \underline{X}, \underline{z}, \underline{u}, \underline{t})}{P(\underline{Z} = \underline{z}, \underline{U} = \underline{u}, \underline{T} = \underline{t})} \quad (2.19)$$

where $\underline{Z} = \{Z_0, \dots, Z_n\}$, $\underline{U} = \{U_0, \dots, U_n\}$ and $\underline{C} = \{C_0, \dots, C_n\}$. The condition $\underline{t} = \{t, \dots, t\}$ indicates in each time stamp the task specifications are the same. Again, the nominator is computed by marginalizing the joint probability of all the variables which are neither query nor conditions. The denominator is a renormalization constant. The joint probability distribution in the nominator can be computed based on the graph structure by

$$\begin{aligned} P(S_f = 1, \underline{F}, \underline{C}, \underline{X}, \underline{z}, \underline{u}, \underline{t}) &= \prod_{i=1:n} P(F_i | X_i, u_i) \cdot P(C_i | u_i, t) \cdot P(z_i | u_{i-1}, X_i) \\ &\quad \cdot P(F_0 | X_0, u_0) \cdot P(C_0 | u_0, t) P(z_0 | X_0) \\ &\quad \cdot P(S_i | F_i, C_i) \cdot P(S_f = 1 | S_1, \dots, S_n) \\ &\quad \cdot P(X_i) \cdot P(u_i) \cdot P(X_0) \cdot P(u_0) \end{aligned} \quad (2.20)$$

First, we eliminate discrete variables \underline{F} , \underline{C} and \underline{S} . The term $P(S_f = 1 | S_1, \dots, S_n)$ equals zero as long as one of the S_i does not equal to one. Therefore, after marginalization of \underline{S} the nominator equals

$$\sum_{\substack{F_0, \dots, F_n, \\ C_0, \dots, C_n \\ X_0, \dots, X_n \\ S_0, \dots, S_n}} P(S_f = 1, \underline{F}, \underline{C}, \underline{X}, \underline{z}, \underline{u}, \underline{t}) = \sum_{\substack{F_0, \dots, F_n, \\ C_0, \dots, C_n \\ X_0, \dots, X_n}} P(S_f = 1, \underline{S} = \underline{1}, \underline{F}, \underline{C}, \underline{X}, \underline{z}, \underline{u}, \underline{t}) \quad (2.21)$$

The same principle also applies to F_i and C_i , because the term $P(S_i = 1 | F_i, C_i)$ equals zero everywhere except at $F_i = 1, C_i = 1$. After we eliminate all the F_i and C_i , the nominator of Equ. 2.19 equals to

$$\sum_{\substack{F_0, \dots, F_n, \\ C_0, \dots, C_n \\ X_0, \dots, X_n \\ S_0, \dots, S_n}} P(S_f = 1, \underline{F}, \underline{C}, \underline{X}, \underline{z}, \underline{u}, \underline{t}) = \sum_{X_0, \dots, X_n} P(S_f = 1, \underline{S} = \underline{1}, \underline{F} = \underline{1}, \underline{C} = \underline{1}, \underline{X}, \underline{z}, \underline{u}, \underline{t}) \quad (2.22)$$

Substituting the term after sum operation by 2.20, we obtain

$$\begin{aligned}
& \sum_{X_0, \dots, X_n} P(S_f = 1, \underline{S} = \underline{1}, \underline{F} = \underline{1}, \underline{C} = \underline{1}, \underline{X}, \underline{z}, \underline{u}, t) \\
&= \sum_{X_0, \dots, X_n} \prod_{i=1:n} P(F_i = 1 | X_i, u_i) \cdot P(C_i = 1 | u_i, t) \cdot P(z_i | u_{i-1}, X_i) \\
&\quad \cdot P(F_0 = 1 | X_0, u_0) \cdot P(C_0 = 1 | u_0, t) P(z_0 | X_0) \\
&\quad \cdot \underbrace{P(S_i = 1 | F_i = 1, C_i = 1)}_{=1} \cdot \underbrace{P(S_f = 1 | S_1, \dots, S_n = 1)}_{=1} \\
&\quad \cdot P(X_i) \cdot P(u_i) \cdot P(X_0) \cdot P(u_0) \\
&= \prod_{i=1:n} \sum_{X_i} \underbrace{P(F_i = 1 | X_i, u_i) \cdot P(C_i = 1 | u_i, t) \cdot P(z_i | u_{i-1}, X_i) \cdot P(X_i)}_{\phi_i(u_i, u_{i-1})} \\
&\quad \cdot \underbrace{P(F_0 = 1 | X_0, u_0) \cdot P(C_0 = 1 | u_0, t) P(z_0 | X_0) \cdot P(X_0)}_{\phi_0(u_0)} \\
&\quad \cdot \underbrace{P(u_i) \cdot P(u_0)}_{\text{constant}} \\
&\propto \prod_{i=1:n} \phi_i(u_i, u_{i-1}) \cdot \phi_0(u_0).
\end{aligned} \tag{2.23}$$

Incorporating the renormalization constant in the denominator of Equ. 2.19, the conditional probability of grasping success is proportional to the multiplication of a set of factors ϕ_i .

$$\begin{aligned}
& P(S_f = 1 | \underline{Z} = \underline{z}, \underline{U} = \underline{u}, \underline{T} = \underline{t}) \\
&\propto \prod_{i=1:n} \phi_i(u_i, u_{i-1}) \cdot \phi_0(u_0).
\end{aligned} \tag{2.24}$$

As same as in the previous grasping scenario, we aim to find u_0^* to u_n^* to maximize the conditional probability. However, due to the temporal property of the model, we can not condition all the measurements \underline{Z} at once. The conditioning of the measurement depends on what control trajectory u is sent to the robot at the last time stamp. Therefore, the optimal control sequence $u_0^* : u_t^*$ can only be found in an iterative fashion. We propose to use the following paradigm to find optimal controls for the robot.

2.5 Summary

In this chapter, we first propose a unified model based on Bayesian networks to describe grasping problems. The network we define for the grasping problem factorizes the joint distribution into smaller conditional probability distributions. By conditioning on a subset

Algorithm 1 Paradigm for iteratively finding optimal control trajectory

```

1:  $u_0^* = \arg \max_{u_0} \phi_0(u_0)$ 
2: execute  $u_0^*$  on robot
3: wait for a time step
4: for  $i$  from 1 to  $n$  do
5:    $u_i^* = \arg \max_{u_i} \phi_i(u_i, u_{i-1}^*)$ 
6:   execute  $u_i^*$  on robot
7:   wait for a time step
8:   if pre-touch configuration is reached then
9:     return success
10:  end if
11: end for

```

of variables such as measurement and robot controls, we can compute the conditional probability of success. The goal is to determine the controls that maximize the conditional probability of success. Then we derive how to compute the conditional success probability in three different grasping scenarios. The focus of each scenario is different. In the first scenario, we consider grasping known objects under task constraint. As the object is known to the robot, we can assume that an object model for detection and localization exists. Furthermore, a set of grasp configuration can be defined in advance. As a result, the force closure conditions are always fulfilled during grasp executions. The focus in this scenario is how to model the task constraint satisfaction. In the second scenario, we consider the problem of grasping unknown objects. The focus is how to model the probability of force closure and how to model the object. In the third scenario, we consider the problem that the robot only has limited sensing capability. To reduce perception uncertainty, the model is extended in a temporal fashion to allow continuous measurement integration. The focus is how to find optimal controls online in each iteration. In the following chapters, we will elaborate how the grasping problem is solved in each grasping scenarios.

3 A Skill Framework for Assembly Tasks Using Mobile Manipulator

Grasping is usually associated with task intentions. The content of a task determines the grasping strategy needed. For example, a bottle can be both successfully grasped from the side or from the top. When pouring is the task intention, the bottle has to be grasped from the side. On the contrary, when the bottle is intended to be placed into a beer box, the right way to grasp the bottle is grasping from the top. In another example, an assembly task usually requires the parts to be putted together by following a certain direction, so how to (re)grab the parts before assembly must be considered. In this chapter, we introduce how to generate a suitable grasping strategy and corresponding motion trajectories in this scenario.

3.1 Introduction

Assembly is the act of combining components together in a logical sequence. Assembly widely exists in the chain of production. Conventional assembly tasks are either done manually or completed through a specially designed automation equipment. The traditional assembly automation is mainly aimed at mass production. The product usually has a long life cycle. The product parts and assembly processes do not need to be changed in a long time. In order to meet the demand of production with small lot size, the traditional assembly automation is unable to meet the rapidly evolving product demand and labor costs. Moreover, the traditional assembly process is automated by designing special fixtures, grippers, even pre-programmed motion of robots. It requires significant effort to apply change to the assembly automation process.

General purpose mobile manipulators can help to increase the flexibility of assembly automation because it can be applied to various environments and perform different tasks. A general purpose mobile manipulator is an autonomous system, equipped with a perceptual system for sensing the environment, as well as an arm and a gripper for manipulating objects. Compare to a fix mounted robot arm, a mobile manipulator requires minor effort for setting up and can reach more area of space by moving itself. Especially in the situation where the workspace is limited, it can choose the best suitable position to perform manipulation tasks. Based on its flexibility, we chose it as the platform to solve assembly problems.

The major question we want to answer is how to realize an adaptable and parameter-

izable assembly process so that it can be easily adapted to different assembly sequence. Since in assembly tasks grasping is always related to a task, thus the problem of grasping under task constraint must also be addressed. For this purpose, we propose a skill framework and a probabilistic model to solve the problem. The main idea is to decompose the assembly process into hardware independent assembly steps. Each step is an instruction which describes the relationship between the parts being assembled. One example can be ‘Insert part A on the top of part B’. A user is only required to provide such instructions and their execution sequence to complete an assembly task. Thus, the sequence of assembly and the parameters of the assembly parts can be easily changed or reconfigured. The user does not need to worry about how the instructions are performed, such as the motion of the robot, the positions of the parts, etc., because they are all handled by the proposed skill framework. It encapsulates these instructions using a set of skills which are reusable across different instructions. The skills implement the actual planning and execution logic.

For the purpose of demonstration, two types of instructions are provided, later we call them high-level skills. One is ‘Pick and Reconfiguration’ and another is ‘Pick and Insert.’ In both instructions, the problem of grasping under task constraint has to be considered. In ‘Pick and Reconfiguration’, the robot grasps a part from a support plane and then place it back in the desired pose. Since the desired pose is different than the original pose, it has to be ensured that the robot does not collide with obstacles both for the pickup posture and the place posture. In ‘Pick and Insert,’ the robot grasps a part from the support plane and then insert it into another part. The robot should guarantee the first part is picked up in the right way that the gripper does not cover the insert area. Otherwise, the insert operation can not be conducted. In both high-level skills, the feasible grasp postures, as well as the robot configurations for grasping the parts have to be determined. We propose a sampling algorithm to find the desired configuration which maximizes the probability of task satisfaction. As a by-product, it also solves the optimal placement of the robot that during the execution of either high-level skill, the traveling distance of the robot is minimized.

We evaluate the proposed skill framework for assembly using a 3D printed toy radio assembly shown in Fig. 3.1. The result shows that with the proposed algorithm, the task constraints of the grasps can be satisfied. Besides, the framework allows not only the parts being placed in various initial configurations without any fixtures, but also makes it possible for flexible parameterization of assembly steps and configuration of their execution sequences.

3.2 Related Work

In this chapter, the major problem is how to calculate task constraint grasp configurations with application to the assembly problem. The grasp configurations have to satisfy task constraints in the following operation prior to grasping. In the context of assembly problem,

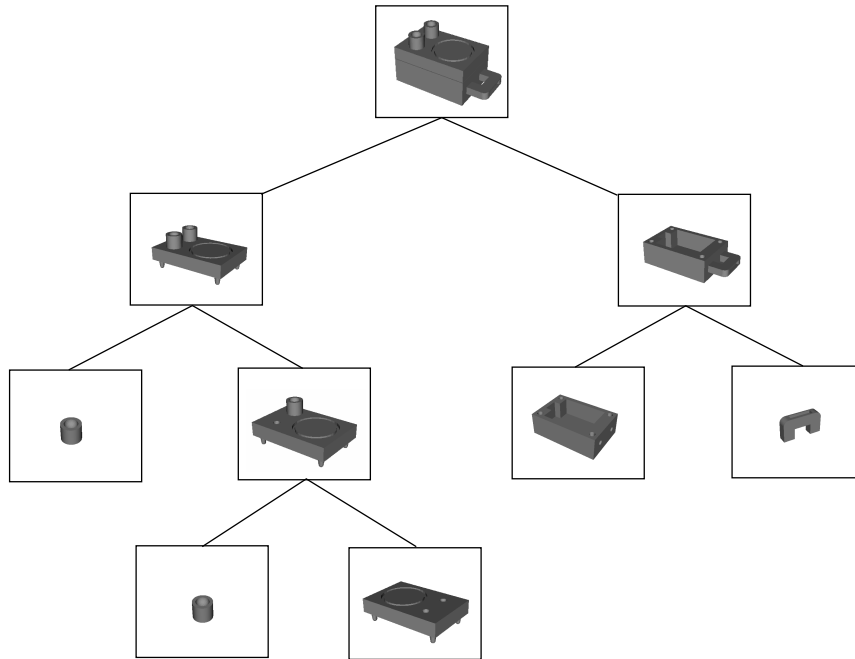


Fig. 3.1: Assembly of a 3D printed toy radio.

we need to compute a viable configuration to satisfy a task operation such as ‘peg-in-hole’ or ‘pick-and-place’ [47]. Here we provide a review of the methods from two perspectives. The methods from the first perspective concentrate on grasp synthesis when the task information is given. Given an object, a gripper and their associated task, how the object should be grasped. The methods from the second perspective concentrate on how assembly problems are solved by the robots. Here we especially focus on the methods which use a skill concept to tackle the problem.

3.2.1 Methods of grasp synthesis considering task information

Although generic grasp synthesis has been studied over the last decades, only in recent years task oriented grasping has been identified as an significant problem. Researchers usually have their own representation to describe the task information. Dang et al.[18] proposed a semantic affordance map which links object geometry to a set of pre-defined semantic grasps that are suitable for conducting a task. Berenson et al.[3] proposed task space regions to represent tasks. The regions form a continuous space in which the task constraints are satisfied. The intersection of task space region and a region of pose uncertainty defines the viable goal region that a robot should reach. Borst et al.[9] proposed task wrench space to encode the task information. They demonstrated how to integrate this concept in a well-known grasp quality measure. Another approach to encode task information is through human guidance. In [1], human planned grasping is studied by an approach called Physical Human Interactive Guidance. The result shows that the grasps produced by this method perform better than grasps generated through a state-of-the-art

automated grasp planner. Lin et al.([82], [83]) proposed a grasping planning approach that extracts grasp strategies from human demonstration. The grasp strategies not only represent important human grasp intentions, but also provide meaningful constraints on hand posture and wrist position. The choice of object representation is also different in

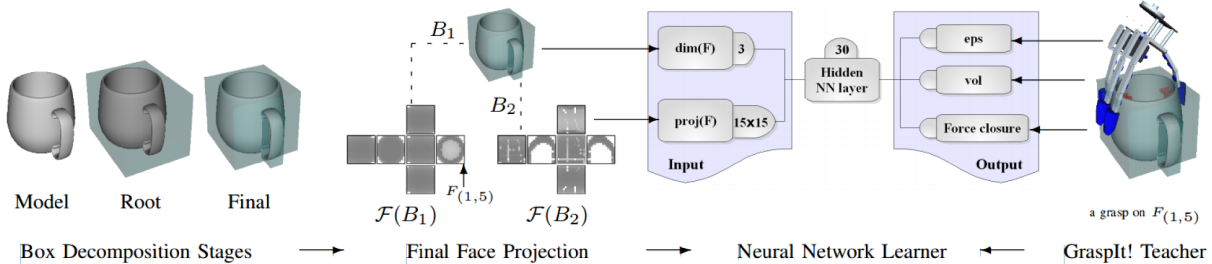


Fig. 3.2: A grasp synthesis method based on bounding box representation proposed by [56]. Objects are first decomposed into minimum volume bounding box. The parameter of the bounding box as well as grasps generated by a simulator are fed into a Neural Network for learning the quality of a grasp.

several approaches. Huebner et al.([56],[55],[38]) uses simplified bounding box to represent an object. The task-relevant information is labeled on each side of the bounding box. This information are used for calculating task oriented grasps (Fig. 3.2). Many researchers ([18],[9]) use a detailed mesh to represent objects, since a bounding box approximation is not always sufficient accurate and powerful to represent the complex shape. The object can also be represented by its category. Bohg et al.[8] proposed an approach towards autonomous grasping based on the category of the objects and a given task. Their approach allows task-specific grasp experience to be transferred between objects of the same category (Fig. 3.3). In ([123],[124]), Song et al. provide a learning based approach to the learning of a distribution to predict whether a given grasp satisfying the task constraints or not. To achieve a task specific grasp, one can either directly execute the planned grasp, using pre-grasp manipulation or through tactile exploration. Chang et al. [13] proposed to use pre-grasp rotation to change the orientation of an object prior to acquire a task oriented grasp. In [51], Ksiao et al. proposed to use tactile sensing to achieve a task oriented grasp by selecting among grasping and information-gathering trajectories. The tactile exploration is used for localization of objects in their work.

Some tasks require precision in execution. In these case, the robot has to reason about the in-hand pose of the object after the object being grasped. Paolini et al. [104] proposed a data-driven framework for post grasp manipulation. Their approach builds a statistical model of grasp state conditioned on sensor values. Based on the learned model, the robot is able to estimate the in-hand pose distribution of the object and choose the action which maximizes the probability of task success.

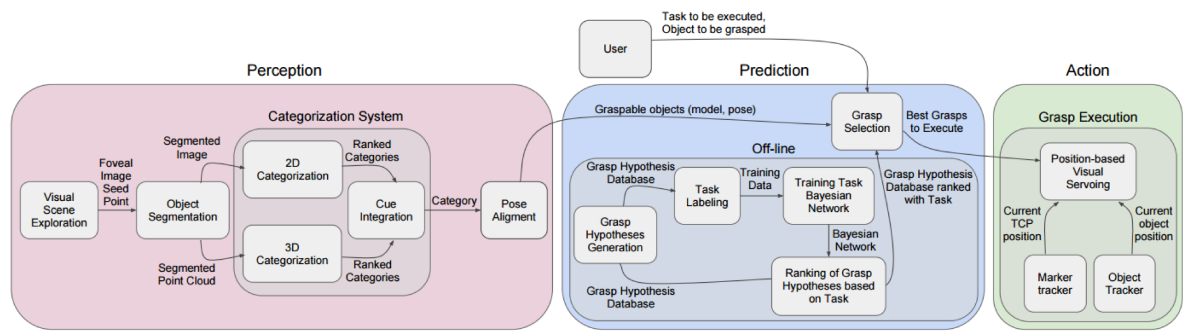


Fig. 3.3: A grasp pipeline for generating task-oriented grasp proposed by [8]. Objects are represented by their categories. The categorization of an object is based on their perception system. The result of perception as well as the user input are fed into the grasp prediction for generating grasps. The best grasp suited for the given task is selected by an offline-trained Bayesian Network. Visual servoing is used to execute the selected grasp.

3.2.2 Methods of solving assembly problem using skill concept

Methods based on this concept decompose an assembly task into a set of skills. These skills are parameterizable and adaptable for various assembly tasks so that an assembly program can be configured in a flexible manner. Solving assembly problem using a skill concept contains several aspects, such as the design of skill architecture, task decomposition, and skill acquisition. In the following, we provide a short review of the existing methods and focus on the above mentioned three aspects.

One of the earliest work that proposes a skill architecture goes back to the middle 90's. Morrow et al. [94] proposed a sensorimotor primitive layer which bridges the gap between the robot/sensor system and a class of tasks. The sensorimotor primitives are parameterized, domain general command which integrate sensing and action and can be applied to many skills within a task domain. Later they extend the sensorimotor primitives to task primitives for composing robot skills [93]. Visual servoing combined with force sensing ([98], [85]) is one of the most widely used sensorimotor primitives to conduct manipulation tasks, such as peg-in-hole [10], kitting [47], etc. Kröger et al. [70] proposed a generic framework based on manipulation primitives for sensor-based motion control. They introduced an adaptive selection matrix for hybrid switched-system control, which allows any number and any type of sensors into one control system.

Later, as the assembly tasks to be executed becomes more complex, so decomposing a given task into skill primitives becomes a demand. Mosemann et al. [95] proposed a method to decompose complex sequences of assembly operations into skills. The method first analyzes hyperarcs of the underlying AND/OR graphs which is generated by an assembly planner. The result represents the sub-tasks to be conducted by the robot. The sub-tasks are then classified by features like local depart spaces, symbolic spatial relations, and tools. The classification allows the sub-task being further decomposed into a set of skill

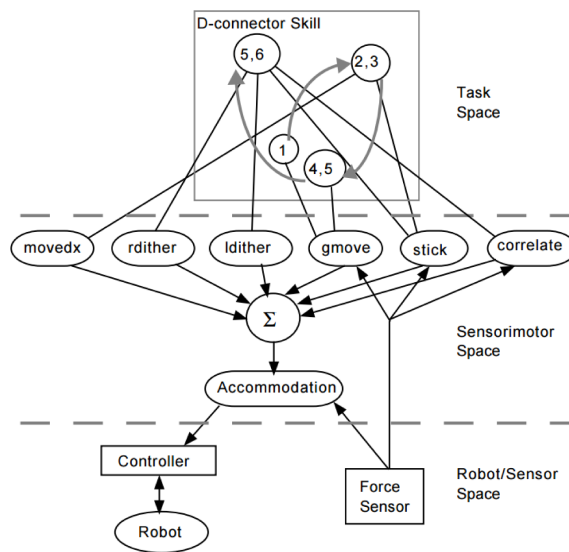


Fig. 3.4: An example skill architecture proposed in [94]. The sensorimotor primitives are served as middle layer to connect low-level robot/sensor space to high level task space. The sensorimotor primitives can be reused within a D-connector skill, which is implemented by a Finite State Machine (FSM).

primitives. Pederson et al. [106] presented the concept of general, self-asserting robot skills for manufacturing. They demonstrated how a relatively small set of skills can be derived from current factory worker instructions, and how these can be transferred to industrial mobile manipulators.

Recently, more work focus on how to acquire the skills more intuitively, since designing a generic applicable skill normally requires expert knowledge [48]. Learning by demonstration is one of the promising methods that reduce the effort of developing a skill. Dillmann et al. [25] followed programming by demonstration paradigm and provided a system approach which allows integrating the overall process of skill transfer from a human to a robotic manipulation system. Skubic et al. [122] proposed a method to teach robots force based assembly skills from human demonstration. They modeled sensorimotor skills using a hybrid control model, which provides a mechanism for combining continuous low-level force control with higher-level discrete event control. Methods for acquiring assembly skills can also be found in the context of human-robot collaboration. Ogata et al. [101] proposed an approach to human-robot collaboration based on quasi-symbolic expressions. A recurrent neural net with parametric bias model is used to acquire the skills. Wallhoff et al. [129] presented a hybrid assembly station, in which an industrial robot can learn new tasks from worker instructions. The learned task can be performed by both the robot and the human worker together in a shared workspace.

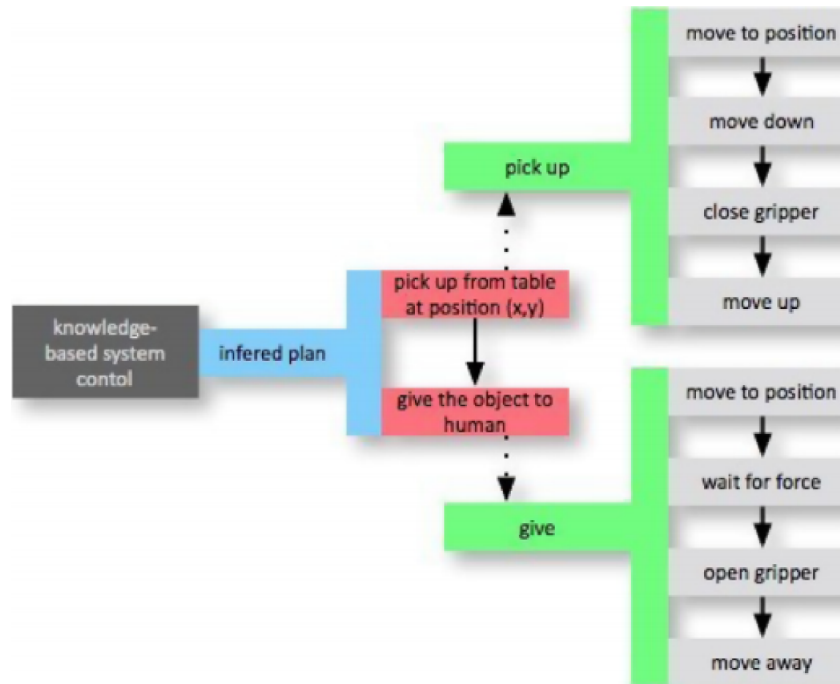


Fig. 3.5: An example of skill architecture proposed in [129]. A knowledge-based system control module infers the required skills to execute a given task. The skill acquisition is based on multi-modal inputs. A skill is learned by teaching the correct sequence of atomic actions.

3.3 Contributions

We propose a skill-based framework for executing assembly tasks with mobile manipulators. The proposed framework provides a new way for mobile manipulation based assembly. Most previous works put the main effort on the design of skill. However they ignore the pre-condition and the post-condition for executing a skill. Different from the previous work, we follow a new design principle that allows the execution of skills are independent of each other. Namely, the postcondition of executing one skill fulfills the pre-condition of executing another skill. We demonstrate two examples of how to realize an insert and a place skill respectively. Both skills encapsulate an object localization and a grasping low-level skill so that a complete sequence including searching, localizing, picking up, inserting, releasing is realized. In addition, we provide a very intuitive way to parametrize not only for the skill itself but also for the sequence of executing the skills.

In both skills, the problem of generating a task oriented grasp is addressed. The goal of most previous work is only to plan a set of gripper configurations. Besides generating the gripper configuration, we also consider the whole robot configurations including the base position and the arm position of a robot. In this way, we can use the redundant degrees of freedom to optimize the performance of executing a task. For instance, in the context of performing a skill, minimum platform traverse is chosen to evaluate the

efficiency of a task. We solve this problem by a novel arm-base coordinated modeling approach and a sampling technique. The modeling approach allows calculating whole body inverse kinematic solutions while the sampling technique is used to select the best solutions which represent the most suitable grasp configuration for the task.

The proposed approach is generic and is hardware independent. Thus, we can apply the approach to any type of mobile manipulator. The parametrization of the skill framework is intuitive and flexible. Therefore it is suitable for executing a wide range of assembly tasks.

3.4 Framework description

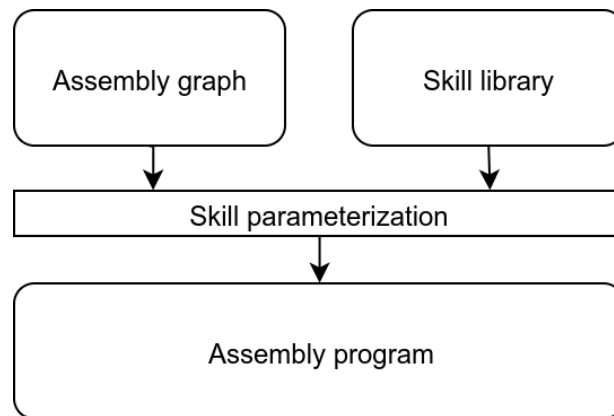


Fig. 3.6: The structure of the framework

The general structure of the skill framework is depicted in Fig. 3.6. The input of the framework is an assembly graph. The assembly graph contains all the information needed to perform an assembly task. As depicted in Fig. 3.7, the basic structure of an assembly graph is a binary tree. The leaves of the tree represent the basic parts to be assembled. In each leaf, the properties of the basic parts such as the bounding box, the mesh model and the stable orientations are stored. The internal nodes of the tree represent an intermediate assembly state, while the root of the tree is the final assembled product. The siblings are connected with an additional node which stores geometric relationship about how they should attach to each other. The node also specifies which operation should be performed. This can be any operation needed to bring the parts together. Example operations are insertion, screwing, welding, etc. Finally, a list of assembly steps can be generated by extracting the pairs of siblings from the bottom to the top of the graph.

The information of an assembly step need to be transformed into a sequence of actions that a robot can perform. In our framework, the sequence of actions are generated by the skills which are provided by the skill library. As shown in Fig. 3.8, two types of skills are defined in the library: high-level skills and low-level skills. A high level skill specifies a high-level task the robot need to perform. The high level skills are reusable functions

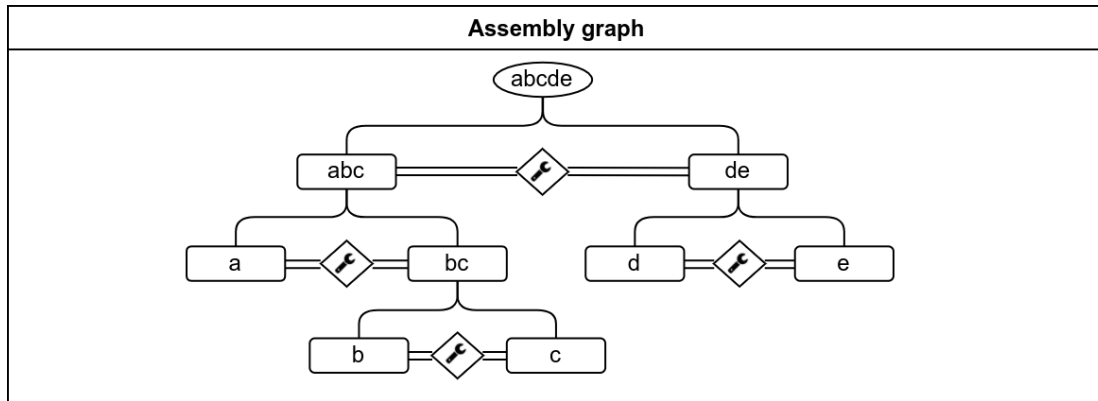


Fig. 3.7: Assembly graph

which provided by the skill library. In order to make the high-level skill independent of the assembly parts, we need further elements to make high-level skill reusable and configurable. These elements are called low-level skills. Low-level skills are elementary functions or actions that are reusable across different high-level skills. Example of low-level skills can be ‘detect object pose’, ‘open/close gripper’, ‘plan trajectory’, etc. To realize a specific high-level skill, selected low-level skills are organized in a finite state machine. The finite state machine realizes the internal logic and action sequences.

The assembly program is generated by parametrization of the high-level skills in each assembly step. Which high-level skills to choose and in which order to perform depend on the operation type of the assembly step and the capability of the robot. For demonstrating the advantage of the framework on an one-arm mobile manipulator, two types of high level skills are elaborated in this work. They are used to handle a commonly used operation in assembly, namely insertion. The one we call it ‘Pick and Reconfigure’ skill (P.R) and the other we call it ‘Pick and Insert’ skill (P.I). The P.R skill aims to change the orientation of a part, so that the desired surface lies in favor of the next task. To realize this skill, a robot must have a gripper as its end-effector. The P.I skill allows a robot to pick another part and perform the actual insert action.

3.5 High-level skills

In the following, we elaborate how these two skills are realized. The P.I. skill is designed to perform an insert operation between two assembly parts, while P.R. skill is designed to reconfigure the pose of an assembly part. The P.R. skill is necessary because in some special cases, P.I. skill can not be executed directly. The direction of the insert motion does not meet a required constraint. For example, the insert motion is not perpendicular to a table top. In this case, P.R. skill has to be executed first to reconfigure the pose of the assembly part so that the object pose satisfies the insert condition.

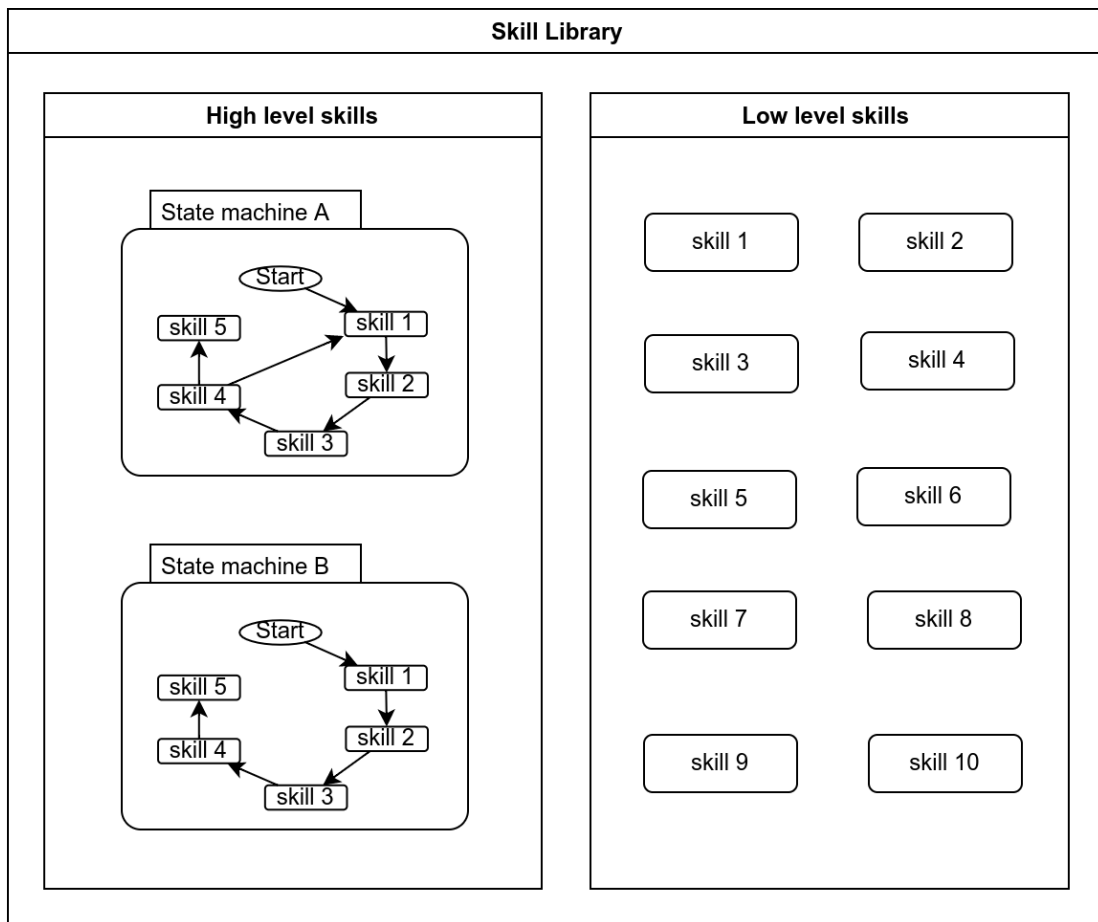


Fig. 3.8: Skill library

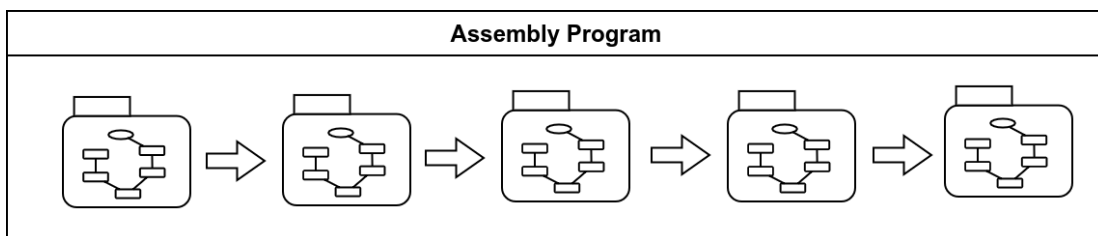


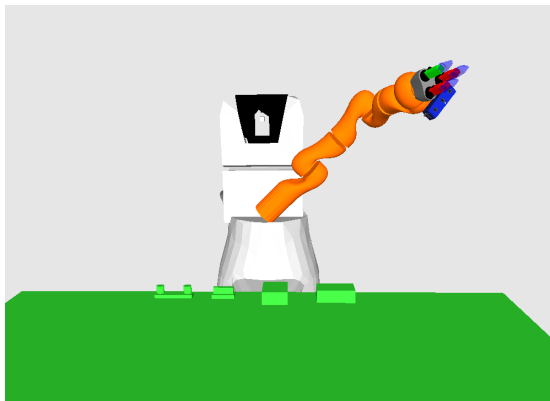
Fig. 3.9: The assembly program

3.5.1 Pick & Reconfigure

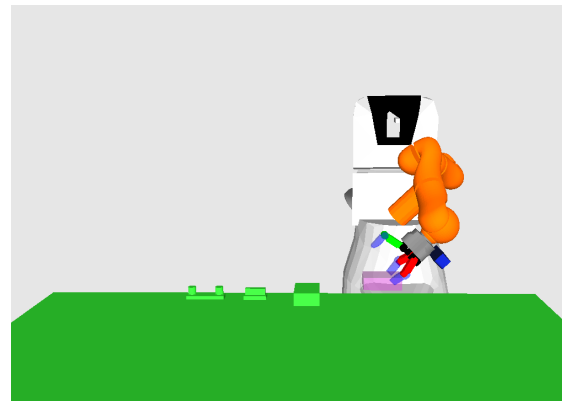
This high-level skill is realized as a hierarchical state machine. It includes seven low-level skills which are realized as states. The diagram of state transitions is visualized in Fig. 3.10. The P.R. begins with the ‘Plan platform locations’. The skill calculates the optimal placement of platform for pick and place location so that the distance of platform movement in between is minimized. The output of this skill is the selected grasp configuration along with the IK solution both at the pick and the place location. After planning the base platform location, the robot performs ‘Grasp Object.’ The ‘Grasp Object’ has a hierarchical structure, which contains additional low-level skills. After executing ‘PickObject’, the robot grasps an assembly part and holds it within the gripper (see. Fig. 3.11b). The next skill to execute is called ‘Move Joint Goal’, which plans and executes the robot trajectory in joint space. The state that uses this skill is ‘MoveToPrePlace’, which moves the robot to a so-called pre-place configuration (see. Fig. 3.11c). After the robot is succeeded to move to the pre-place configuration, the state ‘PlaceDown’ which uses the ‘MoveCartesian’ skill, plans and executes a linear trajectory towards a given pose configuration. In this case, the robot moves in a linear path to place the assembly part on a table-top (see. Fig. 3.11d). After ‘PlaceDown’ is executed, the robot starts to execute the ‘Control gripper’ skill, in which the robot releases the object by state ‘ReleaseObject’. The next two skills are ‘DetachObject’ and ‘UpdateObjectPose’, which update the planning states in a motion planner so that the assembly part is not considered as a part of the robot later. The last state ‘MovetoPrePlaceLinear’ uses ‘MoveCartesian’ skill again to moves the gripper linearly back to the pre-place pose 3.11e. Fig. 3.11 illustrates the state flow of this skill.



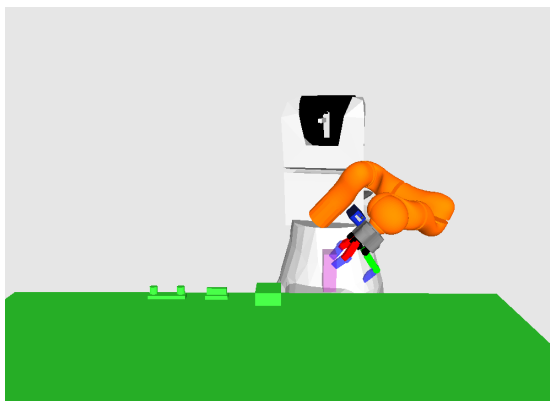
Fig. 3.10: The diagram of state transitions for 'Pick & Reconfigure' skill. The low-level skill 'Plan platform locations' and 'Grasp Object' have an hierarchical structure, which includes further low-level skills organized by a state machine. In total seven low-level skills are used in the high-level skill 'Pick & Reconfigure'. The low-level skill 'Move Cartesian Goal' is used twice for 'PlaceDown' and 'MoveToPrePlaceLinear'.



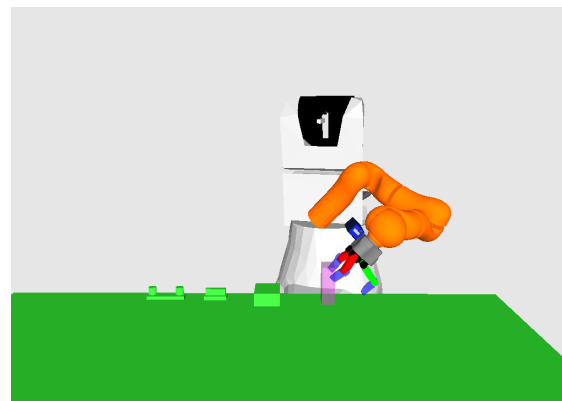
(a) Initial state before executing the P.R. skill.



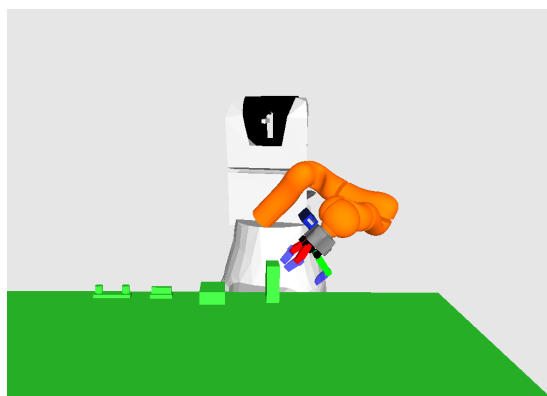
(b) After executing the 'PickObject' state, an assembly part is picked by the robot. The color of the assembly part changes from green violet, which indicates that the assembly part is attached to the robot.



(c) The robot reconfigures the pose of the assembly part and moves to the pre-place location by the state 'MoveToPrePlace'.



(d) The robot executes the 'PlaceDown' state to put the assembly part down on the table.



(e) The color of the assembly part changes back to green, which indicates that the robot releases and detaches the assembly part after executing the 'DetachObject' state. To finish the P.R. skill, the robot moves in a linear path back to the pre-place configuration.

Fig. 3.11: An execution sequence of the Pick & Reconfigure skill

3.5.2 Pick & Insert

The Pick & Insert skill is also realized as a state machine. Fig. 3.12 illustrates the transition diagram of the states. The state machine also starts with ‘Plan platform locations’ skill, which calculates the optimal placement of platform for pick locations and insert locations. After the optimal platform location is computed, the state machine moves to ‘PickObjectA’ state, which represents the ‘Grasp Object’ skill. The successful execution of the skill leads to that the assembly part to be inserted is grasped by the robot (see Fig. 3.13a). If the ‘PickObjectA’ succeeds, the state machine transits to the ‘ComputePreinsertIK’ state, which represents the skill ‘Compute pre-insert IK’. This skill calculates an IK solution of the arm, so that it can reach a so-called pre-insert pose. The pre-insert pose is defined as 5 cm linearly opposite from the insert pose. The platform positions for insert operation calculated by ‘Plan platform locations’ is then used as the input for the next state ‘LocateObjectB’. This state represents a hierarchical low-level skill called ‘Move and locate’. The goal of this skill is to jointly move the mobile platform and arm of the robot to the configuration where it can first perceive and then reaches the target positions for insertion (Fig. 3.13b). After the pose of target assembly part is located, the ‘Compute Insert Pose’ skill is executed to update the desired gripper pose relative to the insert point. A successful execution of this skill leads to the states in which the robot actually performs the insert action in sequence, the state ‘MoveToPreInsert’ followed by the ‘MoveToInsert’ state. Former uses the ‘Move joint goal’ skill, while latter uses the ‘Move cartesian goal’ skill. After successful execution of both skills, the assembly part which grasped by the robot is by the moment inserted to the target point (Fig. 3.13c). Then the robot releases the assembly part by the state ‘Release Object’, which is a ‘Control Gripper’ skill. After the assembly part is physically released from the gripper, the ‘Detach Object’ tells the motion planner to separate the assembly parts from the robot for the following actions. ‘Remove Parts’ and ‘UpdateSubProduct’ update the collision models of the ‘Pick & Insert’ skill. The collision models of the both assembly parts, which are involved in this skill, are merged together. The last state ‘Lift’ plans for a lift action for the robot so that the gripper moves away from the assembled part (Fig. 3.13e).

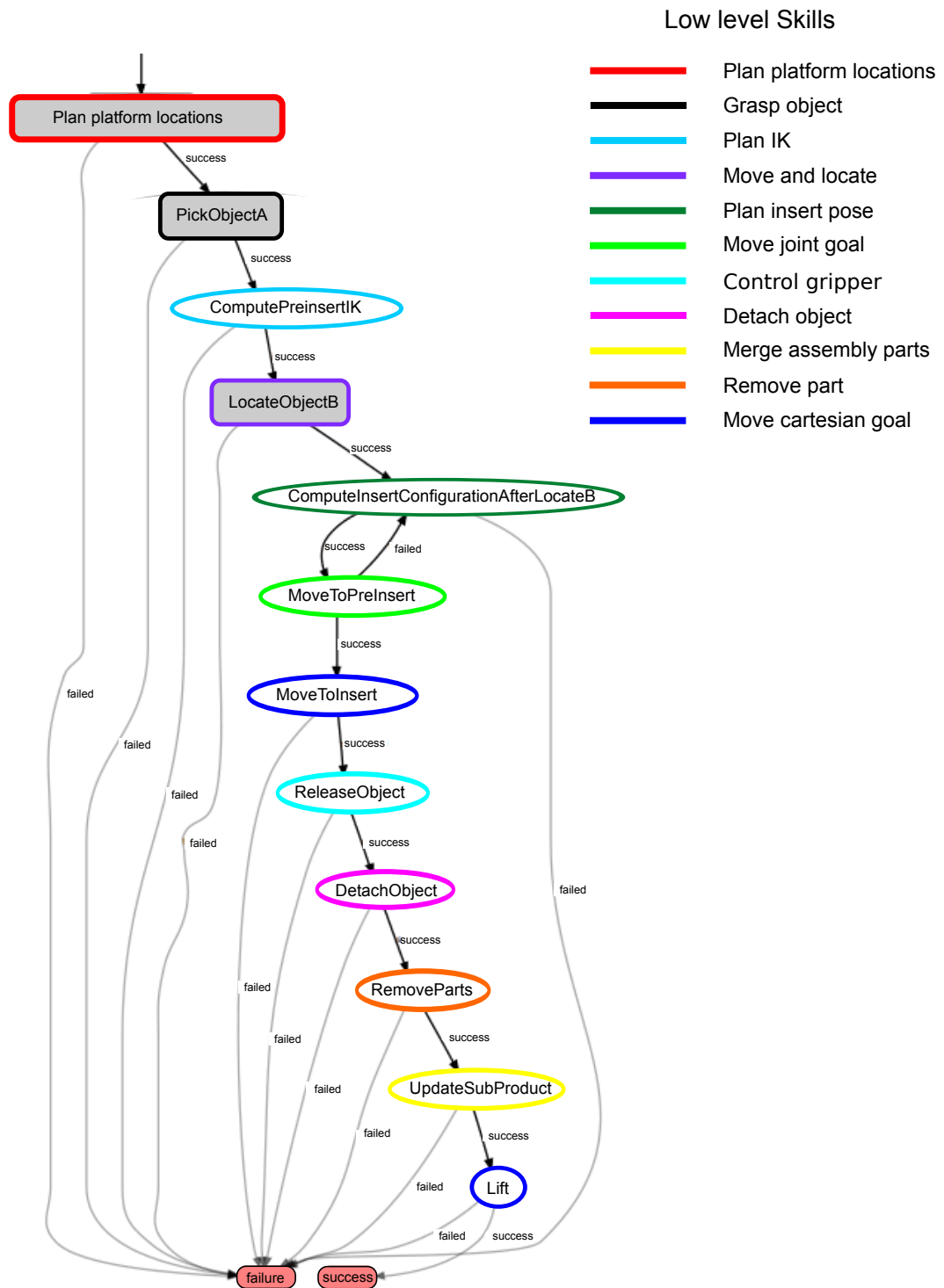
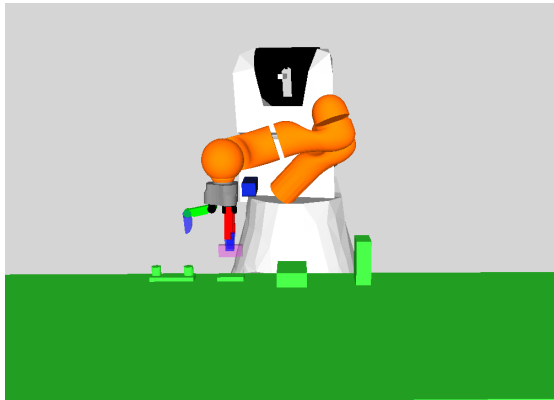
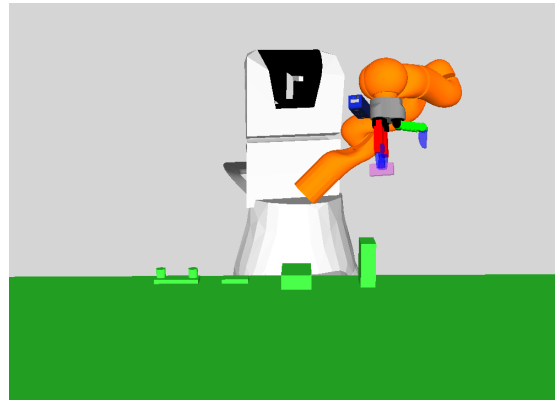


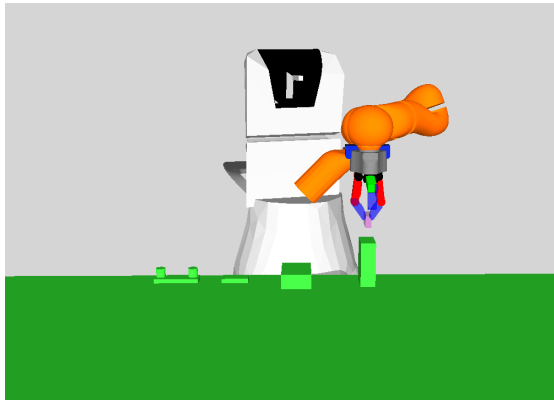
Fig. 3.12: The state transition diagram of Pick & Insert skill.



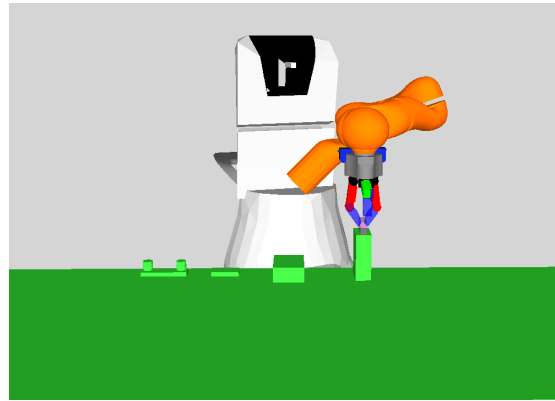
(a) The robot picks up an assembly part after executing the 'Grasp Object' skill.



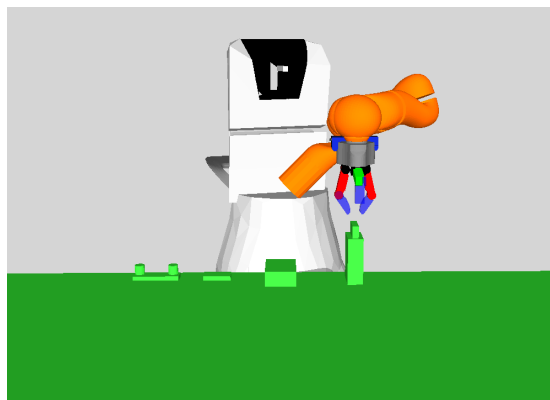
(b) The robot moves to the arm and platform jointly to the configuration where it can perceive the object with its in-hand camera and also reaches the insert configuration.



(c) After executing the 'MoveToPreinsert' state, the robot moves its arm to a so-called pre-insert configuration.



(d) The actual insert action happens in the 'MoveToInsert' state, in which the assembly part is inserted in another one.



(e) The robot releases the assembly part and moves back to the pre-insert configuration. The collision models of both assembly parts are merged into a new one.

Fig. 3.13: An execution sequence of the Pick & Insert skill

3.6 Low-level skills

low-level skills serve as basic elements to build a high-level skill. All the logic and execution sequence of low-level skills are handled by their corresponding high-level skill. In this way the specification of an assembly program is simplified. In total 11 low-level skills are implemented to realize the ‘Pick & Insert’ and ‘Pick & Reconfigure’ skill in our framework. Among them, some skills implement planning algorithms such as ‘Plan platform locations’ skill, while other skills are used to perform a physical action, such as ‘Grasp Object’ skill. A list of low-level skills is explained as follows,

- Move joint goal: This skill implements a collision-free movement of the robot from its current joint configuration to the desired goal configuration. We use ‘*Moveit*’ to manage the internal planning environment and RRT-connect to plan trajectories in joint configuration space.
- Move cartesian goal: This skill is used when a linear movement of the gripper is required, such as approaching a pre-touch configuration, lift up an object, etc.
- Control gripper: This skill is used to control the desired gripper configurations when the robot grasps a part or release a part.
- Plan IK: This skill is used to solve an inverse kinematic query.
- Detach object: This skill is used after the object is released from the gripper. It gives a signal to the internal motion planner that the object is not required to be considered as a part of the gripper.
- Merge assembly parts: This skill is used when a part is inserted into another part. After an insert operation, the collision models of two separated parts are treated as a whole in the planning environment.
- Remove part: This skill is used to remove the collision model of a part out of the planning environment.
- Update object: This skill is used when the pose of a part is reconfigured. This skill updates the pose of the part.

Among all the skills we proposed, three of them have a hierarchical structure, which means they depend on other low-level skills to realize its functionality. In the following, we will elaborate the following hierarchical skills ‘Plan platform locations’, ‘Move and locate’ and ‘Grasp object’.

- Plan platform locations

This skill addresses one of the aforementioned challenge: Optimal placing of a mobile manipulator. For the high-level assembly skills we have proposed, it is required to plan two platform locations, one for picking an assembly part and the other for placing or inserting the part. Ideally, the distance between them should be as close as possible, so that the distance that a robot traverses can be kept in minimal. Additionally, two constraints should be satisfied at the planned locations. First, joint configurations of an arm must stay away from the singularity. This condition guarantees that a valid linear approach motion for grasping or insertion exists. Second, the part must lie inside the field of view of the robot’s camera in each planned location. This constraint allows the robot to locate the pose of assembly parts before it performs a grasping or an insertion action.

Let’s specify p_1 as the gripper pose for picking an assembly part, p_2 as the gripper pose for either placing or inserting the assembly part. p_3 is the position of an assembly part to be picked, while p_4 is the insert position of a target object. The problem of planning platform locations can be formulated as follows. Given p_1, p_2, p_3, p_4 , the objective is to find the configurations of joints \underline{c}_1 that planned for the pick location and \underline{c}_2 that planned for the place or the insert location, that

$$\begin{aligned}
 & \text{minimizes} && |h(\underline{c}_1) - h(\underline{c}_2)| \\
 & \text{subject to} && g(\underline{c}_1) = p_1, \\
 & && g(\underline{c}_2) = p_2, \\
 & && \text{proj}(p_3, \underline{c}_1) \leq \underline{w}, \\
 & && \text{proj}(p_4, \underline{c}_2) \leq \underline{w}, \\
 & && \text{manip}(\underline{c}_i) \geq \epsilon, \quad i = 1, 2.
 \end{aligned} \tag{3.1}$$

where $h(\bullet)$ computes the virtual joint position of a platform. The cost function to be minimized is the distance between platform positions. $g(\bullet)$ calculates the forward kinematics of a given joint configuration. The equality constraints require that \underline{c}_1 and \underline{c}_2 are the inverse kinematic solution of p_1 and p_2 . The first two inequality constraint calculate a back projection from 3D points in the world to 2D points in the image. The 3D point here is the position of an assembly part. \underline{w} is the pixel width and height of a camera image. This constraint requires that the assembly part should lie in the field of view of the camera. In a head mounted camera setting, these constraints are mandatory, because the assembly parts have to be located in the configurations before the robot performs the actual grasping and insertion action. In the case that the camera is fix mounted to the gripper (eye-in-hand setting), these constraints are optional, because the robot can additionally move the arm to satisfy the constraints for perception and then perform the actual grasping or insertion. The third inequality constraint calculates the manipulability. It guarantees the robot can perform a linear motion to approach grasp pose or insert pose. $\text{manip}(\bullet)$ is a function which computes the distance from a given arm joint configuration to a singular

configuration. According to [133], the measure of manipulability can be calculated by

$$\text{manip}(\bullet) = \sqrt{\det(JJ^T)}, \quad (3.2)$$

where J is the Jacobian matrix of a given arm configuration.

According to Equ. 3.1, we formulate a non-linear non-convex constraint optimization problem. The dimension of the solution space is equal to 20. It is not trivial to solve this problem exactly. We propose to solve this problem as follows. First, we draw valid samples from the solution space, which must satisfy all the constraints. Second, we compute the cost of each sample and choose the sample which has the minimal cost as the solution. Algorithm. 2 gives details of the method. One challenge is how to draw samples efficiently since the unit time used to draw a valid sample affects the total computational time of the algorithm. In line 2 and line 6 of the algorithm 2, we use ‘ComputeIK’ function to generate the samples. The function is an inverse kinematic (IK) solver which computes the whole body joint configurations given an input pose. The output of the function automatically satisfies the first two equality constraints. Furthermore, since the robot has 10 DoF, it is possible to meet the additional inequality constraints with its redundancy. The technique we used to exploit the redundancy in the IK solver is called null space optimization ([81],[97]). Here each inequality constraints is defined as a cost function of joint positions. For the first two inequality constraints, the cost can be defined by the quadratic distance of a given point p_3 or p_4 to the optical axis of the camera. The sum of negative gradient of the cost functions determines the direction to minimize the total cost. The IK solver works in an iterative fashion. The solver begins with an initial estimate. In each iteration, an error \underline{e} is computed between the input pose and the pose computed by the initial estimate. The initial estimate $\underline{q}_{\text{init}}$ is then updated by

$$\underline{q}_{\text{init}} \leftarrow \underline{q}_{\text{init}} + \underline{q}_t + \underline{q}_p, \quad (3.3)$$

until the pose error \underline{e} is smaller than a threshold. Here \underline{q}_t and \underline{q}_p are computed by

$$\underline{q}_t = \underline{J}^+ \underline{e} \quad (3.4)$$

and

$$\underline{q}_p = (\underline{I} - \underline{J}^+ \underline{J}) \underline{\xi}, \quad (3.5)$$

where $\underline{J}^+ = \underline{J}^T (\underline{J} \underline{J}^T)^{-1}$ and \underline{I} is a unit matrix. $\underline{\xi}$ is the sum of negative gradient of the cost functions. If the output of ‘ComputeIK’ is verified by collision checking, the result is then considered as a valid sample. Since this is an iterative method to solve inverse kinematic, the convergence of the algorithm depends on the initial estimate $\underline{q}_{\text{init}}$. An initial estimate that is far from the solution influences speed and convergence of the algorithm. We propose to use an IK database to accelerate the algorithm. As soon as a valid solution is computed, we save it into the database. To compute the IK for a new pose, we randomly select a

solution from the database to initiate the algorithm. This method increases the probability of finding a solution, while reduces the total time of planning.

Algorithm 2 Plan platform locations

```

1: Input:  $p_1, p_2, p_3, p_4, n$ 
2:  $\underline{c}_1^* = \text{computeIK}(p_1, p_3)$ 
3:  $\underline{h}_1 = \underline{c}_1^*[1 : 3]$ 
4:  $\text{cost}^* = 10^5$ 
5: for  $i$  from 1 to  $n$  do
6:    $\underline{c}_2 = \text{computeIK}(p_2, p_4)$ 
7:    $\underline{h}_2 = \underline{c}_2[1 : 3]$ 
8:    $\text{cost} = |\underline{h}_1 - \underline{h}_2|$ 
9:   if  $\text{cost} < \text{cost}^*$  then
10:     $\underline{c}_2^* = \underline{c}_2$ 
11:     $\text{cost}^* = \text{cost}$ 
12:   end if
13: end for
14: return  $\underline{c}_1^*, \underline{c}_2^*$ 

```

- Move and locate

This low-level skill is designed for locating a target object by a hand-mounted camera prior to grasping or insertion. The state transition diagram of the skill is shown in Fig. 3.14. The skill begins with the ‘Get initial location’ state, in which the initial pose of the target assembly part is determined. Based on the initial pose, ‘Plan camera pose’ generates a camera viewing pose relative to the part. Then the robot uses ‘Move joint goal’ skill to move its arm and platform jointly to the viewing pose. If the viewing pose is not reachable by the skill, ‘Plan camera pose’ generates another viewing pose until the execution of ‘Move joint goal’ is successful. Since the virtual joint position of the platform is already planned by ‘Plan platform locations’, ‘Move joint goal’ guarantees after locating the assembly part, the robot can reach the grasp or insertion configuration. Finally, a ‘Locate object’ is conducted to refine the pose of assembly part.

- Grasp object

‘Grasp object’ is another hierarchical skill which is designed for picking up an assembly part. Similar to the previous low-level skills, it contains additional low-level skills to build up the internal logic. Fig. 3.15 illustrates the transition diagram of this skill. The state machine starts with the ‘Verify Object’ state, which checks whether the target assembly part is contained in the motion planning environment. If the verification succeeds, the state machine transits to the ‘Locate Object’ state, which is a ‘Move and locate’ skill. Since the object pose is relocated by this skill, ‘Compute pick condition’ calculates again the inverse kinematic for the robot to reach the pre-grasp pose. This pose is defined 10 cm

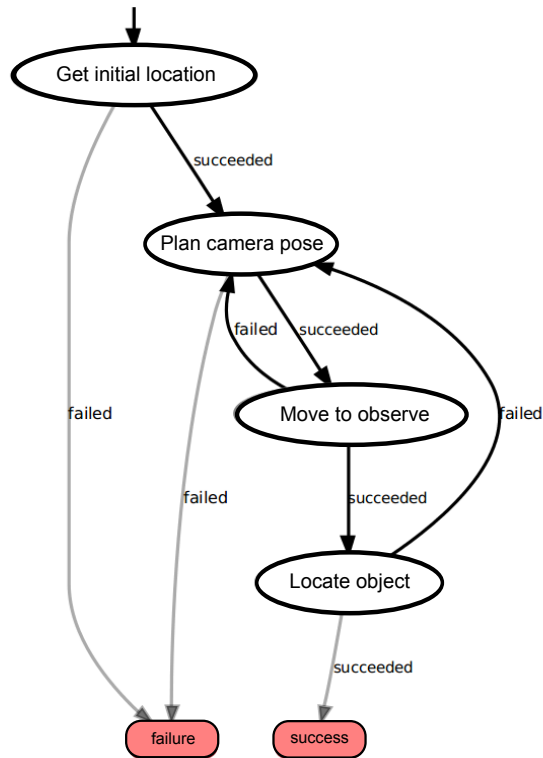


Fig. 3.14: The state transition diagram of 'Move and locate' skill.

back from the final grasping pose. If the inverse kinematic solution exists, the robot opens its gripper to the pre-grasp posture by 'Control gripper pre-grasp' state. The next skill to be executed is 'Move joint goal', in which the robot moves its arm to the IK solution computed for the pre-grasp pose. After the robot reaches the pre-grasp pose, the state machine transits to 'Grasp Strategy' state, which is a skill that evolves moving linearly towards the final grasp pose, then closing the gripper to reach force closure. The last state 'Lift object' is a 'Move cartesian goal' skill, in which the robot lifts the object from the working surface.

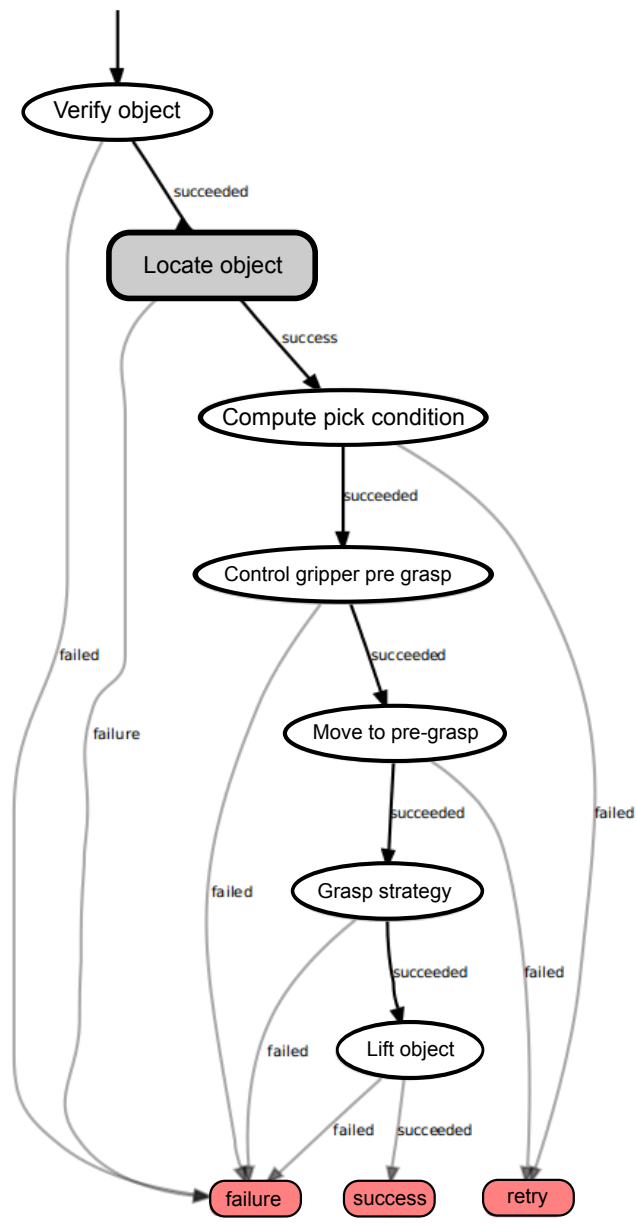


Fig. 3.15: The state transition diagram of 'Grasp object' skill.

3.7 Grasp planing under task constraints

In the real-world, grasping is normally associated with a purpose. This purpose includes object transportation, using an object as a tool, assemble an object, hand-over an object, etc. The purpose defines how the object must be grasped to perform a given task. After successfully grasping an object, the gripper blocks a certain face of the object. It must be guaranteed that the blocked face of the object is not task-relevant. We use a set of planes to represent these task-relevant faces to model the task constraints. These planes restrict the approach direction of the gripper and encode task-relevant information.

In the low-level skill ‘Grasp Object,’ the robot need to find a suitable grasp configuration to execute a pickup task while without colliding with the environment. The selected grasp configuration has to fulfill the conditions which the object can be placed or inserted in the correct direction since the object model is known before the task. One can plan a set of possible stable grasp configurations for the object model using, e.g. a model based grasp planner [132]. The problem can then be defined as follows. Given a set of stable grasp configurations, find the best setup which is suitable for the task.

One naive approach to this problem is to check each possible grasp configuration using a brute force strategy. For each firm grasp in the set, the inverse kinematics (IK) of the robot will be calculated both for the pickup and place task. The grasp configuration will be selected if IK solutions exist in both cases. However, for the most real-world object, the number of available stable grasps is infinite. Because computing collision free IK is an expensive operation, this approach does not scale well with the number of firm grasps. To reduce the number of computing inverse kinematics, one needs to use the implicit task information which is already available. For a typical pick and place task, the orientation of the object before picking and after placing are given. This information can be regarded as a set of task constraint and is valuable for reducing the search space.

We propose to solve the problem in the following steps. First, we compute for each grasp the probability of success given a set of task constraints. Then we sorted the grasp configuration with respect to the probability value. At last, we evaluate the IK solution from the grasp configuration with the highest probability and return the first one which has IK solutions both for the pickup and for the place action.

3.7.1 Grasp and task parameterization

A grasp can be parameterized in different ways. One common parametrization is to use 6-D to represent the spatial relation of the gripper to the object and a vector to represent the joint position for holding the object. However, this parameterization cannot generalize across different grippers. Depending on the shape and DOF, the grasp configuration for one gripper usually differs from another one despite grasping on the same points. To avoid this restriction, we focus on the antipodal grasp and assume that complex grippers can realize an antipodal grasp like a parallel gripper. Therefore, we can now use a tuple $(p_g,$

n_{ap}, n_{ef}) to represent a grasp, where p defines the 3-D position of the gripper, n_{ap} defines the approach direction of the gripper, and n_{ef} gives the line of establishing the force.

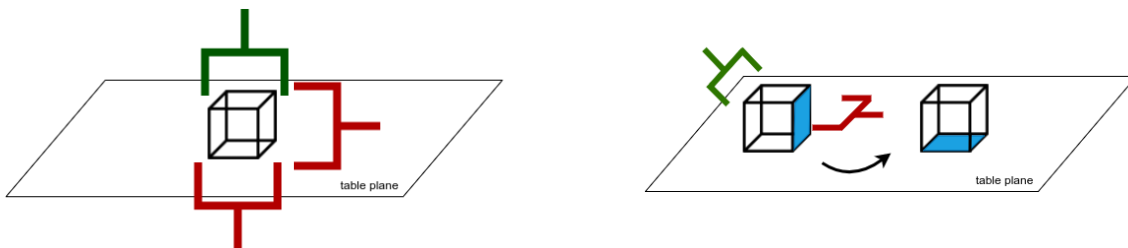
In order to perform a task on an object, the robot should choose a grasping orientation, in which the target surfaces are not covered by the gripper. Therefore the target surfaces can be specified as a set of planes. The planes represent the task constraints of grasping. We further categorize the planes into two types. One type represents the contact face prior to grasping. The other type represents the surface relevant to the task after grasping. The task constraints are defined as

$$T = \{n_p^1, n_p^2 \cdots n_p^i\}, \quad (3.6)$$

where n_p^i denotes the normal of i -th planes associated to the object and task.

3.7.2 Probabilistic model of satisfying task constraints

In this section, we derive the probabilistic model of grasp success with the help of the following example. Fig. 3.16 gives the case of two everyday grasp situations. The robot must grasp the object placed on the table. In the first grasp situation, no task is explicitly specified. The only constraint is the table plane. As shown in the Fig. 3.16a, the grasp configurations shown in red are infeasible. The one is to grasp the object from the bottom while the other is to grasp from the right side. In both cases, the grasping cannot be performed because the gripper will collide with the table. One possible grasp configuration is shown in green. In this configuration, the robot approaches the object from the top, and the grasping force is established on the side. The second grasp situation (Fig. 3.16b) is to grasp the object and place it on the table in the desired orientation. The face in blue shows the desired contact face after pick and place. We also provide one bad grasp configurations (red), which can not be used for executing the place operation. By contrast, the grasp configuration in green can be used both by pick-up action and place action.



(a) Case 1: no task is specified for grasping. The only constraint is the table plane. (b) Case 2: the object must be placed in the desired orientation. The blue surface specifies a task constraint which must be considered during grasp planning.

Fig. 3.16: Two common grasp situations

From the observation above, the approaching and the closing direction of the gripper are essential to the task-oriented grasping. The smaller the angle between the approaching

direction and the normal of the table, the more unlikely the gripper will be in a collision. The larger the angle between the closing direction and the normal of the table, the gripper is more unlikely in a collision. By considering the mentioned two factors, we propose the following mixture model to calculate the probability of task satisfaction $P(C = 1|u, t)$.

$$P(C = 1|u, t) = \frac{1}{n_p} \sum_{i=1}^{n_p} (\omega_1 \cdot P_1 + \omega_2 \cdot P_2), \quad (3.7)$$

where n_p denotes the number of planes to be considered. P_1 and P_2 denote the probability of success by considering the approaching and closing direction individually. ω_1 and ω_2 are the weighting factors which satisfies the following equation,

$$\omega_1 + \omega_2 = 1 \quad (3.8)$$

The value P_1 is further given by

$$P_1 = \frac{4}{\pi^2} \cdot \theta_{cp}^2, \quad \theta_{cp} \in (0, \frac{\pi}{2}). \quad (3.9)$$

θ_{cp} is the angle between the normal of the plane n_p^i and the gripper closing direction n_{ef} . P_2 is given by

$$P_1 = \frac{1}{\pi^2} \cdot (\pi^2 - \theta_{ap}^2), \quad \theta_{ap} \in (0, \pi), \quad (3.10)$$

where θ_{ap} is the angle between the normal of the plane n_p^i and the approaching vector n_{ap} .

3.7.3 Model verification

In this section, we use the aforementioned scenario, grasping the cube from the table, to verify the proposed model. The verification is conducted in the following steps. First, we generate a set of grasps which already fulfilled the force closure condition. Then, we specify the task constraints which are given by a set of planes. Finally, we evaluate the grasps with the proposed model.

Because cubes have parallel planes, we can simply generate valid grasp configurations by rotating an initial parallel grasp around the major axis of the cube. This can be done for all three major axes of the cube. The generated grasps are visualized in Fig. 3.17. The grasp configuration is visualized by a red arrow showing the approaching direction and a green line representing the closing direction.

The proposed model is verified by four grasping scenarios which have different configurations of task constraint. Fig. 3.18a depicts the first grasp scenario, where only one plane is considered. The probability of satisfying task constraints is computed and shown at each individual grasps. The best grasp is marked with a blue circle. In Fig. 3.19a, the probability of satisfying task constraints for each grasp is plotted against the rotation angle around major axes. The grasp receives the highest probability value by rotating 1.57 rad about the

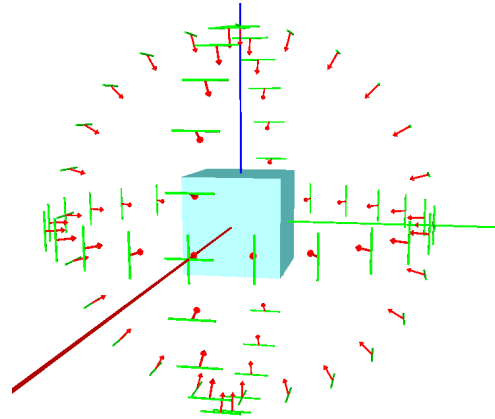


Fig. 3.17: Grasps generated for verifying the proposed model

x-axis and by rotating 0 rad about the y-axis. The two maxima represent the grasps which approach the cube from the top. In the second case, two planes which are perpendicular to each other, are configured as task constraints. In this case, the grasps approaching the object around 45 degrees receive the highest probability values. Fig. 3.18c depicts the distribution of probability if two parallel planes are specified as task constraints. The best grasps calculated by the proposed model are the two grasps which approaching the cube either from the left or from the right. In Fig. 3.18d, an additional plane is added based on the previous setting. The best configuration, in this case, is approaching to grasp from the top because other directions are much more probable to cause a collision than the top grasp. The complete probability of satisfying task constraints for each grasp around each axes are given by Fig. 3.19.

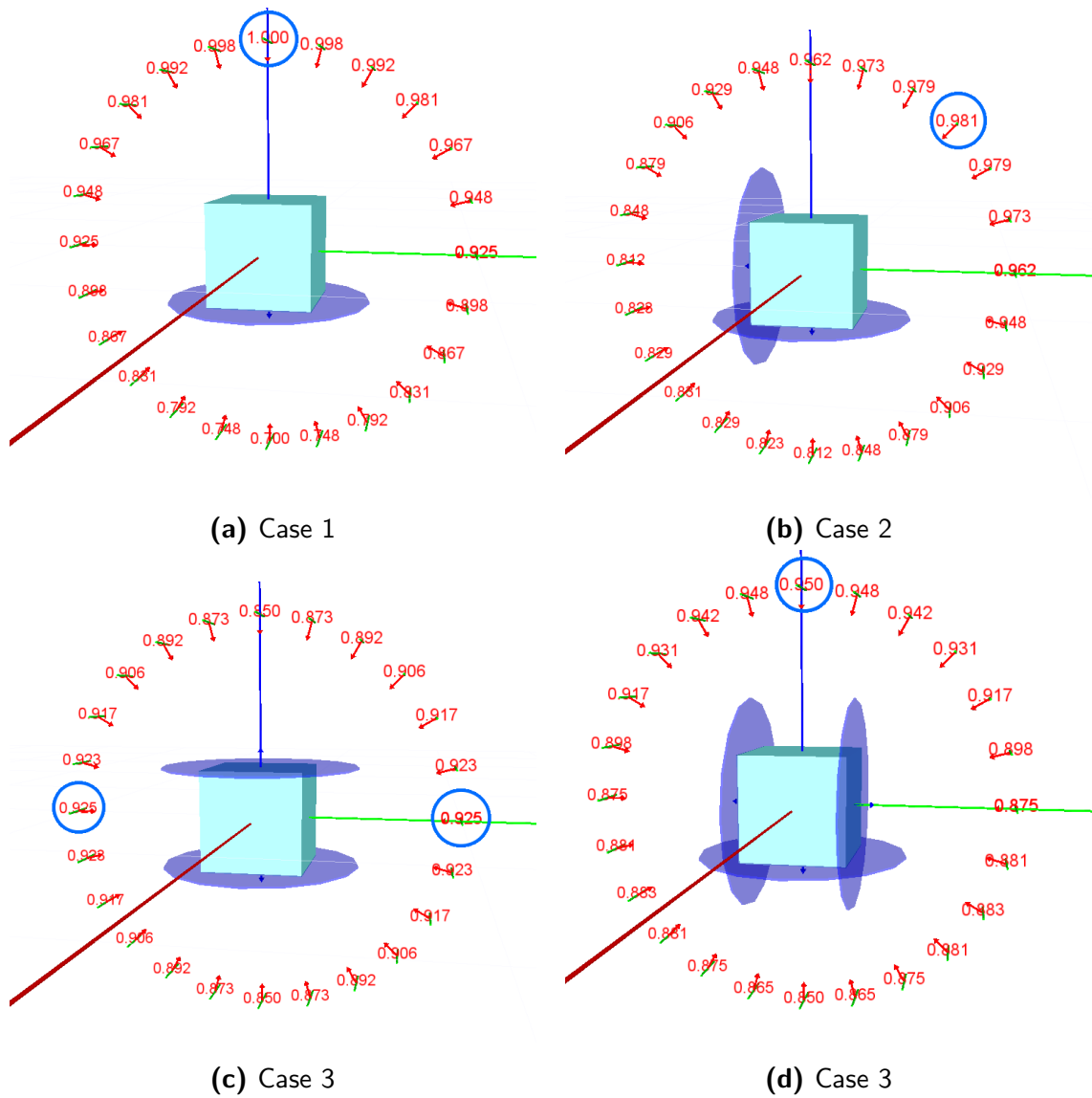
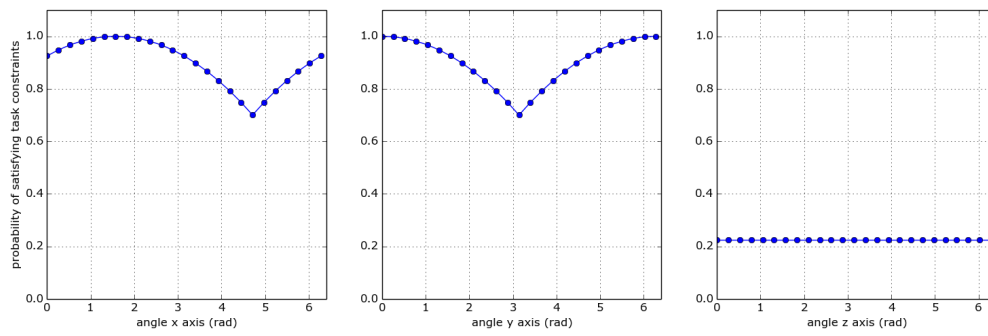
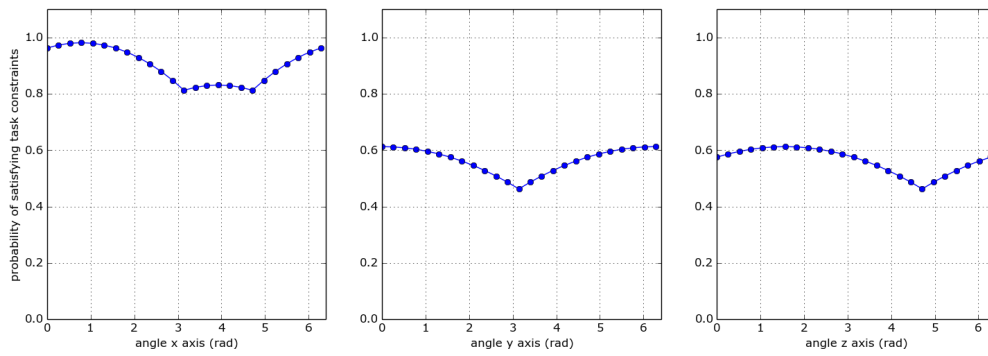


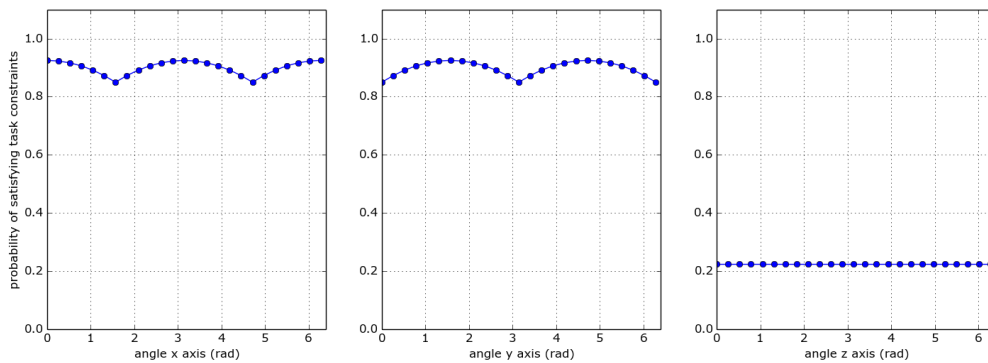
Fig. 3.18: Grasping cube problem with different task contexts.



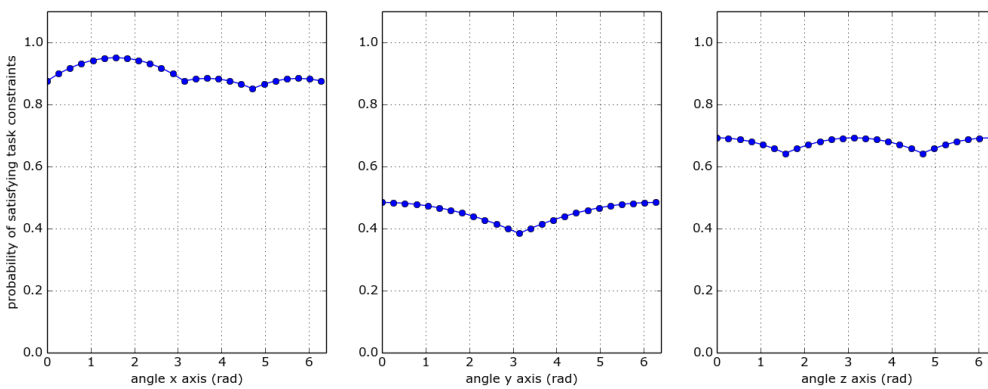
(a) Case 1



(b) Case 2



(c) Case 3



(d) Case 4

Fig. 3.19: Probability of satisfying task constraints

3.7.4 Grasp realization

In order to realize the antipodal grasp on the real robot. Three grasp configurations must be defined based on the grasp planning result. They are the pre-touch, the grasp and the release configuration. A grasp configuration is defined by a tuple $\{g_p, \underline{\lambda}, g_t\}$, where g_p denotes a 6-D pose that represents the relative transformation between the gripper and the object. $\underline{\lambda}$ is a vector which controls gripper postures. g_t defines which grasp type is used. For the gripper we use in the experiment, we can either use a 3-finger cylindrical grasp or a 2-finger parallel grasp to pick up the object. Fig. 3.20 illustrates some examples of grasps realized for the assembly part.

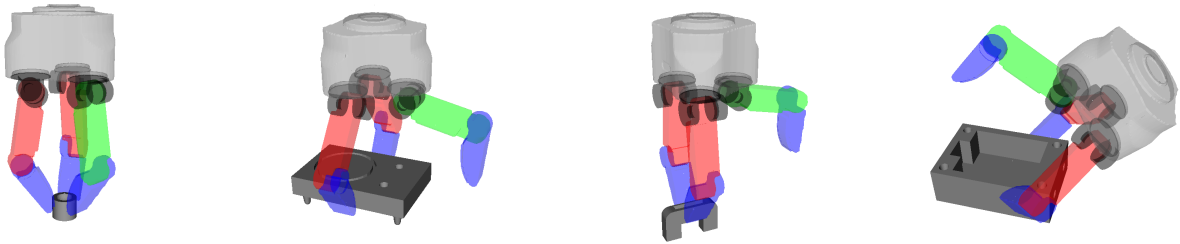


Fig. 3.20: Grasp realization on the SCHUNK-SDH gripper.

3.8 Sensing and actuation coordination

A robotic arm mounted at a fixed location has a limited working space. The working space can be extended by mounting the arm on a mobile platform. Such construction makes more flexible for robots to perform manipulation tasks. However, flexibility also brings challenges in mobile manipulation. In this section, we elaborate two challenges associated with a mobile manipulator in detail. The first one is hand-eye coordination: where to place the mobile platform optimally so that the working space are both reachable by the gripper, and also visible to the sensor of the robot. The second one is arm-base coordination: how to exploit the motion of mobile platform to perform tasks in narrow working space. To face the challenges, we propose to create a model which combines DOF of a mobile platform and an arm systematically.

An omnidirectional mobile platform has typically three degrees of freedom (DoF). We model the three DoF as three virtual joints. Two of them are translational joints and the other one is a rotational joint. The three virtual joints are then organized in a serial kinematic chain structure. The sequence of the joint in the structure can be defined arbitrary. We determine the order of virtual joints in the following sequence. The translational joint in the x direction is defined as the first joint beginning from the root of the kinematic chain, followed by the translational joint in the y direction. The rotational joint is defined as the third joint from the root of the chain. Fig. 3.21 illustrates the modeling of the virtual joints. Organizing the virtual joints in this order has an advantage that the positions of

the virtual joints are equal to the odometry measurement. Let $\underline{q}_m = [q_x, q_y, q_\theta]^T$ be the virtual joints, $x_{\text{odom}}, y_{\text{odom}}, \theta_{\text{odom}}$ be the odometry measurement provided by the mobile platform. The positions of the joints are given by,

$$\underline{q}_m = \begin{pmatrix} q_x \\ q_y \\ q_\theta \end{pmatrix} = \begin{pmatrix} x_{\text{odom}} \\ y_{\text{odom}} \\ \theta_{\text{odom}} \end{pmatrix}, \quad (3.11)$$

One may also define the rotational joint as the first joint from the kinematic root followed by the translational joints. However, such a definition has some drawbacks. First, further calculation are required to obtain the positions of the virtual joints. Second, the position of translational virtual joints is coupled with the rotational joint, which means a small change in orientation of the mobile platform may cause a large change of translational joints. Such a definition may result in instability of controlling the mobile platform.

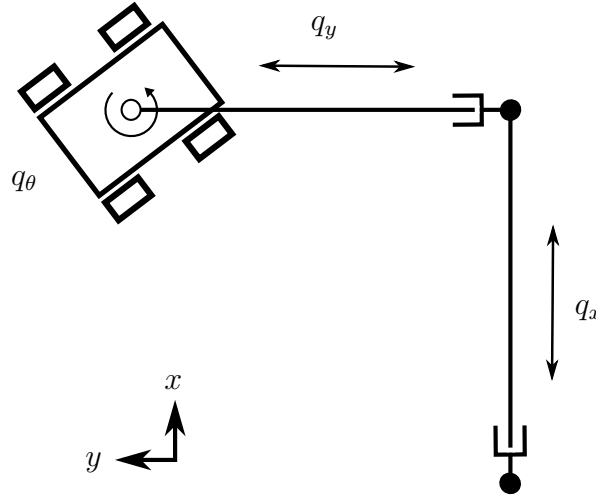


Fig. 3.21: The virtual joints defined for a mobile platform.

To control the real motion of the mobile platform, the desired motion of virtual joints has to be converted to the velocity commands of the platform. Let $\underline{v}_c = [v_x, v_y, v_\omega]^T$ be the translational velocity commands and rotational velocity commands of the mobile platform, the conversion of the desired virtual joint velocity $\underline{\dot{q}}_m$ to the velocity command of the mobile platform is given by

$$\underline{v}_c = \begin{pmatrix} v_x \\ v_y \\ v_\omega \end{pmatrix} = \begin{pmatrix} \cos q_\theta \cdot \dot{q}_x + \sin q_\theta \cdot \dot{q}_y \\ -\sin q_\theta \cdot \dot{q}_x + \cos q_\theta \cdot \dot{q}_y \\ \dot{\theta}_{\text{odom}} \end{pmatrix} = \begin{pmatrix} R_\theta^T & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \underline{\dot{q}}_m, \quad (3.12)$$

where R_θ is a rotation matrix around the z axis. The desired virtual joint velocity $\underline{\dot{q}}_m$ is usually the output of a motion generation module.

So far we have defined the virtual kinematic chain for the mobile platform. To enable arm platform coordinated motion, we define another virtual kinematic structure which

joins the kinematic chain of the arm to that of the platform. The configuration space (c-space) of the arm is extended by additional three DoF of the platform. The creation of the virtual kinematic structures opens a new dimension for mobile manipulation tasks in the following three domains.

Combined inverse kinematic Combined inverse kinematic tackles the first challenge in mobile manipulation, i.e. where to place the robot optimally. The problem of solving combined inverse kinematic is formulated as follows. Given a 6-D pose relative to a fixed coordinate system, what are the joint configurations should a mobile manipulator take to reach the pose by its end-effector. The solution to this problem include the positions of the arm as well as the positions of the virtual joints of the platform. The positions of the defined virtual joints indicate the placement of the robot. As we will show later, solving combined inverse kinematic is a quite frequent action in assembly tasks.

Integrated motion planning Integrated motion planning tackles the second aforementioned challenge. The new defined kinematic structures can be seamlessly integrated into a motion planner. Each kinematic chain defines its own subset of the joint space in which motions are generated. A motion planner can plan platform-only motion, arm-only motion or arm-platform-combined motion according to a planning request. For the robots working in narrow spaces, planning arm-platform-combined motion allows the robot to exploit the mobile platform to reach the narrow space.

Switchable task space control Task space control controls the end-effector to follow a defined trajectory in Cartesian space. Similar to the motion planning, the proposed concept can be used to switch between arm-only or combined motion according to the requirement. For example, task space control based on arm alone can be applied in the grasp motion phase, in which the motion uncertainty of the platform does not have influence. Task space control based on combined-motion can be applied in the pre-grasp phase to execute pre-grasp manipulation strategies such as pushing an object into the desired region.

3.9 Experiments

3.9.1 Evaluation of the skill framework

In order to evaluate the proposed approach for assembly tasks quantitatively, we conduct three experiments to study

- whether arm-platform coordination performs better than the traditional approach,
- the influence of initial object pose on the overall performance,
- the influence of different assembly sequence on the overall performance.

In the first experiment we aim to compare the assembly performance between arm-platform coordination and a traditional approach. For arm-platform coordination, the motion of arm and platform is planned jointly, while for the traditional approach, the motion of arm and platform is planned in sequence. We measure the performance of an assembly task by accumulating the duration of all the trajectories. We use the same initial configuration of the robot and assembly parts. We also keep the same parametrization of the assembly sequence in the experiment given by Fig. 3.23. The assembly task is repeated 10 times for both methods to obtain a statistic result. In Fig. 3.22, we use a box plot to illustrate the result. The median of total time is 85 s for arm-platform coordination and 108 s for the traditional approach. The performance of using arm-platform coordination is improved by 21 %.

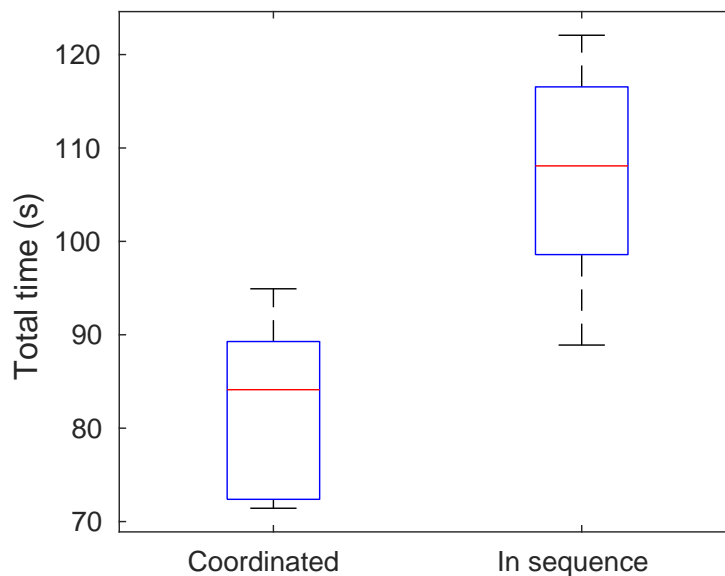


Fig. 3.22: The total time used in an assembly task by two methods. Planning arm-platform coordinated motion reduces the total time of assembly by 21%.

In the second experiment we study whether the initial placement of objects may influence the assembly performance. For this purpose, we select three test scenarios where the initial object poses are different from each other (see Fig. 3.24). The same parametrization of assembly sequence is used for each test scenario. The assembly task is repeated for 10 times as we did in the first experiment. Besides the total time used for accomplishing the task, we also record the total platform traverse in translation and rotation. Fig. 3.24a depicts the total time used in each initial configurations respectively. Only several seconds deviation can be observed from the median of each box plots, which indicates the average time used in an assembly task does not depend on the initial configurations. The width of each box also only deviates from several seconds, which infers the variation of the total time is also tiny. Therefore, we can conclude that the initial configurations of the assembly parts only has minimal influence on the total time of an assembly task. Fig. 3.24b and 3.24c show the total traverse of the robot in translation and rotation respectively. The robot

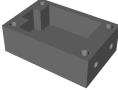

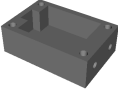
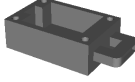
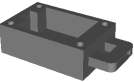

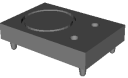
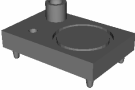

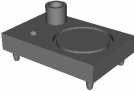
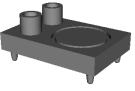
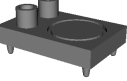
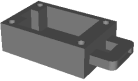
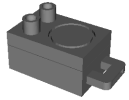
Step	Skill	Parameters		Subproduct
1.	P.R		Back	
2.	P.I			
3.	P.R		Bottom	
4.	P.I			
5.	P.I			
6.	P.I			

Fig. 3.23: Assembly sequence parameterization for experiment I., II. & III.1

traverses about 1 meter more, while about 1 radian less, in initial configuration 1 than in configuration 3. In configuration 2, there is an outlier that indicates the robot traverses more than 7 meters in total to finish the assembly task. Further outliers can also be observed in total rotational traverse both in configuration 1 and configuration 3. These outliers reflect the nature of sampling-based motion planner such as the RRT-Connect that we use in the experiment. The planner may encounter difficulty in high dimensional state space and produces sub-optimal longer trajectories. By comparing the plots, there is no evidence to indicate a clear correlation between the platform traverse distance and the total time the robot uses to finish the assembly task. The reason lies in that the length of the trajectory not only depends on the platform traverse distance but also on the arm traverse distance. The joint of the arm may require frequently accelerate and slow down to avoid the obstacles in the configuration space. In these cases, it may happen that although the traverse distance is short, the time of arm trajectory is still long. In summary, the result of this experiment shows under the condition that the parametrization of the assembly sequences are equal as well as the objects are placed within a limited region, the overall assembly performance does not depend on the initial configurations of the assembly parts.

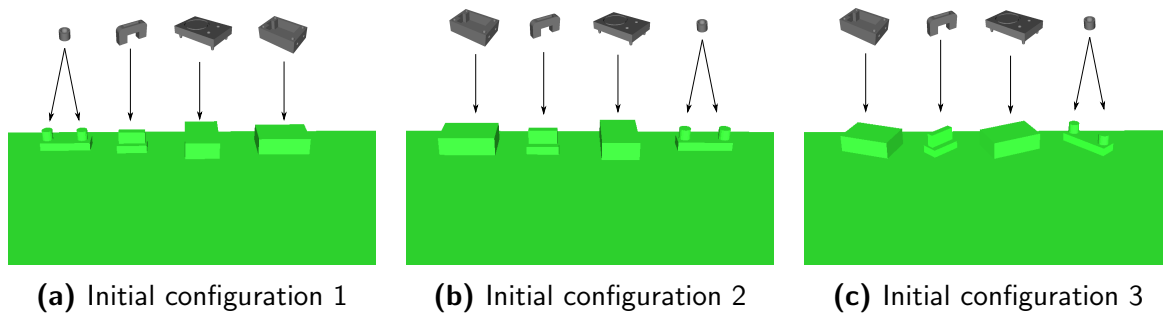
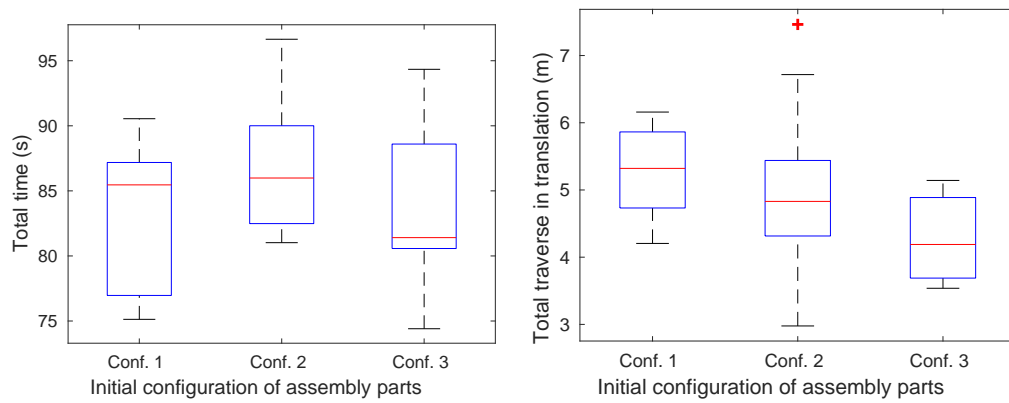
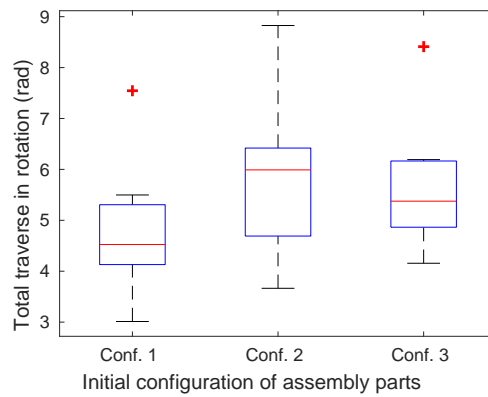


Fig. 3.24: Performance evaluation in terms of different initial configurations

In the third experiment we aim to study the assembly performance with respect to different parameterization of an assembly sequence. For this purpose, we choose three different assembly sequences for the experiment. The sequence parameterizations are depicted in Fig. 3.26. The total time and the platform traverse of an assembly task are evaluated. The same configuration depicted in Fig. 3.24a is chosen for comparing individual assembly sequences. As same as in experiment II, we repeat the experiments for each assembly parameterization for ten times and present the results using box plots. Among the three sequence parameterizations, only five steps are required to assemble the product using sequence 2, while six steps are needed using other two sequences. Therefore, the total time and the platform traverse is expected to be less using the sequence 2 than using the other two sequences. The result depicted in Fig. 3.26 verifies the expectation. Both sequence 1 and sequence 3 have six assembly steps. The median time using sequence 3 is less than using sequence 1, while the traverse distance in the translation of sequence 3 is more than sequence 1. This result again indicates that the time and the traverse dis-






























(a) Total time used to assemble the product (b) Total platform traverse in translation



(c) Total platform traverse in rotation















Fig. 3.25: Assembly performance comparison in terms of different initial configurations

tance are not positive correlated as one may expect. Comparing the width of each box plot, the deviation of the measurements is nearly equal, which indicates the sequence have little influence on the variation of the assembly task. In summary, this experiment shows that there is no large difference of choosing which assembly sequence to use as long as the number of steps is same. This result has some practical advantages. In reality, one may prefer one assembly sequence than another when considering other possible reasons. As soon as the number of assembly steps is not increased dramatically, the total time used to conduct an assembly task does not increase a lot.

Step	Skill	Parameters	Subproduct	Step	Skill	Parameters	Subproduct
1.	P.R	 Back		1.	P.I	 	
2.	P.I	 		2.	P.I	 	
3.	P.R	 Bottom		3.	P.I	 	
4.	P.I	 		4.	P.R	 Back	
5.	P.I	 		5.	P.I	 	
6.	P.I	 					

(a) Sequence 1

(b) Sequence 2

Step	Skill	Parameters	Subproduct
1.	P.R	 Back	
2.	P.I	 	
3.	P.R	 Bottom	
4.	P.I	 	
5.	P.I	 	
6.	P.I	 	

(c) Sequence 3

Fig. 3.26: Performance evaluation in terms of different assembly sequence parameterization.

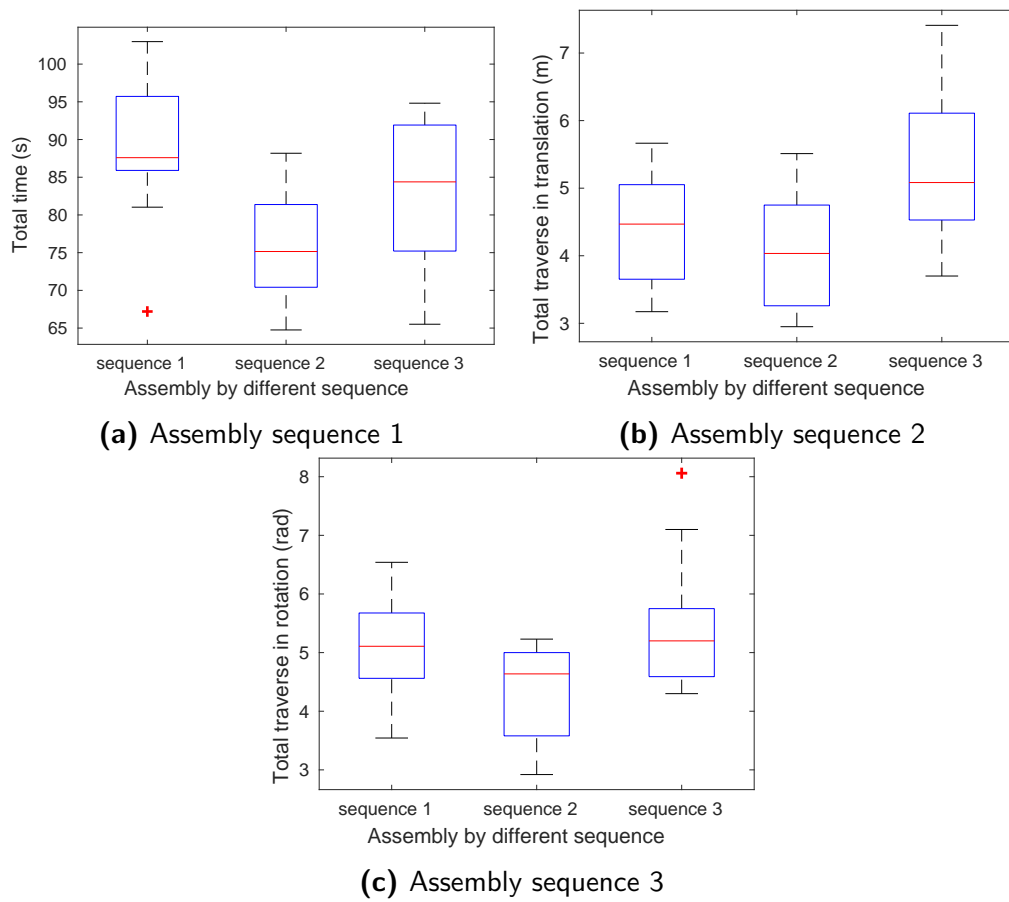


Fig. 3.27: Evaluation in terms of different assembly sequence

Validation on a real robot We validate the proposed skill framework on our real robot. Fig. 3.28 depicts a video sequence of our robot performing the assembly tasks. The assembly sequence is identical to the one shown in Fig. 3.23. The initial place orientation of the assembly parts are similar to Fig. 3.26a. Fig. 3.28a to Fig. 3.28c show the execution of the first assembly step which uses the P.R skill. Fig. 3.28d to Fig. 3.28g show the second assembly step, in which the robot pick the red part and insert to the box. From Fig. 3.28h to Fig. 3.28i, the robot reconfigures the assembled sub-product. Then the robot use the P.I skill to insert both ‘button regulators’ to the top part of the ‘radio’ shown in Fig. 3.28j to Fig. 3.28n. In the last step, the robot insert the assembled sub-product together with the P.I skill.

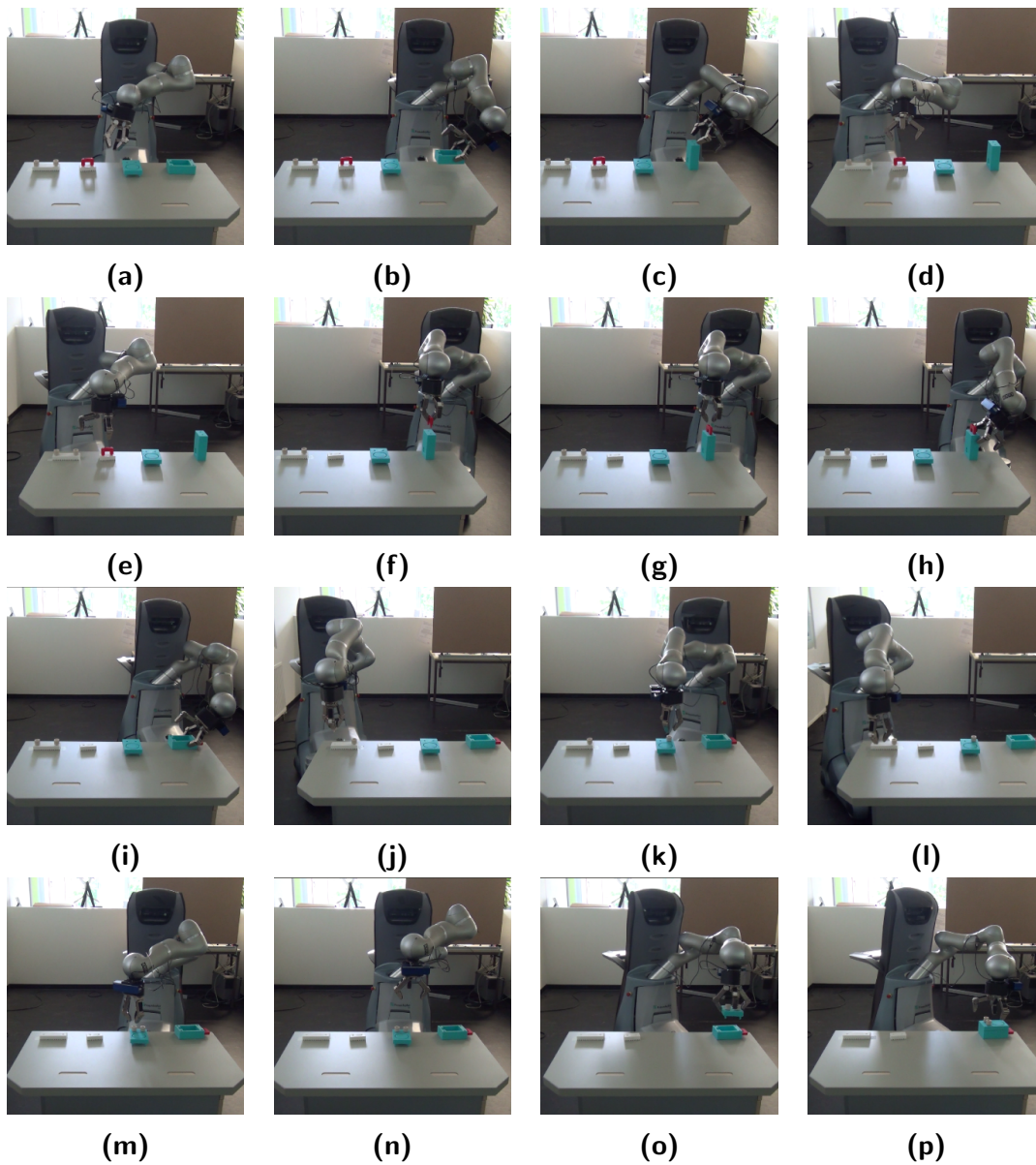


Fig. 3.28: Execution of the assembly task on a real robot. The parametrization of the assembly sequence corresponds to the one shown in Fig. 3.23.

3.9.2 Reachability analysis

Analysis of the reachability of the whole mobile manipulator can help to reason about when and how to use the mobile platform to perform manipulation tasks. In this section, we exploit a tool, capability map, originally proposed in ([135],[134]) to analyze the reachability of a mobile manipulator. The capability map represents the reachability of a robotic arm in a given working space. The generation of a capability map requires three basic components: kinematic description of the robot, an inverse kinematic (IK) solver of the robot and a simulation environment for collision checking. The algorithm for generating the map include three steps. In the first step, a set of poses is calculated systematically in the working space. To cover the entire working range of an arm, the workspace is first discretized into a voxel grid with each voxel having the same size. In each voxel, an inscribed sphere is defined. Then, the poses are generated evenly on the surface of the inscribed sphere. Fig. 3.29 depicts the inscribed spheres which we create for a robot. The length of each voxel is 10 cm. Fig.3.30 shows the poses generated on the surface of each inscribed sphere with different density parameters. In the second step, the IK solver is used to compute a solution and check the validity of the solution for all the poses. The number of the poses which have an inverse kinematic solution is recorded. This is the most time consuming step of the entire generation process. In the last step, a reachability index is computed for each voxel by

$$\text{Reachability index} = \frac{\text{Number of poses pass IK}}{\text{Total number of poses}} \quad (3.13)$$

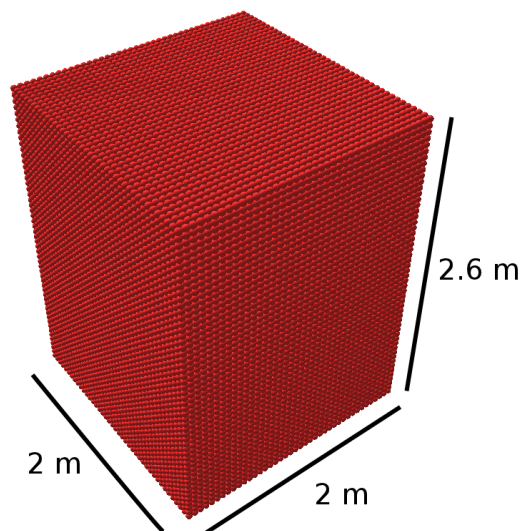


Fig. 3.29: A 2 m by 2 m by 2.6 m working space is discretized to voxels with the length of each voxel equaling 10 cm.

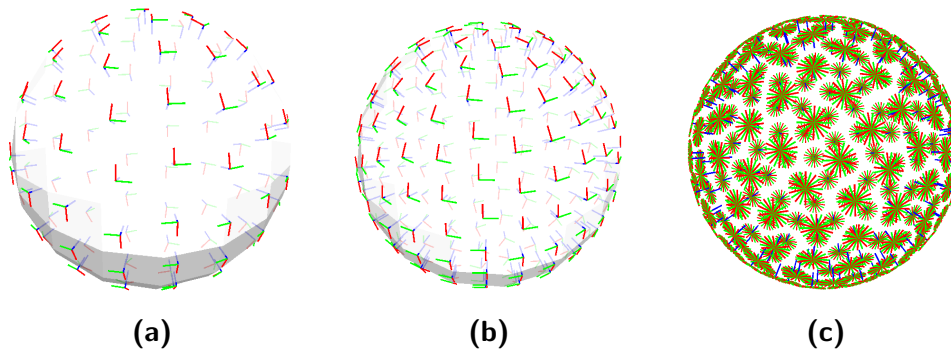


Fig. 3.30: Poses generated in different density on the surface of an inscribed sphere. From left to right: sparse to dense

The algorithm that we use for generating the capability map is implemented in the ‘OPENRAVE’ simulator [22]. The ‘IK-FAST’ inverse kinematic solver is chosen to compute whether a solution exists or not. An extension to the original method is that additional environment models are included in the simulator so that a collision checker provided from the simulator can be used to rule out the IK solutions which contain self-collisions or collisions with the environment. Fig. 3.31 depicts the capability map at the height of 0.75 m. We parametrize 200 poses to be generated on each inscribed sphere. The total time for generating the map is about 30 minutes. The maximal reachability index of the arm is about 85 %. Fig. 3.32a and Fig. 3.32b illustrate a side-view and a top-view of the capability map respectively. The region occupied by the magenta spheres are the optimal place which grasp or manipulation tasks should be conducted because in these locations the arm has the largest reachability index.

3.9.3 Benchmarking manipulation scenarios

In the previous section we introduce the capability map and use this tool to inspect the reachability of a mobile manipulator. In this section, we analyze two manipulation scenarios and benchmarking the success rate of motion planning by different motion planners.

Optimal placing of a mobile manipulator

The first scenario we consider is manipulation on table-tops. In this scenario, a robot grasps, places and manipulates objects above a table-top. We include a box to represent the collision model of a table and put a robot in front of it. Two capability maps are generated by placing the robot at two different distance. Fig. 3.33 depicts the results. The number of magenta spheres above the table in Fig. 3.33a is more than in Fig. 3.33b. The figures indicate that the closer we place the robot to the table, the more region the robot can reach. This phenomenon follows the intuition. However, if we consider the field of view of the sensor which is mounted on the head of the robot simultaneously, placing a robot as close as possible to the table is not the optimal solution. In this case, the object

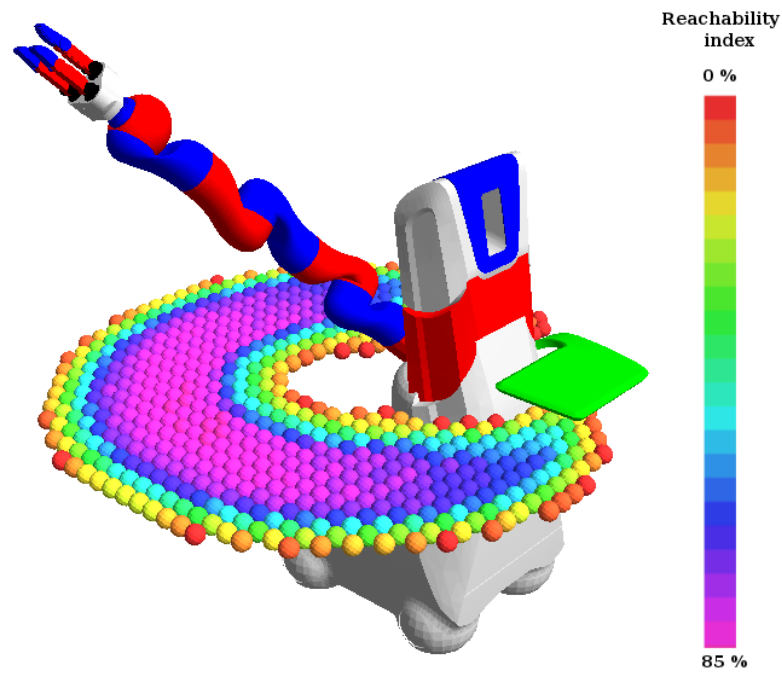


Fig. 3.31: Visualization of a capability map at the height of 0.75 m. The magenta spheres indicate the region where the arm has the largest manipulability.

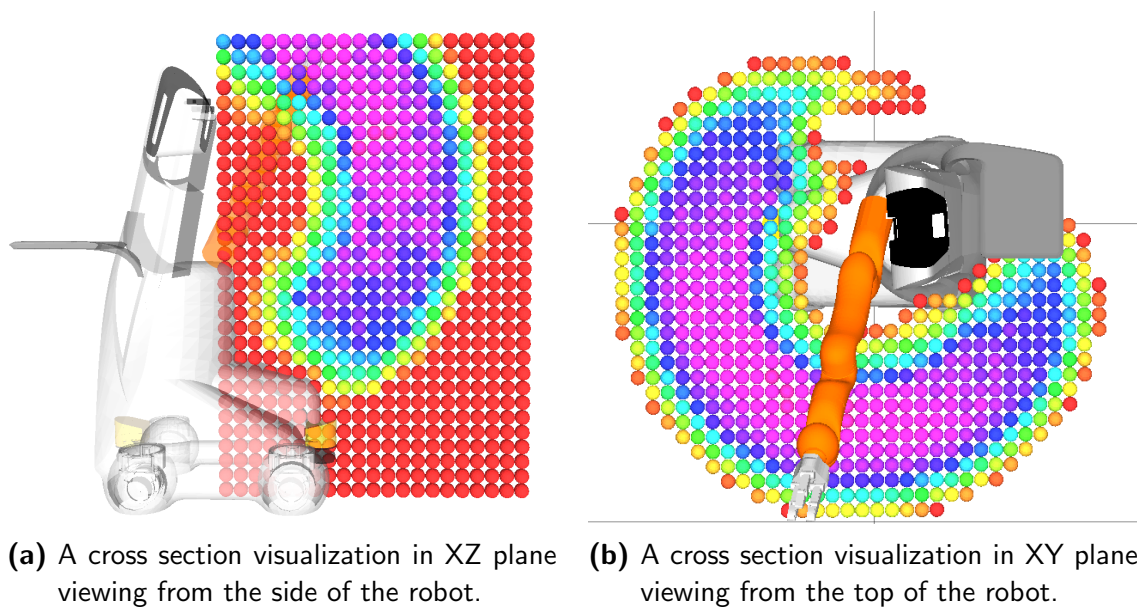


Fig. 3.32: A cross-section visualization of the capability map from two perspectives.

to be grasped (marked in red) may lie outside the field of view of the sensor, so that the robot can not locate the object before the grasping. Another problem exists in the opposite situation as depicted in Fig. 3.33b. The robot is placed farther to the table comparing to the location in Fig. 3.33a. The region above the table can be covered by the complete field of view of the sensor which indicates the robot can locate the object at this distance. However, the arm of the robot can not reach the table as shown from the capability map. Although we demonstrate the problem with a fixed head-mounted camera, the problem also exists for an eye-in-hand configuration (sensor attached to the end effector). The capability map, which combines a given environment model in computation, reveals the first challenge we want to address in the mobile manipulation: where to place the robot so that the grasping and manipulation tasks can be performed.

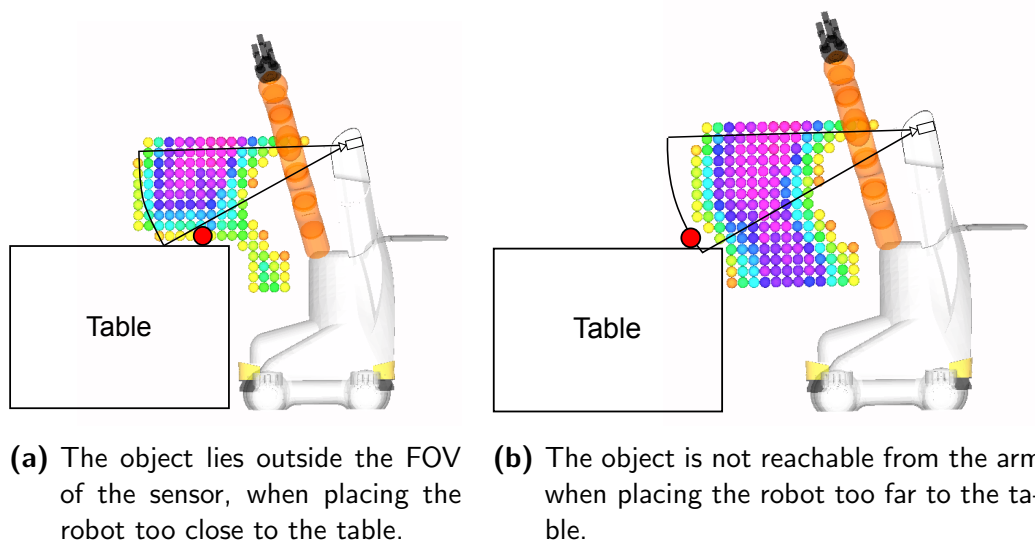


Fig. 3.33: Captability maps generated for two configurations of the robot

Planning motion in limited working space

The second challenge for the mobile manipulation is how to grasp the objects in narrow working space. We select picking from a shelf as an example scenario to demonstrate the challenge, since this task is a typical scenario in human environments. We configure the simulation by placing the robot in front of a shelf model. The capability map is generated for the entire working space inside the shelf model. A cross section visualization of the capability map in this environment is given in Fig. 3.34. The maximal reachability index of this environment is only about 20%. This not only means the number of reachable poses inside the shelf is very low, but also indicates that the valid configuration space, in which the arm does not collide with the shelf, is very limited. Although the arm can reach inside the shelf, it may be difficult for a motion planner to plan a collision-free path to reach the inside of the shelf. To verify this problem, we construct a motion planning scenario in ‘*MoveIt*’ [126] and benchmark the success rate of motion planning in this environment.

We use one start configuration and four goal settings to conduct the benchmarking (see Fig. 3.35). Three motion planners are used for comparison. They are RRT-connect [74], Probabilistic Roadmap Method (PRM) [62] and Expansive Space Trees (EST) [108]. For each configuration, 10 planning queries are conducted for each planner. Table. 3.1 gives the success rate for each configuration/planners. The overall success rate of motion planning in this limited working space is only 26 % in spite of all the configurations are reachable by the arm. The result of the benchmarking explain the utility of the arm alone is not the solution for mobile manipulators to work in a limited environment. The mobility of the robot has to be considered with the arm together to overcome the problem. The challenge remains how to exploit and combine the mobility to allow a robot still be able to work in such narrow environments.

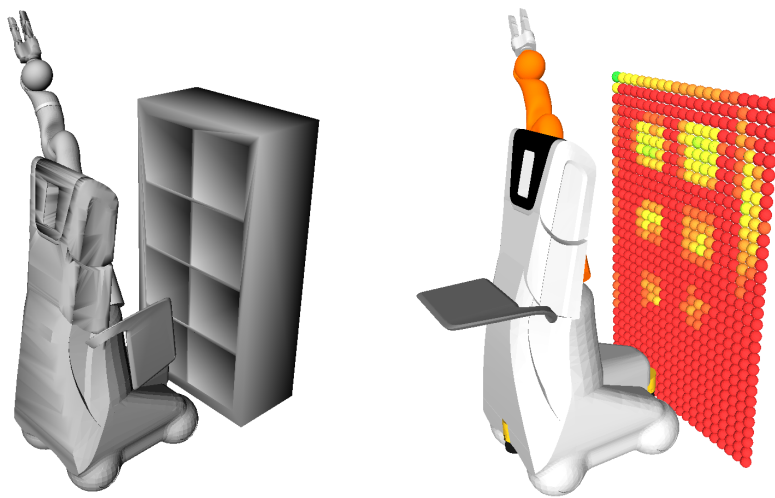


Fig. 3.34: Capability map generated for picking-from-shelf scenario.

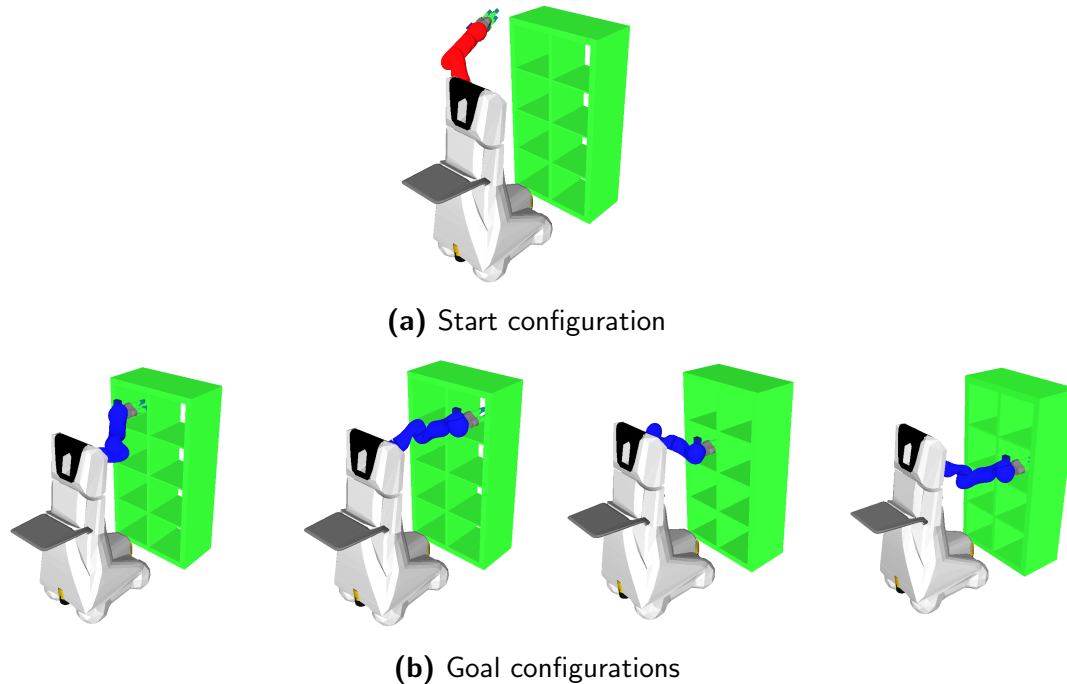


Fig. 3.35: Start and goal configurations for benchmarking the motion planners in a limited working space.

Arm-Base coordination vs. none coordination

In previous section 3.9.3, we discuss the challenge of planning in narrow space for mobile manipulation. A benchmark, grasping from a shelf, is used to demonstrate the difficulty of the challenge. In the previous benchmark, motion planning is only planned for the arm. In this section, we run the same benchmark problem again by choosing the joints of the arm and the platform altogether. Based on the proposed virtual kinematic chain, the planner is able to handle planning 10 DoF simultaneously. Fig. 3.36b illustrates one of the motions planned by RRT-Connect. To reach the inside of the shelf, the platform moves back to leave more free space, while the arm reconfigures in the free space and finally approaches the goal. This kind of coordinated motion helps the robot to move to the goal even in narrow free space. Tab. 3.2 shows the benchmarking result by using the proposed arm-platform coordinated motion. Comparing with the result that we obtain from the previous benchmark only using arm motion, a significant improvement is achieved by all the planners. The average success rate of the planners increases to 83 % comparing to 26 % in the earlier case. The reason for such improvement is the additional three DoF of the platform creates a lot of free configuration space so that it is easier for a planner to generate motion in more free space.

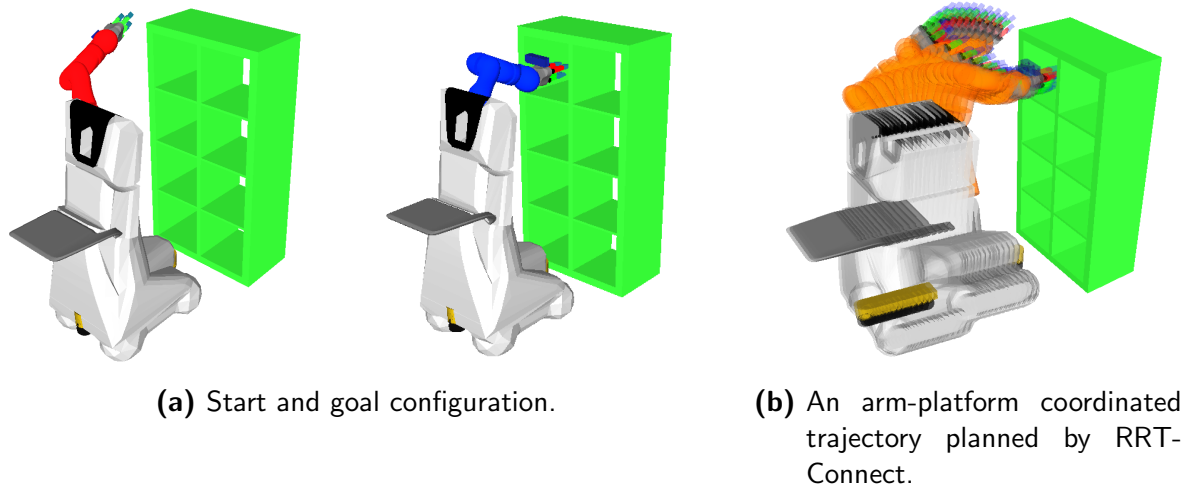


Fig. 3.36: An example planning result using arm-platform coordination

Planner	goal 1	goal 2	goal 3	goal 4	Average
RRT-Connect	40%	100%	40%	10%	47%
PRM*	10%	60%	0%	10%	20%
EST	10%	40%	0%	0%	12%
Overall					26%

Tab. 3.1: Planning success rate of three state-of-the-art motion planners in terms of 4 different goal configurations. 10 planning queries are conducted for each goal configuration to compute the statistic. In summary, 120 planning queries are conducted for the given scenario, only 26% overall success rate is achieved by the planners.

Planner	goal 1	goal 2	goal 3	goal 4	Average
RRT-Connect	100%	100%	90%	80%	92%
PRM*	100%	100%	90%	60%	87%
EST	60%	100%	60%	50%	70%
Overall					83%

Tab. 3.2: Arm-platform combined planning success rate of three state-of-the-art motion planners in terms of 4 different goal configurations. 10 planning queries are conducted for each goal configuration to compute the statistic. Comparing with the 26 % success rate of arm-only motion in Tab. 3.1, arm-platform combined planning achieves 83% overall success rate, which is a significant improvement.

3.10 Summary

In this chapter, we solve the problem of grasp selection of known object under task constraints with sufficient sensing accuracy. The assembly problem is selected as a use case to demonstrate the method. We aim to use a mobile manipulator to tackle the problem because of the flexibility it can offer. For this purpose, we propose a skill-based framework which follows a new design principle. The skills defined in our work not only encapsulate the low-level action primitives such as object localization, control of actuators, but also allows the skills to be executed independently to each other. In this way, the skills itself and the sequence of executing the skills can be parametrized very intuitively.

To apply the framework with mobile manipulators, we have to address two additional challenges. The first one is ‘Optimal placement of the mobile platform’ and the second one is ‘Motion planning in limited spaces.’ We propose to tackle these challenges by modeling the movement of mobile platform using virtual joints. Thus, we can handle the motion of the arm and motion of the platform in a coordinated fashion. In this way, two challenges are handled simultaneously in a unified method. Based on this technique, methods such as whole body combined inverse kinematic, coordinated motion planning and switchable task space control can be built on top of it. These methods are embedded in the skills defined in our framework.

One problem that we also address in our framework is grasping under task constraint. Both high-level skills demonstrated in this chapter require a grasping action. The grasp configuration must satisfy the task intention defined in the skill. We propose a sampling based method to select the most suitable configuration to execute that grasp. The method draws many samples both for the grasp configuration and for the configuration intended for that task. This is done by using the aforementioned whole body inverse kinematic technique. The method considers both the grasp action and the task action (place/insert) to select the most suitable configuration for executing the skill. The grasp configuration calculated by this method not only satisfy task constraints but also reduces the total time of skill execution. Therefore, only by putting the problem of grasping under constraint in a concrete assembly scenario, additional perspective such as task execution efficiency can be considered.

In principle, we demonstrate how the problem of grasping under task constraint are solved in a challenging setting: Assembly using a mobile manipulator. The method we proposed here shows an overall package which can be used for the assembly task which requires flexibility. Although we only show the result with a toy example with our robot, the proposed methods are generic and can be applied to any mobile manipulators. To transfer the grasp selection method for other tasks, we need to parameterize the task constraints of the object for a specific task. To transfer the skill library for other mobile manipulators or a different gripper, low-level skills such as ‘Control gripper’, ‘Plan motion’, ‘Locate Object’ must be configured and adapted. It is also quite straightforward to extend

the capability of the framework by adding additional skills such as ‘screw’, ‘bayonet mount’, etc. for other assembly tasks which require these skills.

4 A Probabilistic Approach to Grasp Synthesis

Humans can grasp a wide variety of objects which differ in the shape, color, mass, etc. They can also well handle unknown objects which are presented to them for the first time. In order to have a general purpose robot which can help people in the unstructured human environment, grasping unfamiliar items is a must-have skill. In this chapter, we consider this aspect of the grasping problem and provide methods on how this problem can be solved.

4.1 Introduction

Robotic grasping is a quite old problem which has been studied over several decades. Although the speed and resource of computers increased dramatically, the grasping capabilities of robots are still far behind these of the human. These capabilities, which are essential for the grasping tasks, include cognitive level, the ability of abstraction and manipulation experience. In the recent year, with the emergence of low-cost sensors, high-speed graphical card, some progress has been made from theoretical grasp quality analysis to real grasping problems. Solving the problem of unknown object grasping becomes one of the frequently addressed problems in the research community because it is one of the necessary steps for robots toward human-level intelligence of manipulation [87].

In order to solve grasping unknown objects, the robot has to face some challenges. One challenge lies in the wide variety of objects existing in the world. Objects are often different in their appearance, shape, geometry, dynamics, etc. To model them individually is not a solution. How a robot can handle this variety is hard. On the other hand, robots also have a limited sensing capability. A measurement of a sensor is never perfect. For example, the sensor data from a RGB-D camera contains not only errors caused by pixel noise but also errors caused by reflective or transparent object surface. To manage the uncertainty of sensing is another challenge.

With the emergence of low-cost RGB-D cameras, methods which address this problem shift from the traditional way of analyzing grasp stability to a data-driven / sensor-guided fashion. RGB-D cameras such as Microsoft Kinect, Intel Real Sense are often used for training object model or for grasps detection, because they provide rich sensor information about both the appearance and the geometry of an object. Previous works which address the problem in this new direction can be categorized into two classes. The methods in the first class typically use a learning based methods ([73],[127],[125]). These methods infer the grasp configuration directly from the sensor data. Supervised learning is used to build

a model for grasp evaluation. A large training set is normally required to build the grasp evaluation model. Good training examples can be labeled either by human intuition, running grasp experiment in a simulator[90] or with real robots[78]. The methods in another class use a shape based approach ([92],[110],[41]), they first infer the shape of objects from the sensor data. Then, the grasp configurations are searched based on the shape estimate. Our method can be categorized into the second class, as we believe the form of an object provides the most clues on whether a grasp configuration may succeed or not.

We propose a probabilistic approach to address the grasp synthesis problem. Our approach contains two building blocks for finding a good grasp configuration. In the first building block, we solve the problem of evaluation of a given grasp. To do this, we need a model to represent the shape of an object and a model of evaluating a grasp. As we have suggested that shape contains the most information for grasping, it is most suitable to represent an object by its shape. For this purpose, we propose an object model called probabilistic signed distance function (p-SDF) to represent unknown object surface. p-SDF models measurement uncertainty explicitly and allows measurement from multiple sensors to be fused in real time. To evaluate a given grasp configuration, we propose four rules to perform the calculation. Each heuristics models one aspect which can influence force closure. The heuristics are combined in a weighted fashion to obtain the probability of force closure .

In the second building block, we address the problem of finding the optimal grasp by searching the configuration space. For this purpose, we propose a generic gripper parametrization method which reduces the grasp configuration space. This is done by first organizing the joints from the same gripper finger into a collective group, and using a single parameter to control the movement of the joints belonging to the same finger. In this way, the grasp configuration space can be searched more efficiently. Finally, a two-stage method based on simulated annealing is proposed to find the optimal grasp configuration. We validate the whole approach using real world challenging objects in our experiment. The result shows that our method can handle a variety of objects in a table-top environment.

4.2 Related work

Up to 2010, most previous works addressed this problem by assessing the grasp stability with grasp quality metrics [33]. However, this approach is not scalable to the variety of real world objects, as most grasp quality metrics assume an existing 3D object model. Later, the emergence of low-cost RGB-D sensors such as Kinect opened a new opportunity to study grasp synthesis in a data-driven fashion. Bohg et al. [7] took a comprehensive survey on data-driven grasp synthesis. These approaches exploit 2-D vision [39], RGB-D [2] or multi-modal information [6] to generate grasps for real world objects. In this section,

we provide a review on methods of grasp synthesis, and focus on those which consider the following two aspects, how to handle the uncertainty, and pros/cons of learning based methods.

4.2.1 Methods considering pose uncertainty

Handling uncertainty is considered as one of the most difficult but highly relevant problems, not only in grasping but also in many other fields of robotics. In ([40],[50],[142]), grasping is modeled within a probabilistic framework to take variance in shape and pose into account. The methods can choose the most robust grasp configuration in spite of uncertainties in the perception and control. Pose and shape uncertainty are two common sources that make grasping difficult. Pose uncertainty can be reduced by exploiting finger tip tactile sensing [19][32][28], planning pre-grasp [27][11] or post-grasp [104] strategies. Some other approaches ([130],[66]) proposed new grasp quality metrics that evaluates a grasp with pose uncertainty. Approaches considering pose uncertainty always assume an available object model and a perceptual system which predicts the pose uncertainty.

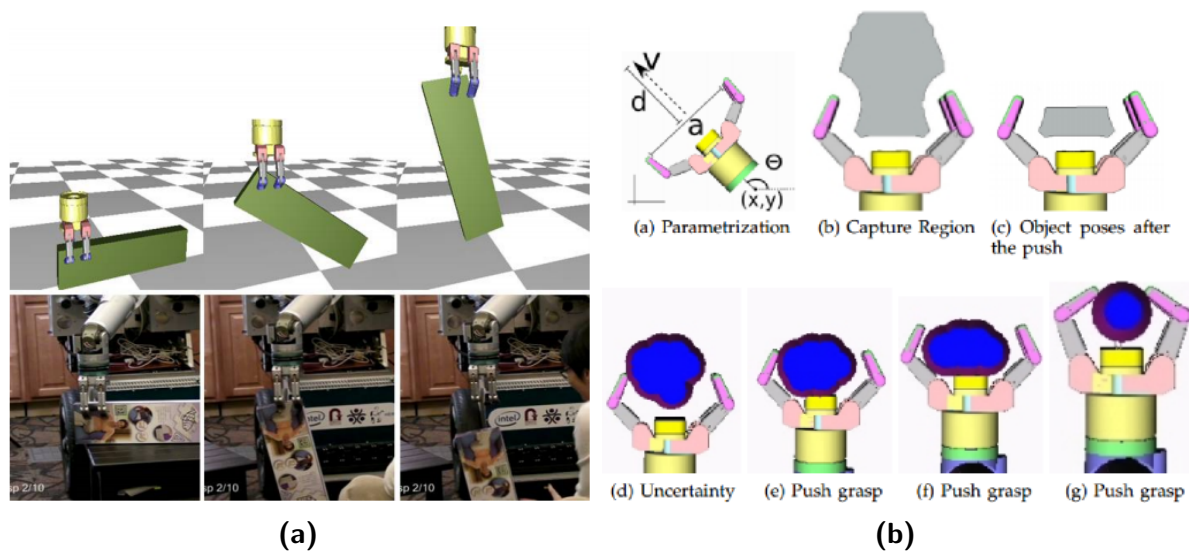


Fig. 4.1: (a) A grasp synthesis method proposed by [66]. They consider the pose uncertainty by simulating object dynamics. (b) A method using pre-grasp manipulation to address the pose uncertainty [27]. They propose a planning framework which allows a robot to interact with objects by a library of actions such as push, slide or sweep operations. The pose uncertainty is addressed by the funneling effect of pushing.

4.2.2 Methods considering shape uncertainty

By contrast, approaches considering shape uncertainty do not require to estimate the pose of an object. Some methods simplify the problem to a two-dimensional planar setting. In [16], Christopoulos et al. proposed how to handle shape and contact location uncertainty of

a 2D object contour segmented from an image. Kehoe et al. [64] proposed an approach to generate zero-slip push grasps based on object shape perturbations. For the methods which explicitly consider shape uncertainty, the choice of object representation is a problem. Gaussian process (GP)[112] is a powerful tool for modelling uncertainty. In [88], [80] and [29], Gaussian process implicit surfaces (GPIS) [131] are used to represent shape uncertainty. Mahler et al. [88] exploited GP in two dimension points, whereas Li et al. [80] used GP to represent 3-D objects. However, there are some remarkable drawbacks in GP-based representation. The first drawback is the heuristic choice of training points. GPIS is obtained by a given set of points which represent the object surface, the space outside the object and the space inside the object. While surface points can be obtained from sensor measurement, points inside and outside the object are chosen heuristically. The shape uncertainty is dependent on this heuristics. The distance between two neighboring points determines the shape uncertainty. The real measurement uncertainty of the sensors can not be considered. The second drawback is related to computational complexity. The complexity for training a GP is $o(N^3)$, where N is the number of points used for estimating the shape. Thus, GPIS does not scale to a large number of points. To find a trade-off between accuracy and computation complexity, these approaches filter out a large proportion of points obtained from sensor data, thus reduce the accuracy of the shape. In this work, we also use an implicit surface to represent objects, however, instead of using GP to estimate shapes, we use our approach described in [24] to determine the object shape. This method is an extension of [99] for modeling shape uncertainty. We use all the measurement points and integrate them into a consistent model so that the resulting object shape is as accurate as possible. Besides, we also model the error characteristics of the sensors so that uncertainties expressed in the model explicitly reflect the characteristics of the used sensors.

4.2.3 Methods using learning based approach

Several learning based methods are proposed to solve the problem. Learning can be conducted either from human demonstration ([46],[30],[118]), from labeled database [77] or from robot's own experience ([109],[78]). There are two classes of methods to address this problem. The first class is a feature-based approach. These approaches try to find specific features such as shape features, multi-modal features from the input data. These features are used to access the feasibility of a grasp. Eppner et al. [31] demonstrated one method in which they tried to match the shape of objects to the grasp type. A similar idea is also described in [54]. Herzog et al. [46] proposed to select grasps based on shape template features. Jiang et al. [59] extracted a rectangular feature to detect grasp configuration. All these approaches require some degree of discretization of the input data, which removes the details of the surface information. For tasks which require precision grasp, these methods are not suited.

Fischinger et al. ([34],[35]) proposed height accumulated features to handle the problem

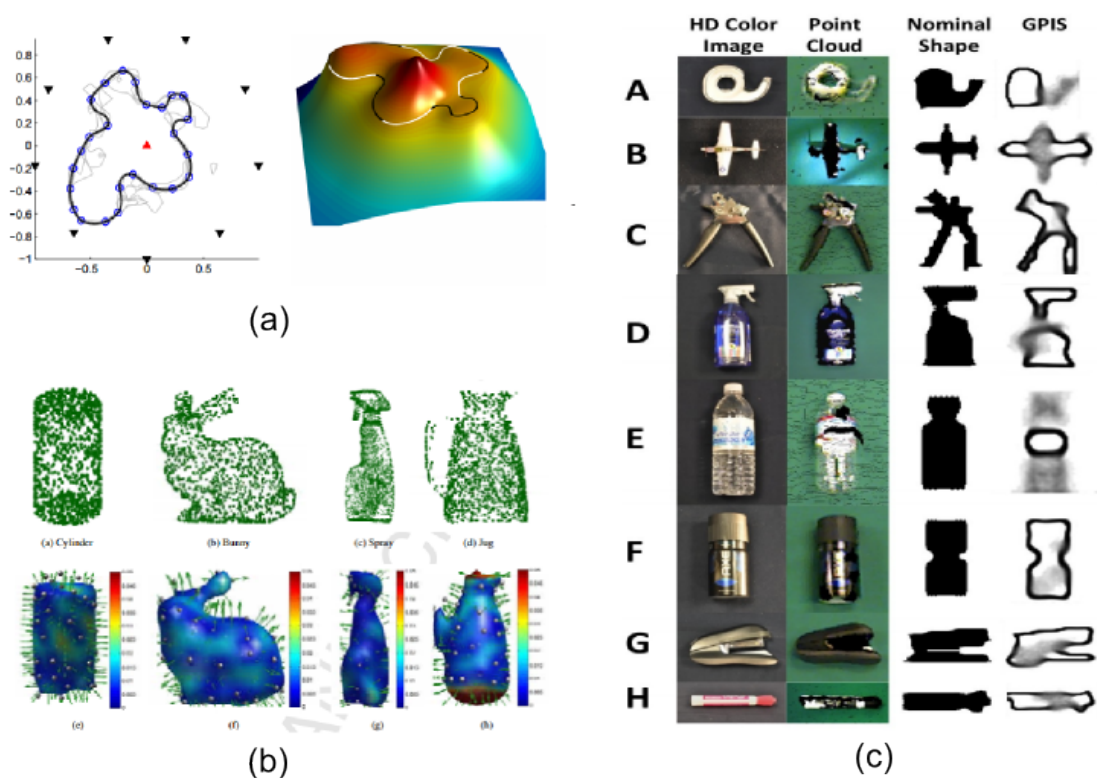


Fig. 4.2: (a) A method of representing object shape by Gaussian process implicit surface (GPIS) [131]. (b) A method for grasp synthesis considering shape uncertainty in 3D. They use GPIS to represent shape uncertainty of objects. The points for training the shape model is obtained from synthetic data [80]. (c) A method for grasp synthesis considering shape uncertainty in 2D [88]. The points for training the shape model is obtained from real sensor.

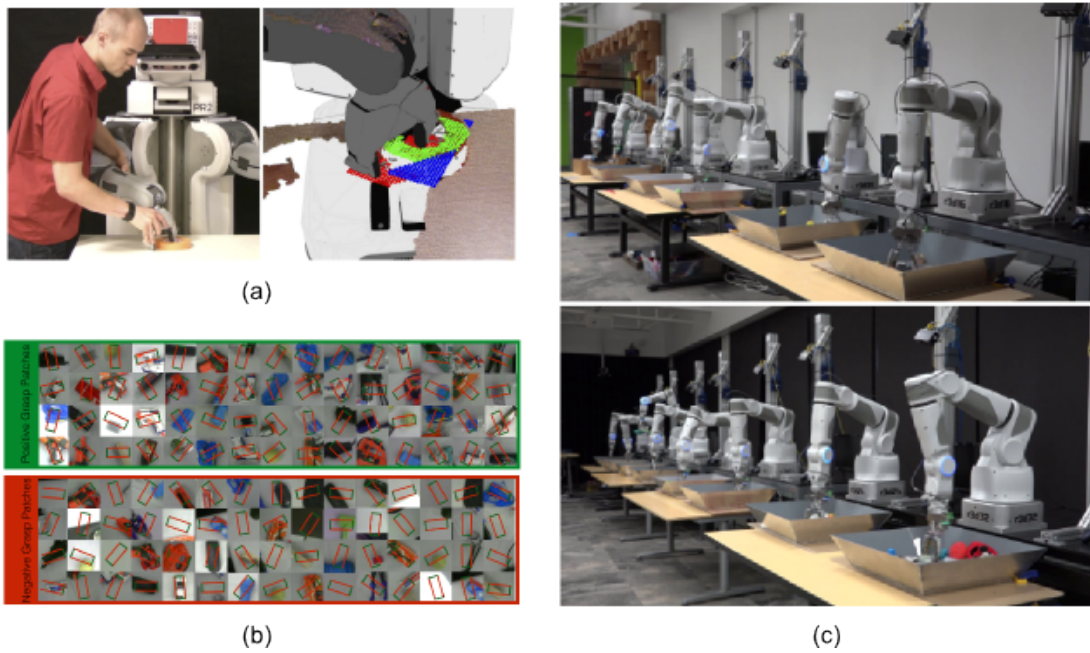


Fig. 4.3: Learning based methods require a training set which include objects and associated grasps examples. Different methods can be used to acquire the training set. (a) Using human demonstration [46] (b) Execute random grasps with a single robot to acquire positive and negative grasp examples. [109] (c) Using a robot farm to acquire the training set. [78]

of grasping unknown objects in clutter in a table-top scenario. The process of finding a grasp is illustrated in Fig. 4.4. An RGB-D sensor first captures the target scene. The obtained point cloud is then discretized into a height grid. A set of features is extracted from the height grid. The feature along with a candidate grasp is then classified using a set of pre-designed features to determine if the candidate grasp is feasible or not. Multiple candidates are tested and the method returns the grasp with the best score for final execution. Later, we use this method as the baseline for comparison.

The second class of approaches exploits machine learning methods to transfer the capability of grasping known objects to unknown objects. It has been noticed recently, the representational power of hand-crafted features is not always sufficient. It might be better to let the system learn them itself. For this purpose, some researchers apply deep learning to solve this problem. Lenz et al. [77] demonstrated how deep learning can find features automatically from multi-modal information. Redmon et al. [114] optimized the performance of grasp detection by a convolutional neural network using the same dataset as Lenz did. In [109], Pinto et al. demonstrated that by gathering the grasping dataset with a real robot over 700 hours, the same approach achieves higher accuracy. Learning can also be exploited to approximate the probability density of contact models. Kopicki et al. [68] proposed this concept to generalize dexterous grasp from known objects to novel objects of the same object categories and transfer to other object categories. An SVM-based learning method is exploited in [107] to find optimal grasps using a simulator.

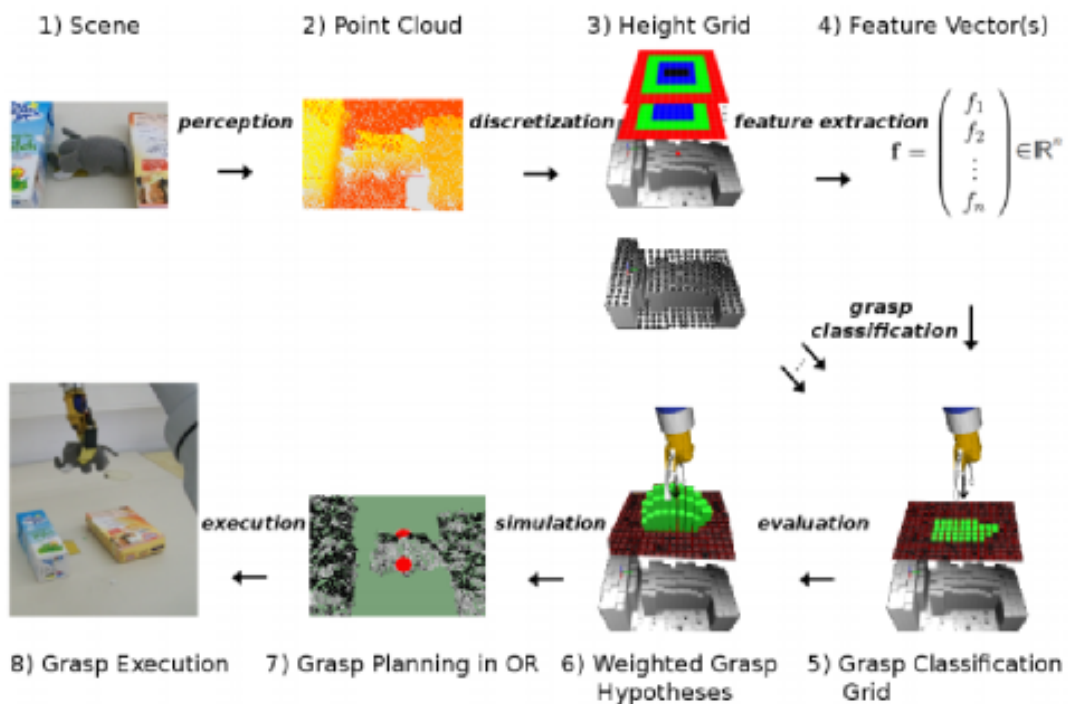


Fig. 4.4: A method using hand-crafted feature for grasp classification proposed by [35]. The method discretizes input point cloud into height map. Features are extracted from height map for grasp classification. The best grasp is calculated by weighted grasp hypotheses and realized in a grasp planning simulator. Later, we use this method as baseline for comparison.

All these works do not consider to fuse the sensor data. Instead of using only single measurement to learn how to grasp, we use sensor fusion in our framework so that the robot can integrate measurements from different sensors and different perspectives into a consistent object representation. In this way, object surfaces are modeled in detail while measurement uncertainty is also represented. Comparing to approaches which require a labeled database for training, we do not need such database or grasp experience in our approach.

4.3 Contributions

We propose a new probabilistic approach to synthesize grasps for unknown objects. Our approach can generate precision grasps with high accuracy. First, a model to calculate grasp success probability is introduced. We separate the modeling of joint grasp success probability into two independent models. The first model describes the physical condition for achieving a stable grasp, while the second model reflects sensing uncertainty, which is a problem that any robots have to handle in reality.

In order to model the shape uncertainty of unknown objects, we propose a new object representation and an associated fusion algorithm called p-SDF. p-SDF is an extension of signed distance functions, which not only models object surface itself, but also indicates the uncertainty of surface caused by sensing. This representation allows surface reconstruction of unknown objects from multiple sensors which may have different noise characteristics. Unlike the previous work, the uncertainty takes the noise characteristics of individual sensors into account. In addition, a fusion algorithm can be configured to run on modern graphic cards so that the reconstruction process can be conducted in real time. In addition, we introduce an approach to calculate the probability of force closure tailed for this surface model. This is done by combining four evaluation criteria in a weighted fashion. Furthermore, a generic parametrization method is demonstrated to reduce the search space of the grasp. A two-stage search algorithm is proposed to find the grasp of the highest success probability.

Our method differs from previous methods, that the uncertainty-aware object representations of those methods do not reflect the true measurement uncertainty of sensors. For the methods which use GPIS, they mostly use a heuristic by defining a set of points inside or outside the objects to construct the model. In addition, the computational time of those methods is too long to run in real time, which significantly limits the performance of a real grasping task. We conduct extensive real grasping experiments to validate the approach, we achieve 93.5 % success rate and outperform the baseline approach which based on machine learning.

4.4 Probabilistic modeling of grasping success

This section gives a mathematic model of grasping success, followed by an illustration of the model on an 1-D grasping example.

4.4.1 Mathematic model

First we define a random variable $S \in \{\text{success}, \text{failure}\}$ as the outcome of a specific grasp motion $\mathcal{G} = \{g_1, \dots, g_n\}$, with g_1, \dots, g_n representing a sequence of gripper configurations and poses. We define $P(S = \text{success}|\mathcal{G}, \mathcal{Z})$ as the marginal probability that the grasp \mathcal{G} succeeds based on the sensor measurement history $\mathcal{Z} = \{z_0, \dots, z_M\}$. It is not straightforward to compute the marginal probability $P(S = \text{success}|\mathcal{G}, \mathcal{Z})$ directly without knowing the state of the object x_o . However, we can compute it by separating it into two probability terms:

$$P(S) := P(S|\mathcal{G}, \mathcal{Z}) = \int_{x_o} P(S|x_o, \mathcal{G}) \cdot p(x_o|\mathcal{Z}) dx_o. \quad (4.1)$$

We call the first term $P(S|x_o, \mathcal{G})$ conditional grasping success model. It indicates how likely a specific grasping action \mathcal{G} will lead to success given the object state x_o . The second term $p(x_o|\mathcal{Z})$ represents the posterior probability of the object given a series of observations, which quantifies the perception uncertainty. In this way, the problem of predicting the marginal grasp success probability is simplified to model the $P(S|x_o, \mathcal{G})$, and to model the perception uncertainty.

After the marginal grasp probability $P(S)$ is computed, the next step is to find an optimal grasping motion \mathcal{G}_{opt} that maximizes the marginal grasp success probability:

$$\mathcal{G}_{\text{opt}} = \arg \max_{\mathcal{G}} P(S). \quad (4.2)$$

4.4.2 Illustration of the concept

To illustrate the proposed mathematic model, we apply this formulation to solve a 1-D grasping problem. We aim to compute the optimal grasp configuration for a simple parallel gripper to grasp a 1-D rigid object with width w . x defines the position of the object. The parallel gripper has a limited opening width constrained by $u_1 \in [0, u_1^{\text{max}}]$. u_2 defines the position of the gripper. Here we assume the grasp motion \mathcal{G} is collision-free with the object and its environment. Thus, the last grasp configuration g_n determines if the final grasp will succeed or not. In this way, the grasp motion can be simplified to the last gripper configuration. For a simple parallel gripper, the configuration is defined by $\{(u_1, u_2)\}$. We call the configuration as a ‘pre-touch’ configuration. Fig. 4.5 depicts this 1-D grasping example.

First let us derive a grasp success probability model $P(S|x, \mathcal{G})$ for this example. Since the positions of the fingertips can determine whether a gripper configuration will succeed,

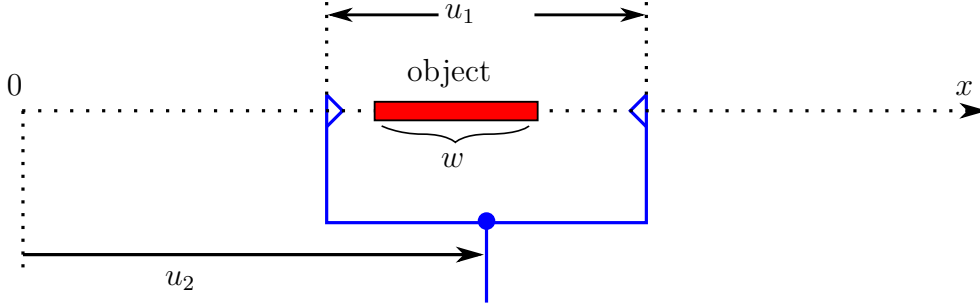


Fig. 4.5: Illustration of a 1-D grasping problem.

we can compute these positions directly from u_1 and u_2 by

$$\begin{pmatrix} c^{\text{left}} \\ c^{\text{right}} \end{pmatrix} = \begin{pmatrix} u_2 - 0.5 \cdot u_1 \\ u_2 + 0.5 \cdot u_1 \end{pmatrix}. \quad (4.3)$$

If the fingertips intersect with the object, the grasp is invalid. In these cases, we assign the probability of grasp success to zero. When the positions of fingertips fulfill the condition ($c^{\text{left}} < x - 0.5w$ and $c^{\text{right}} > x + 0.5w$), the object can be grasped by the gripper. However, for precision grasping, we want to avoid object displacement during grasping. During the phase of force closure, we want the object to move as little as possible. Therefore, a term that penalizes object displacement can be introduced into the grasp success model. We model the grasp success probability by

$$P(S|x, \mathcal{G}) = -\frac{1}{d^{\text{max}}} \cdot d + 1, \quad (4.4)$$

where $d^{\text{max}} = 0.5 \cdot (u_1^{\text{max}} - w)$ denotes the maximum displacement that can be executed by a gripper. $d = |u_2 - x|$ denotes the actual object displacement. By combining this model with a given object posterior $p(x|\mathcal{Z})$, a marginal success probability $P(S)$ can be obtained by evaluating Equ. 5.15. To illustrate how an object posterior can influence marginal success probability, we choose object length $w = 0.2$, the maximal opening width of the gripper $u_1^{\text{max}} = 0.4$. We compare the result by using two different object posteriors to simulate an accurate and a less accurate perception system. We model the $p(x|\mathcal{Z})$ by two Gaussian distributions with $\mathcal{N}(0, 0.3)$ and $\mathcal{N}(0, 0.03)$ respectively. Fig. 4.6 depicts the contour of grasp success probability $P(S)$ as a function of the pre-touch configuration.

4.4.3 Modelling of conditional grasp success probability

For real world objects, it is required to have a conditional grasp success model that considers much more factors than the 1-D example. Two conditions are necessary to ensure a successful grasp. First, the motion of the gripper, which is represented by \mathcal{G} , should not collide with the environment and the object. Second, the last pose and posture g_n which we define as the pre-touch configuration should guarantee that forces from the fingertips are established on the object during force closure. For the first condition, we assume that

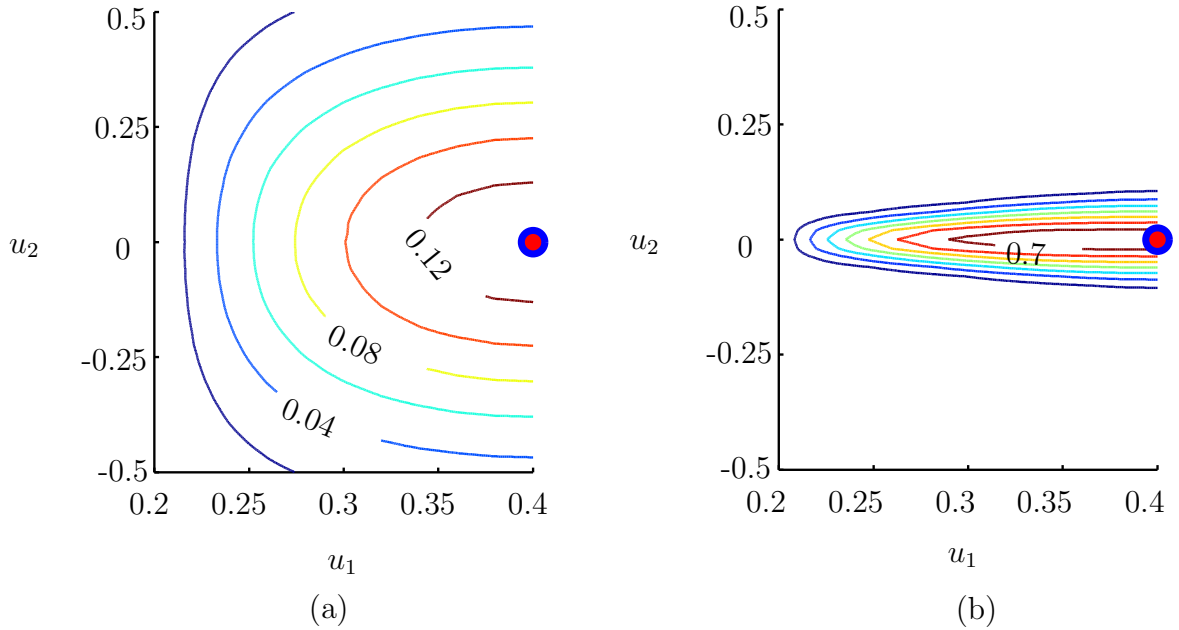


Fig. 4.6: Marginal grasp success probability in terms of two different object posteriors. The optimal gripper configurations are shown by red dots. In (a), $\mathcal{N}(0, 0.3)$ is used to simulate the posterior of the object position. The marginal success probability $P(S)_{u=u_{\text{opt}}} = 0.13$. In (b) $\mathcal{N}(0, 0.03)$ is used to simulate a more accurate posterior distribution of the object position with $P(S)_{u=u_{\text{opt}}} = 0.76$

there is sufficient free space between the pre-grasp configuration and the pre-touch configuration. The collision-free motion from the pre-grasp to the pre-touch can be interpolated linearly. To fulfill the second condition, the choice of a pre-touch configuration g_n is essential, because the grasp success is predominantly affected by the way we make contact with the object. Hence, the grasp success model $P(S|x, \mathcal{G})$ is simplified to $P(S|x, g_n)$.

The dimension of the object state x depends on the choice of the object representation. To represent simple geometry primitives such as cylindrical objects, it is sufficient to choose a parametric representation which includes radius, height, and the position of the object. For objects with irregular form, the dimension of x can be very large. However, predicting the grasp success probability can be independent of the choice of representation, since it is only required to extract the contact normals of an object.

In the following, we elaborate on how we approximate the conditional success probability of a given pre-touch configuration g_n for real world objects. Here we only study the pinch grasp, where the grasp only requires two fingers to make contact with the object. Pinch grasps can be easily realized by most grippers, independent of how many fingers a gripper they have. We choose four criteria to model the conditional grasp success probability.

- *Non-zero contact angle:* The first criterion that we use to approximate the grasp success is the angle between normal of fingertips and desired contact regions. In Fig. 4.7, we denote n_i as the normal defined on a finger tip and c_i as the normal of

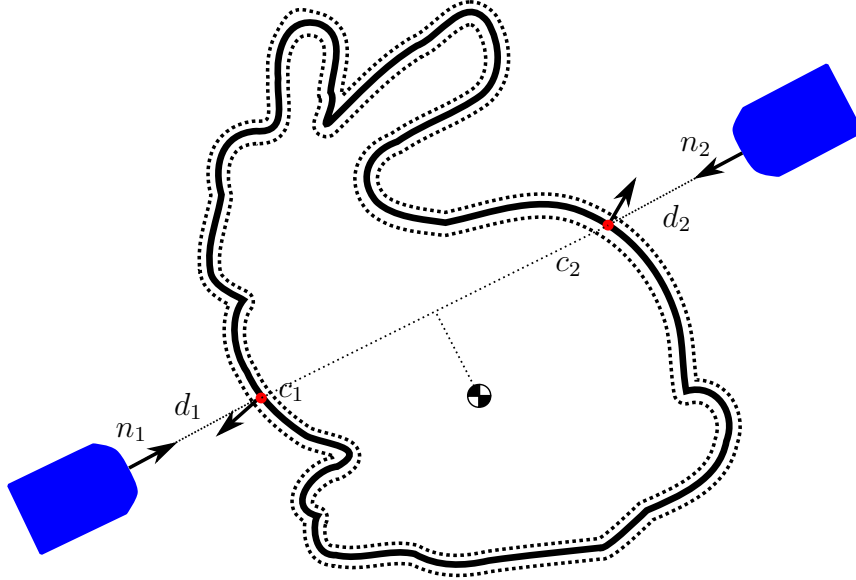


Fig. 4.7: Description of variables which represent a pre-touch configuration g_n on a bunny object. Fingertips are illustrated in blue. Dashed lines indicate variances of the representation. Arrows represent normals on the particular surfaces.

a contact region. d_i is the distance from a finger tip to a contact region. The grasp success probability with a non-zero contact angle is given by:

$$P_1 = \begin{cases} \prod_{i=1}^k (1 - \frac{1}{0.5\theta} |\beta|) & |\beta| < \frac{\theta}{2} \\ 0 & \text{else} \end{cases}, \quad (4.5)$$

where $\beta = \pi - \angle(n_i, c_i)$ and $\angle(n_i, c_i)$ is the angle between n_i and c_i . k denotes the total number of fingertips used to make contact with the object. θ is the maximal angle of a friction cone that the finger tip still supports the object without slipping.

- *Non-optimal finger placement:* Due to the non-optimal control of a gripper during force closure, the closer the fingertips are located to the object surface, the less likely a control error would occur to influence the overall grasp success. Therefore this criterion is modeled by

$$P_2 = \exp\left(-\sum_{i=1}^k \frac{d_i - h}{\sigma_h^2}\right) \quad (4.6)$$

where h is the desired offset distance between fingertips and the object's surface for a pre-touch configuration. σ_h is a shaping factor for the probability.

- *Distance to center of gravity:* This criterion penalizes a grasp where the line connecting two contact points does not cross the object's center of gravity. A large distance between the center of gravity to the line may result in that the object rotates within the gripper after being lifted up. We model the success probability of this criterion

by

$$P_3 = \begin{cases} 1 - \frac{1}{0.5 \cdot l_{bb}} |d_{cg}| & |d_{cg}| < 0.5 \cdot l_{bb} \\ 0 & \text{else} \end{cases}, \quad (4.7)$$

where d_{cg} denotes the distance from the line of contact points to an object's center of gravity. For unknown objects, we do not know exactly where the center of gravity lies. However, based on the reconstructed object model we can fit a bounding box to the object. The center of the bounding box is a good estimate of the center of gravity. l_{bb} is the largest side length of the bounding box.

- *Surface uncertainty*: This criterion is specially designed for the representations that model surface uncertainty. This criterion avoids making the finger contact on uncertain surfaces because forces established on uncertain surfaces can not guarantee firm touch between the fingertips and the object. We model the success probability induced by surface uncertainty with

$$P_4 = \begin{cases} \prod_{i=1}^k (1 - \frac{\langle \sigma(c_i) \rangle}{\sigma_{\max}}) & \langle \sigma(c_i) \rangle < \sigma_{\max} \\ 0 & \text{else} \end{cases}, \quad (4.8)$$

where $\langle \sigma(c_i) \rangle$ is an average variance of c_i and σ_{\max} is a parameter representing the maximal allowed variance. We merge the proposed criteria in a weighted fashion to generate an overall grasp success probability by

$$P(S|x, g_n) = \sum_{i=1}^4 w_i \cdot P_i \quad (4.9)$$

with $\sum_{i=1}^4 w_i = 1$.

The proposed model considers uncertainty over other classical methods for evaluating a grasp. Classical methods usually assume the perfect knowledge of object geometry and perfect control of contacts, using these method usually fails to grasp the object in the reality. The first criterion defined in the proposed model captures the necessary condition of constructing an antipodal grasp. It ensures, the lines connecting the contact points must lies in the friction cone. Other criteria are designed to handle contact uncertainty and shape uncertainty which can not be avoided in real world grasping.

4.5 Modelling perception uncertainty

4.5.1 Surface based probability distribution estimation based on probabilistic fusion

The aforementioned shape-based approach that aims to represent the object with a limited number of parameters does not scale to real world objects. The reason is that the shapes and surfaces of most real world objects are complex and individual. Grasping these objects requires a robot to have the capability of learning the form of irregular objects online even the object is presented for the first time. In this section, we propose a new method to estimate the probability distribution of the surface of the objects. Rather than predicting the shape parameter, this method estimates the course of the object surface achieved by a new probabilistic object representation and a multi-sensor fusion approach. Our surface representation has some significant advantages. First, it models the measurement uncertainty explicitly, as the fusion method explicitly takes it into account. Second, the model can be reconstructed with multiple measurements generated by various sensors, which have different noise characteristics. Third, one can parallelize the fusion method on GPU, so that the reconstruction process is performed in real time. Fourth, the efficient query of the surface information allows fast grasp synthesis.

4.5.2 Surface representation: p-SDF

The model of the surface representation is based on signed distance field (SDF). A SDF in a metric space determines the distance between a point and the object surface. The sign of the distance indicates whether the point is outside or inside the object. Besides the distance information, we augment the SDF with another additional variable σ to represent the uncertainty of the distance. We call this new probabilistic surface representation p-SDF. In this work, a discretized version of p-SDF is used. We discretize a predefined metric space, where p-SDF is defined, into equally sized voxels. We denote the set of all the voxels as V . Since the modeling of an object can never be perfect, we have to take the surface uncertainty of modeling into account. For this purpose, we define the distance to each voxel as a normally-distributed random variable. Thus, an object is represented by

$$\mathbf{f} : v \mapsto \mathcal{N}(\mu, \sigma^2), \forall v \in V, \quad (4.10)$$

where μ is the mean and σ is the variance of the distribution assigned to voxel v .

4.5.3 Surface reconstruction with sensor fusion

In order to fuse the measurements into the representation using multiple sensors, an iterative method is used to update all the voxel elements. Considering only one voxel $v \in V$, we

introduce a random variables D_k associated with voxel v . Here, k denotes a time stamp. $\text{Bel}(D_{k-1})$ and $\text{Bel}(D_k)$ represent the belief signed distance distributions at time stamp $k-1$ and k respectively. We also assume that the major error of one sensor measurement can be approximated as Gaussian white noise so that we can define a measurement model of the sensor by

$$P(z|c, \Omega) = \mathcal{N}(z^*, \sigma_{\text{sensor}}^2), \quad (4.11)$$

where c denotes a depth camera pose and Ω denotes the surface of an object. z^* is the true distance between the surface and the depth camera. Furthermore, we denote the true distance between the voxel v to the surface Ω as z_{sdf}^* . From Fig. 4.8 we can derive

$$z_{\text{sdf}}^* = d_{(v,c)} - z^* \quad (4.12)$$

where $d_{(v,c)}$ denotes the distance between the voxel v and the depth camera. Since z follows a Gaussian distribution, z_{sdf} also follows a Gaussian distribution, which is given by

$$P(z_{\text{sdf}}|c, \Omega) = \mathcal{N}(z_{\text{sdf}}^*, \sigma_{\text{sensor}}^2). \quad (4.13)$$

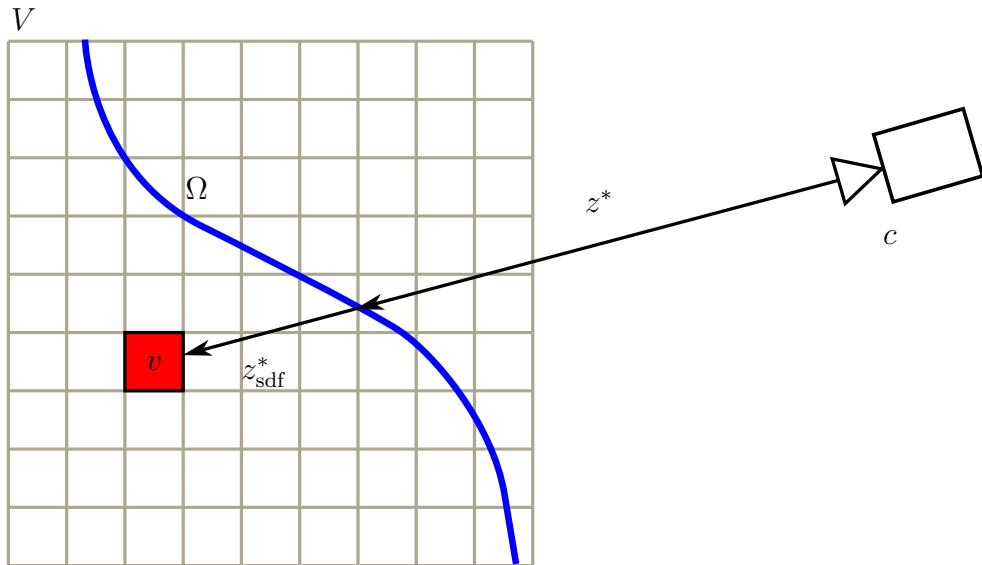


Fig. 4.8: Description of the variables that we defined. The grid denotes the volume we consider for modelling an object. z^* is the true measurement from a depth camera. c denotes the pose of the depth camera. z_{sdf}^* is the true signed distance along the direction of the measurement.

The belief distribution $\text{Bel}(D_k)$ of voxel v can be derived by Bayesian recursive updating by

$$\begin{aligned}
 \text{Bel}(D_k) &= \text{P}(D_k | z_{\text{sdf}}^1, \dots, z_{\text{sdf}}^k) \\
 &= \eta \cdot \text{P}(z_{\text{sdf}}^k | D_k) \cdot \text{P}(D_{k-1} | z_{\text{sdf}}^1 \dots z_{\text{sdf}}^{k-1}) \\
 &= \eta \cdot \text{P}(z_{\text{sdf}}^k | D_k) \cdot \text{Bel}(D_{k-1})
 \end{aligned} \tag{4.14}$$

This equation shows that $\text{Bel}(D_k)$ can be computed iteratively by the measurement model and the belief prior to the measurement update. In this work, we propose to approximate the belief with a Gaussian distribution for each voxel, so the belief $\text{Bel}(D_k)$ can be further computed by

$$\begin{aligned}
 \text{Bel}(D_k) &= \eta \cdot \mathcal{N}(z_{\text{sdf}}, \sigma_{\text{sensor}}^2) \cdot \mathcal{N}(\mu_{k-1}, \sigma_{k-1}^2) \\
 &= \eta' \cdot \mathcal{N}(\mu_k, \sigma_k^2)
 \end{aligned} \tag{4.15}$$

with

$$\mu_k = \frac{\mu_{k-1} \cdot \sigma_{\text{sensor}}^2 + z_{\text{sdf}} \cdot \sigma_{k-1}^2}{\sigma_{k-1}^2 + \sigma_{\text{sensor}}^2} \tag{4.16}$$

$$\sigma_k^2 = \frac{\sigma_{k-1}^2 \cdot \sigma_{\text{sensor}}^2}{\sigma_{k-1}^2 + \sigma_{\text{sensor}}^2}. \tag{4.17}$$

We use Equ. 4.16 and Equ. 4.17 to update the whole volume V . To update the volume with another depth camera, one only needs to exchange σ_{sensor}^2 to the variance which represents the measurement characteristics of the camera.

4.6 Searching for feasible grasp configuration

In general, numerous pre-touch configurations are feasible for a successful grasp, since most objects can be stably grasped in different ways. The marginal probability $P(s|g_n, \mathbf{f})$ is a high-dimensional non-convex function on g_n . g_n is parameterized by the pose of the gripper relatively to the object and the positions of joints. Finding the optimum of g_n is not trivial. We approach this problem in two steps. First, we use a generalizable method to parameterize the grasping posture which reduces the dimension of gripper configurations. The parametrization is then used in a two-stage search process, which involves sampling a set of configurations from a uniform distribution and refining the finger joint configurations of the best sample.

4.6.1 Reducing configuration space of grippers using synergy

We follow the concept of ‘eigen-grasp’[17] to parameterize the gripper. This concept is based on the grasp synergy which allows the joint positions to be controlled only by a subspace of the total available degrees of freedom. In our work, we use grippers with more than two degrees of freedom (DOF). Given a gripper with d DOF, we organize joints on the same finger into a group so that only one parameter is required to control the whole finger. We define the set of joint groups as \mathbb{J} . Later we use this reduced parameter space to search for grasps. After defining the groups, we define a set of grasp types \mathbb{T} that are feasible for the gripper. Typically, they indicate which grasp type is reasonable for a specific object or a specific task, e.g. a pinch grasp is more suitable for grasping a chopstick than a power grasp. A grasp type is parametrized by an opening posture $P_b \in \mathbb{R}^d$ and a closing posture $P_e \in \mathbb{R}^d$. To generate a gripper configuration of a particular type, we must first compute an eigen grasp matrix M for this type. The algorithm for computing M is given by Algorithm 3.

Algorithm 3 Compute M_t of grasp type t

```

1: Input:  $\mathbb{J}, P_b, P_e, d$ 
2:  $n_g = |\mathbb{J}|$ 
3:  $k = 0$ 
4:  $r = 0$ 
5: resize  $M$  to  $d$  rows and  $n_g$  columns
6: for  $i$  from 1 to  $n_g$  do
7:   for  $j$  from 1 to  $\mathbb{J}[i]$  do
8:     row =  $r + j$ 
9:     col =  $i$ 
10:     $M(\text{row}, \text{col}) = P_e[k] - P_b[k]$ 
11:     $k = k + 1$ 
12:   end for
13:    $r = r + n_g$ 
14: end for
15: return  $M$ 

```

The entire joint configuration of a grasp \mathbf{j} is computed by

$$\mathbf{j} = M \cdot \alpha + P_b \quad (4.18)$$

where M is a $d \times n_g$ matrix, α is a $n_g \times 1$ vector and n_g is the number of groups we defined for the gripper. Also note that $\forall \alpha_i \in \alpha$, α_i is defined in $[0, 1]$. Thus the joint configuration \mathbf{j} equals P_b when $\forall \alpha_i \in \alpha$, $\alpha_i = 0$ and equals P_e when $\forall \alpha_i \in \alpha$, $\alpha_i = 1$.

Fig. 4.9 depicts an example how we parametrize a SCHUNK-SDH gripper. SCHUNK-SDH has a total of three fingers. Each of them has two joints with independent actuators. Two of three fingers have an additional joint at the finger root. These two joints are driven by the same actuator to change the directions of force closure. We define in total three

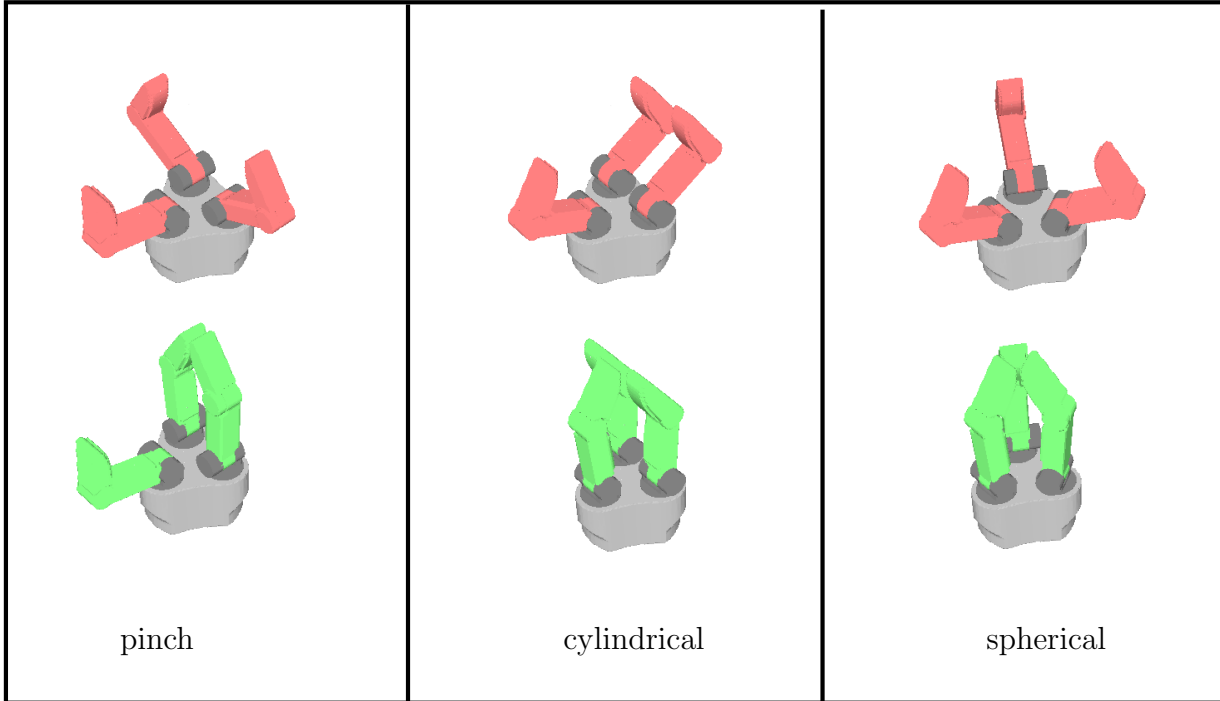


Fig. 4.9: Three grasp types are defined for SCHUNK-SDH gripper. The fingers of the gripper for an opening grasp posture P_b are marked in red, the ones for a closing grasp posture P_e are marked in green.

grasp types and their corresponding parameters P_b and P_e . Which grasp type to choose is normally related to the object shape as well as task intention. We organize two joints on the same finger into one group and additionally the two coupled joints into another group. The total amount of the joint groups adds up to four for this gripper. In this chapter, we do not consider the task intention after grasping, so we choose the pinch grasp in the following experiments. Pinch grasp mimics the most widely used parallel gripper and is capable of picking a large amount types of objects.

4.6.2 Calculation of surface normals

Surface normals on the desired contact points are used to evaluate the conditional grasp success probability. For the objects which can be represented by p-SDF, the surface element can be accessed by an operation called ray-casting. Ray casting evaluates the function along a given direction until a zero-crossing is found. To employ this method for grasp generation, we attach a set of frames organized by a grid to each finger tip. These frames are then used as a set of simulated proximity sensors to determine the distance between fingertips and the object surface. The region covered by these frames can be considered as a desired contact patch region. Fig. 4.10 depicts a realization of simulated proximity sensors on a SCHUNK-SDH Gripper.

Given an object modeled by p-SDF and a pre-touch configuration of a gripper, the desired contact normals are evaluated by first conducting ray casting for each frame. For

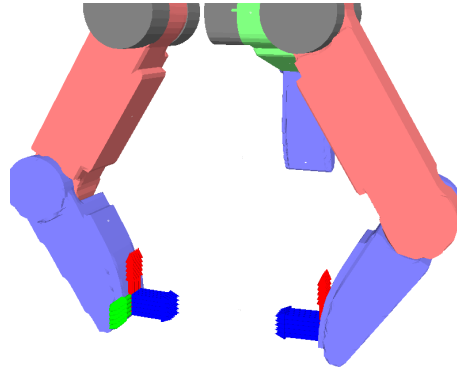


Fig. 4.10: For each finger tip, 25 Frames are attached to the region, where contacts and forces are applied during grasping. Arrows in blue show the direction of operating ray castings.

this gripper, we obtain 25 desired contact points which form a contact region. We use these points as inputs for a RANSAC plane segmentation [136]. The contact region is regarded as feasible, if at least 80% of points are considered as inlier of a plane. This rules out unstable contact region where the course of an object surface is not smooth. The normal of the contact can be calculated from the plane segmentation result. In this way, we avoid using a point-contact model to evaluate contacts but use a contact patch for estimating the contacts and contact normals. It simulates physical contacts in a real grasping scenario. Stable contacts are usually established on the plane of two surfaces, not on the point. Fig. 4.11 depicts the result of the contact points (magenta) and surface normals (yellow arrows) determined by ray casting.

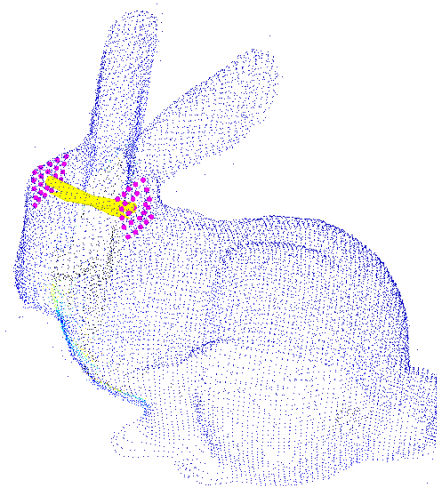


Fig. 4.11: Desired contact positions of a pre-touch configuration determined by ray casting. Magenta: contact points, yellow arrows: surface normals. For the purpose of visualization, this model is generated in Gazebo simulation.

4.6.3 A sampling-based approach to grasp synthesis

The objective function $P(S|g_n, \mathbf{f})$ to be maximized is in general non-linear and non-convex. Directly searching in the parameter space is exhausted and tends to find local-optimal solution, because the dimension of parameter space is quite high. However, by using other scene or task information such as where the robot is performing grasp or what to do after grasping, we can not only reduce the dimension of search space, but also gradually approach a good grasp configuration by selecting the order of parameters to be optimized.

The parameter space of $g_n = \{\mathbf{p}, \alpha\}$ contains a pose parameter \mathbf{p} in \mathbb{R}^6 and a posture parameter α in \mathbb{R}^{N_g} . The pose parameter can be used to place the gripper at an almost satisfactory region, in which simulated proximity sensors have valid measurements, while the posture parameter is used for the fine adjustment of the fingertips. The entire gripper joint configuration is computed by the posture parameter and the eigen-grasp matrix.

To determine the pose parameter of the gripper, we use a coarse-to-fine sampling strategy. Instead of sampling the whole parameter space, we can first choose a subset of pose parameter for starting the search. Which subset to choose depends mainly on the task intention and scene constraints. For example, in a table-top scenario one may choose to grasp the object from its top, thus we can start to sample position and the yaw angle of a gripper. After coarse sampling, one may already find some feasible grasps. In order to further increase the probability of grasping success, we sample the pose configuration uniformly around the best configuration in a two-dimensional plane. The plane should be parallel to the plane formed by the fingertip. A plane for the gripper used in our experiment is shown in Fig. 4.12.

After coarse-to-fine sampling, the pose parameter of the gripper is determined. However, the distance from the individual fingertips to the object may still different. In order to keep the object as static as possible during the grasping process, we iteratively increase the posture parameter and apply the half distance difference as offset to the position of the gripper, until a distance threshold is reached.

4.7 Sensitivity analysis of p-SDF

Experimental setup The aim of this experiment is to analyze the result of proposed method reacting to different level of sensing noise. We use Gazebo to simulate a real grasping scenario as shown in Fig. 4.13. We sample noise from a Gaussian distribution for each pixel of a depth image to simulate the noise characteristic of a real depth camera such as ‘Creative Senz3D’. Then, the noise are added to the perfect depth images provided by the simulator. Fig. 4.14 depicts the point cloud which are generated from the synthetic depth images. Only the hand camera is used in this experiment. The object is observed by moving the arm to several pre-defined configurations. After the object is fused by the measurement, we run the grasp synthesis method and choose the number of samples to 1000 to ensure the convergence of the method. The grasp execution is not performed in

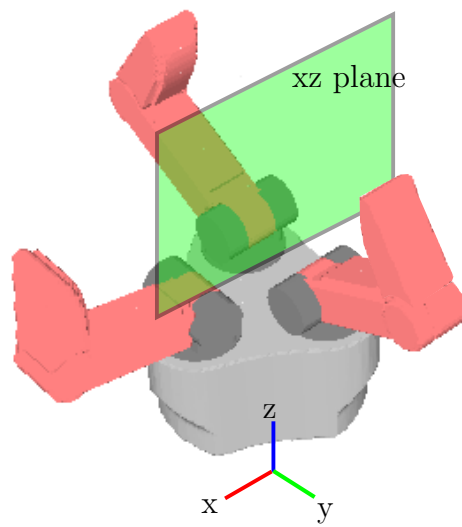


Fig. 4.12: Parametrization for refining a pre-touch configuration.

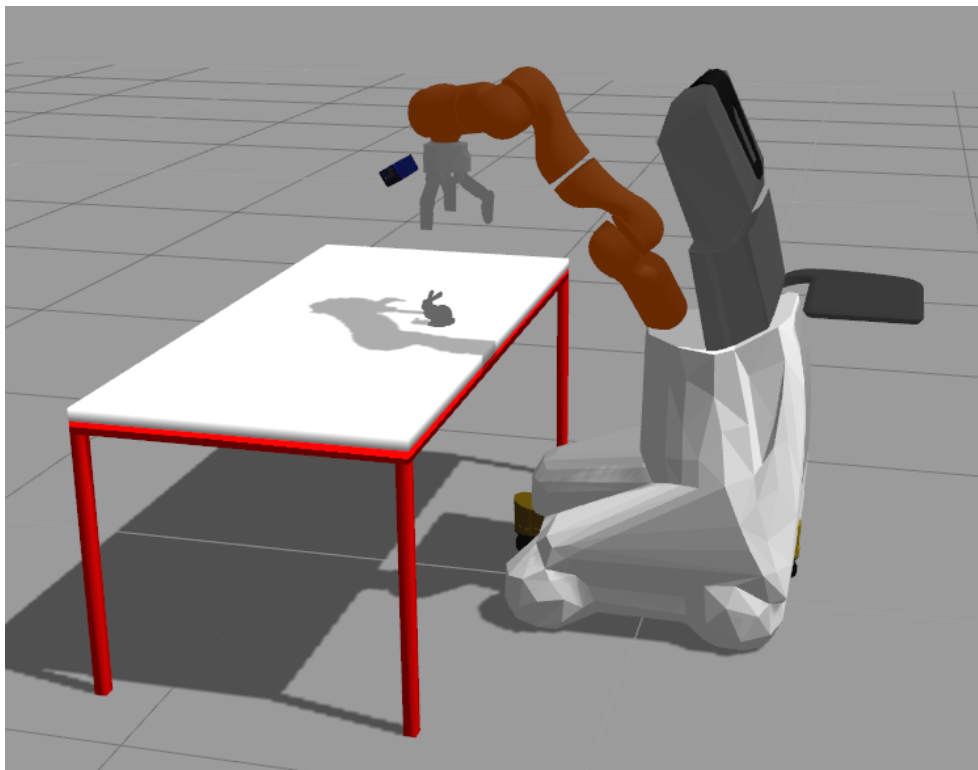


Fig. 4.13: Simulation environment.

this experiment because the simulator does not provide a sufficient good simulation for grasp dynamics of reality.

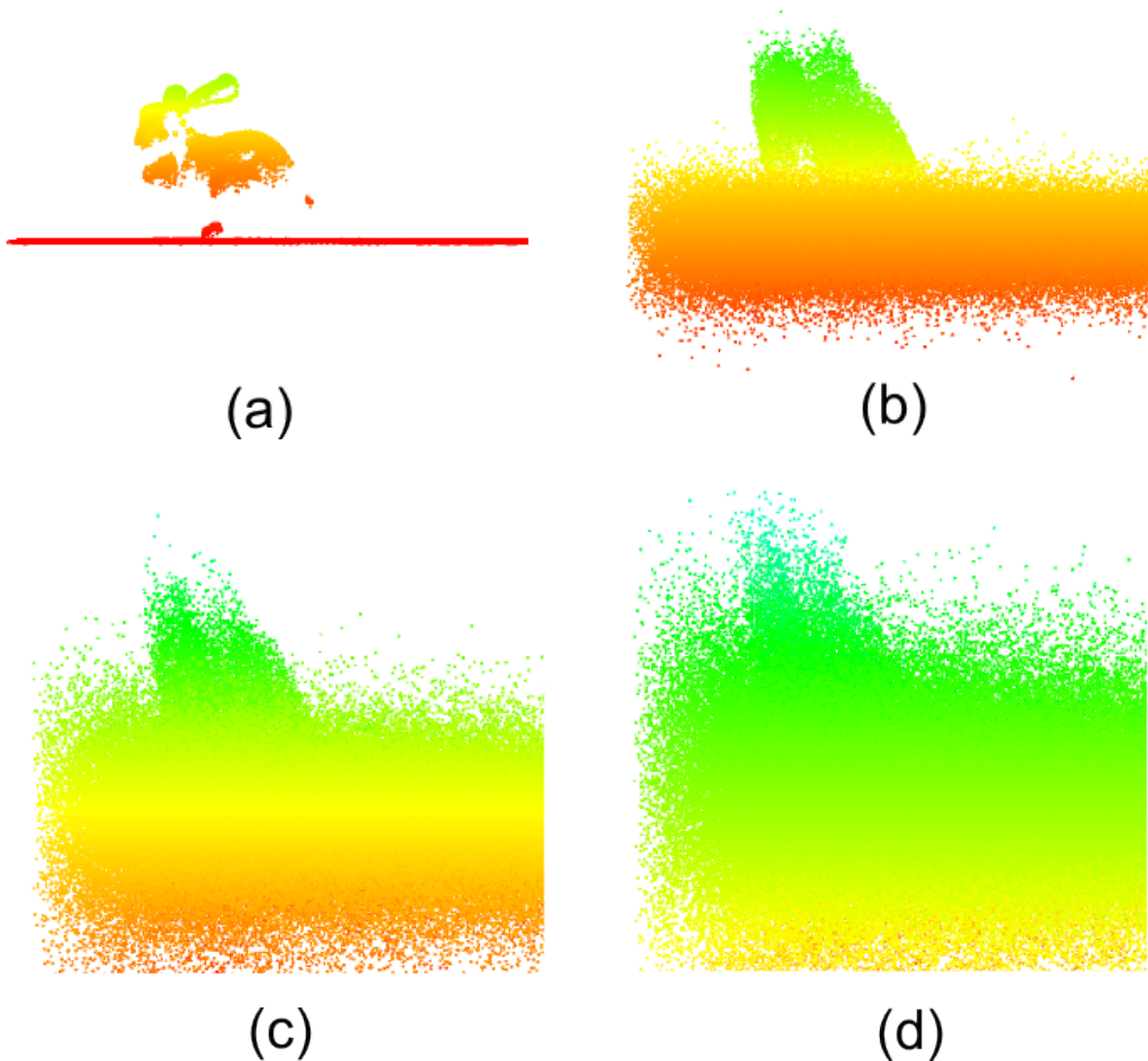


Fig. 4.14: Synthetic noisy measurements with different standard deviation(sd). (a) original measurement (b) $sd = 1$ cm (c) $sd = 2$ cm (d) $sd = 3$ cm

Result Four different levels of noise are compared in the experiment. As shown in Fig. 4.14, the shape of the ‘Bunny’ can only be recognized when no synthetic noise is applied to the measurement. Directly inferring grasps from a single noisy measurement can be very hard. Since our method fuses the object surface by using a series of measurements, the object surface can still be well modelled in the case of 1 cm and 2 cm noise level (Fig. 4.15). Our grasp synthesis method is succeeded in finding a grasp configuration in the most cases except for the case where 3 cm standard deviation of noise is used. The

failure lies in that the noise is so large that the surface can not be well modelled. In addition, a smooth contact area according to the method of surface query can not be found. For the case where 1 cm standard deviation is used, our algorithm finds almost the same grasp configuration as in the noise-free case which shows that our method is robust upto a certain level of measurement noise.

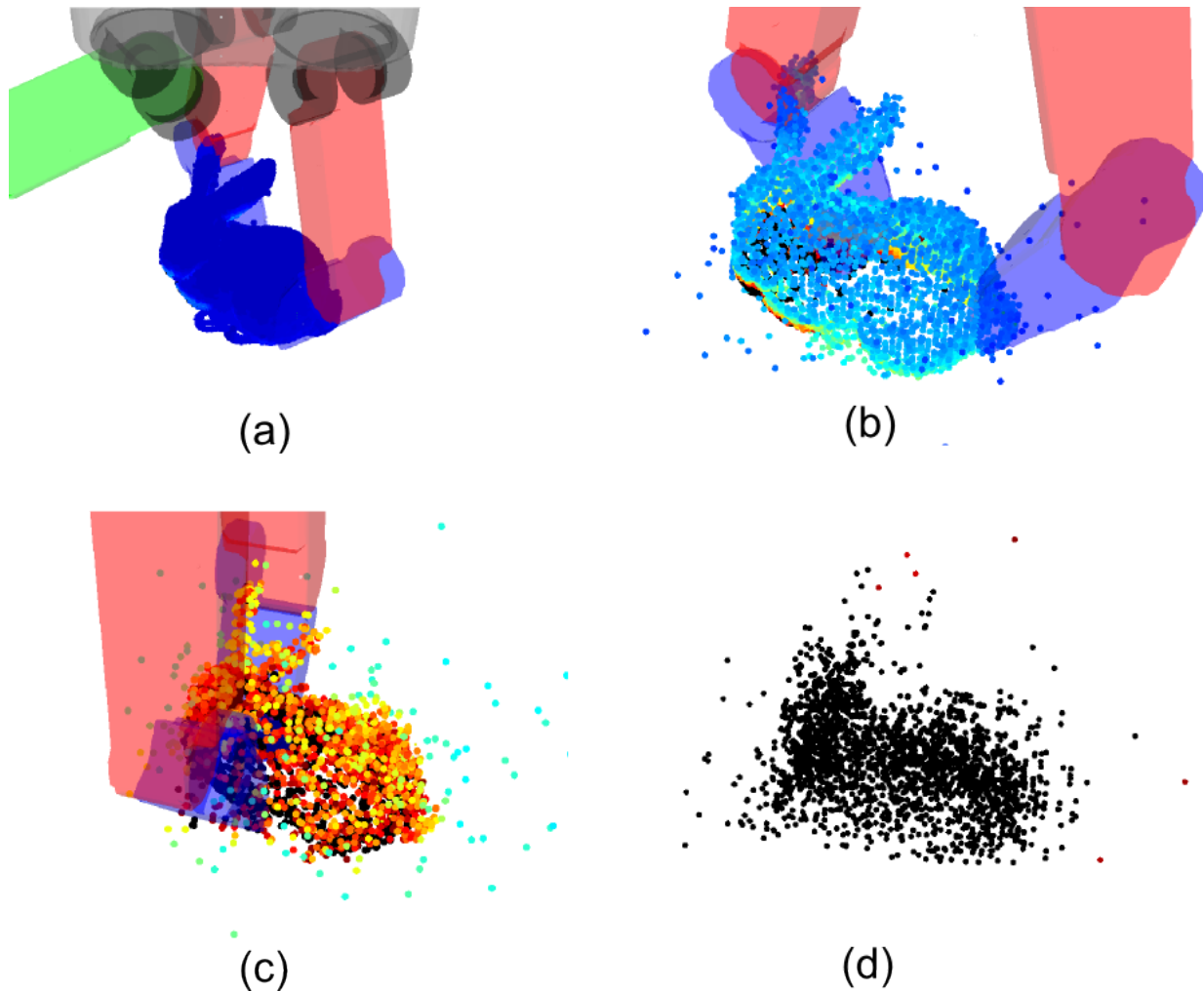


Fig. 4.15: Result of grasp synthesis and reconstructed objects based on measurement with different noise level. (a) original measurement (b) $sd = 1$ cm (c) $sd = 2$ cm (d) $sd = 3$ cm

4.8 Grasping experiments

Unknown objects are the objects which are presented to a robot for the first time. All of the objects we chose for this experiment have their individual shapes, which additionally increase the difficulty. In the following, we first introduce the experimental setup, followed by the evaluation of the algorithm and the performance evaluation of the overall grasping accuracy.

4.8.1 Experimental setup

The test objects that we used in the experiment are shown in Fig. 4.16. Every time we place one object on the table. The robot is required to lift the object 10 cm above the table. If the robot grasps the object successfully, it rotates its wrist of a random angle and places the object back to start a new grasping attempt. For each object, the robot repeats the experiment autonomously for 20 attempts. Human intervention is only involved if the robot fails to pick up an object. In the case that the robot does not recognize the failure and continues to place back the object, we manually remove the object to prevent the robot from breaking its gripper.

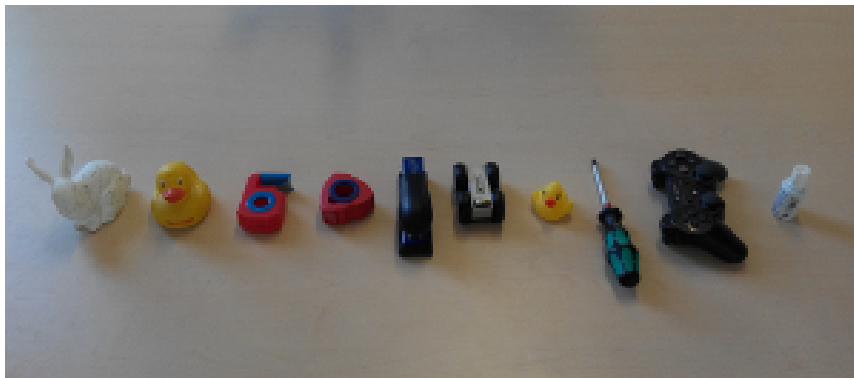


Fig. 4.16: Test objects we used in this experiment. From left to right: a 3-D printed ‘bunny’, ‘big duck’, ‘tape rectangular’, ‘tape triangle’, ‘stapler’, ‘toy car’, ‘duckie’, ‘screw-drawer’, ‘PS3 joystick’, ‘correction fluid’.

Hardware

We conduct the experiments with a fully integrated mobile manipulation robot (Care-o-bot) [42]. The robot is equipped with a 7 DOF KUKA lightweight robot with a 7 DOF SCHUNK-SDH gripper mounted on the wrist. The robot is driven by an omnidirectional mobile platform. The primary sensors of the robot include a head-mounted Microsoft Kinect sensor and a hand-mounted stereo camera (Ensenso N10).

Object modelling

Before each grasp attempt, the robot moves its gripper to several pre-defined poses so that the object is perceived from both head camera (Kinect) and in-hand camera (Ensenso). During the movement, our fusion algorithm integrates the measurements simultaneously from both cameras. Fig. 4.17 illustrates the process of modeling the ‘toy car’ by moving the gripper relatively slowly. For the first 9 s, the top side of the object is visible to the in-hand camera, so the uncertainty of the top region is reduced gradually. The black area of the object indicates that the region of wheels is still under large uncertainty. After 9 s the robot moves the in-hand camera to some other viewpoints to acquire more information

about the region of wheels. Fig. 4.18 shows the number of frames used to fuse the object during the entire modeling process.

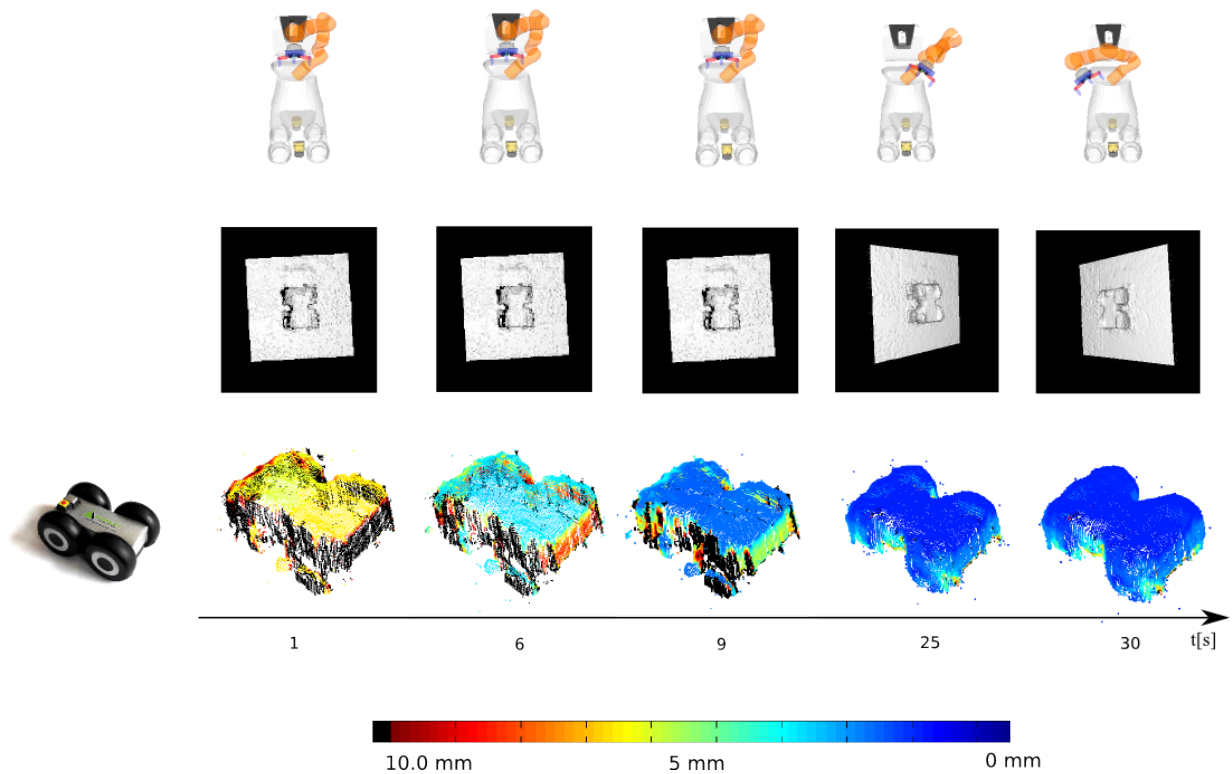


Fig. 4.17: The process of modelling a 'toy car'. Top: The robot moves its in-hand camera to several pre-defined poses to see the object from different viewing angles. Middle: Raycasted views of the model are shown alongside each robot pose. Bottom: A colorized model of the object visualizes the uncertainty of the object during the modelling process. In the first three figures, the black points on the wheels indicate a large uncertainty on those regions, because the object was only observed from its top.

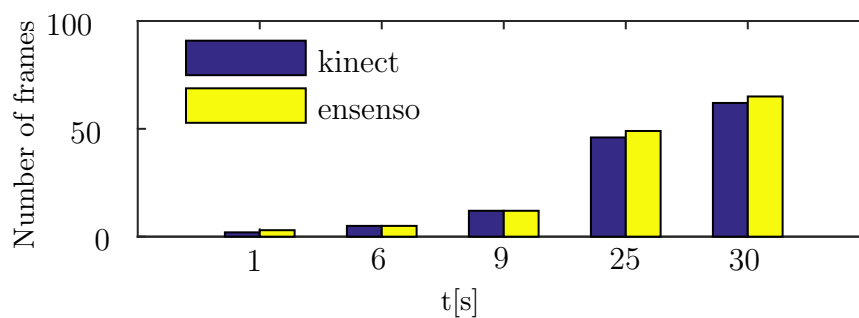


Fig. 4.18: Number of frames used to model the 'toy car' (object see in Fig. 4.16). The time stamp of each bar corresponds to that of modelling result in Fig. 4.17.

Parameter tuning

We choose the bunny object to tune the parameter of weights in Equ. 4.9, because the surface distribution of this object is relatively complex. The intuition behind this is if we have a good evaluation model for a complex object, the model should also be able to generalize for simple objects. Totally we need to determine four weights to evaluate a grasp. Since the sum of all the weights equals one, we only need to tune three of them. In order to compare the effect of each weight, we set one of them to one and the others to zero. Then we use the search algorithm to compute the best grasp configuration. Fig. 4.19a depicts the best grasp we obtain if we only use the first criterion (Non-zero contact angle). The desired grasp position is on one ear of the bunny. The contact normals are almost anti-parallel to each other. However, it is obviously not an optimal grasp. In Fig. 4.19b, only the second criterion (Non-optimal finger placement) is activated, the contacts of the best grasp are on the front and the back of the bunny. The contact normals are however not parallel to each other, which may lead to grasp slippage in reality. In Fig.4.19c, only the third criterion (Distance to center of gravity) is used. The connection line between the contacts passes through the center of gravity, however, the contact normals are not optimal. In Fig.4.19d, only the fourth criterion (Surface uncertainty) takes effect. During the model reconstruction phase, the back of the bunny is visible to many viewing angles. So the best-desired contact points are on the back of the bunny, where the surface has the lowest uncertainty. However, force closure condition of this grasp is not fulfilled. Based on the analysis, the first criterion has a dominant effect which should be given a larger weight to achieve force closure and a good contact condition. But if the w_1 is chosen too large, the best grasp may appear at the side of an object. We need to increase w_3 to penalize this effect. w_2 and w_4 should be chosen as the secondary criterion to motivate more symmetrical contact points as well as making contacts on the surface with low uncertainty. Finally, we find $w_1 = 0.6$, $w_2 = w_4 = 0.1$ and $w_3 = 0.2$ are a good trade-off between the four criteria. In the experiment, we use the same parameter to evaluate grasps on other objects. Although the shape of other objects is way different from the bunny object, the experimental result verifies that the chosen parameters also perform well for these objects.

4.8.2 Performance evaluation on grasp synthesis algorithm

To generate a viable grasp we first fit a bounding box to a point cloud that is segmented from the table. Based on the estimate of the bounding box, we compute for each object an initial pinch grasp, which is aligned to the bounding box. The initial grasp is used to initialize the grasp synthesis algorithm. We evaluate the performance of grasp generation in terms of the number of samples. Fig. 4.20 depicts the time consumption to compute a grasp with a different number of samples. The computation time increases linearly with the number of grasp samples. We get the variance of time consumption by running the algorithm 10 times for a fixed number of samples. By evaluating the algorithm on three

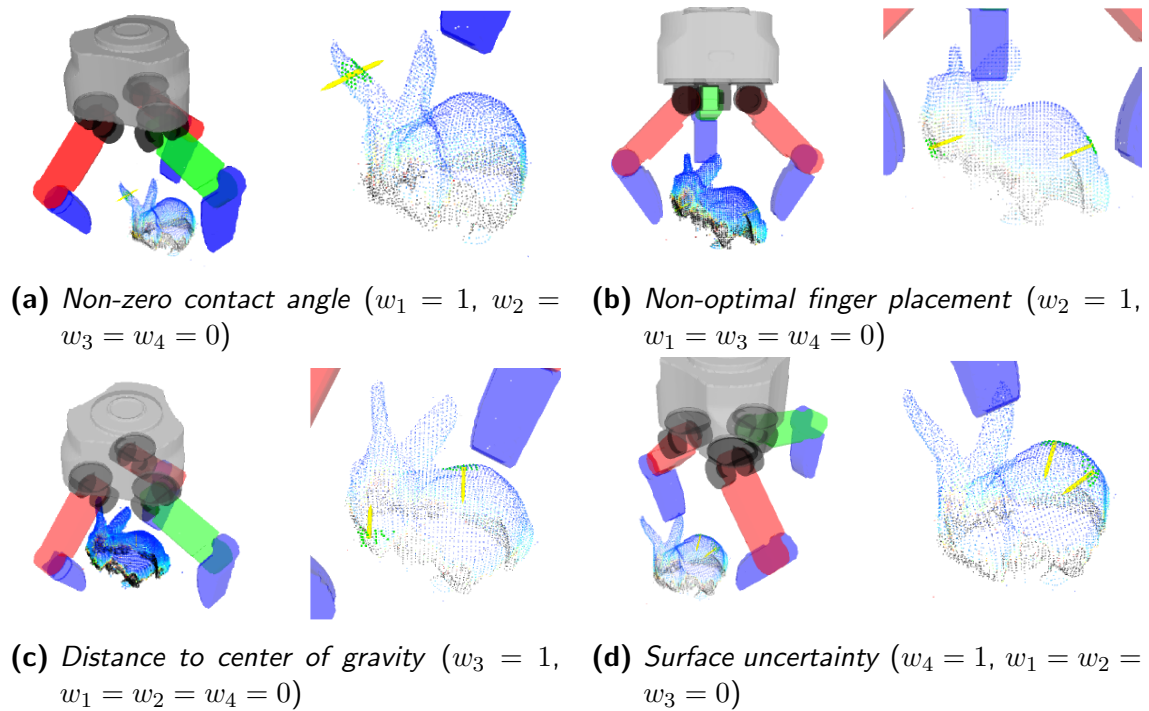


Fig. 4.19: The best grasp configuration found by the search algorithm when only one of the four criteria is activated. Green points indicate the contact regions. The yellow lines show the normal lines of each contact region.

different objects, we also show that the computation time is not correlated with the object type. Fig. 4.21 illustrates the maximum success probability to the number of samples. The maximum success probability can be found after evaluating approximate 200 samples, which means our algorithm can find a viable grasp within 2 seconds. Comparing the speed of a similar work [88], which also uses signed distance function to model objects, our algorithm is at least 30 times faster.

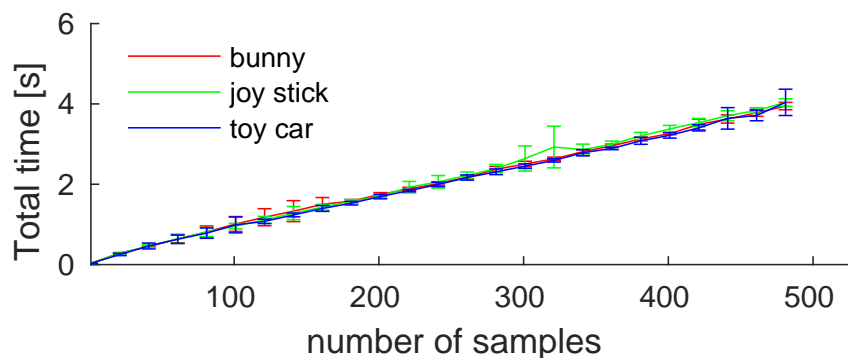


Fig. 4.20: The mean computation time of the grasp generation algorithm scales linearly with the number of samples. The variance is computed by evaluating 10 times for a given number of samples.

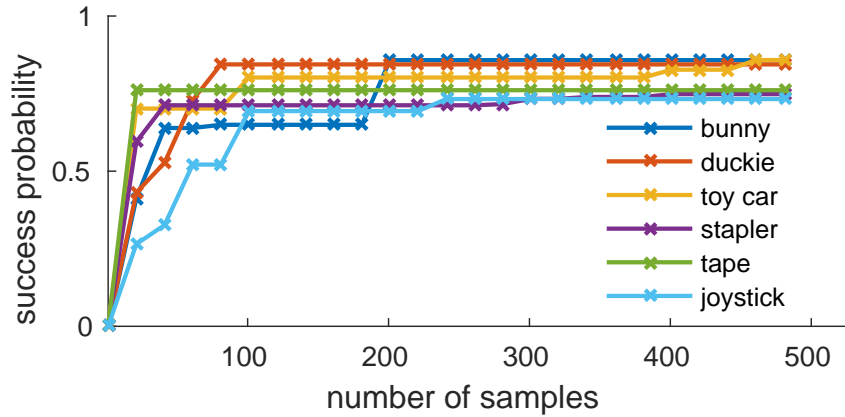


Fig. 4.21: The algorithm converges approximately by evaluating 200 samples.

4.8.3 Performance on grasping accuracy

In our experiments, we define three types of outcomes to judge a grasping attempt. They are success with precision (S.w.P), success (S.) and failure (F). S.w.P means the object is successfully picked up and placed back while the gripper does not displace the object more than 1 cm or more than 30 degrees during force closure. An outcome is marked as S., when the object is successfully picked up in spite of being displaced or rotates within the gripper. An attempt is marked as failure if the object is not successfully picked up or slips out of the gripper.

Table 4.1 summarizes the result for a total number of 200 grasping attempts. Our algorithm has achieved 93.5% success rate of lifting the objects and 79.5 % success rate of grasping with precision. The robot manages to grasp 'duckie' and 'correction fluid' without failure. 7 objects are successfully grasped with only a single failure. Fig. 4.23 illustrates some successful grasps generated from our algorithm.

Among all the objects we used in the experiments, 'PS3 Joystick' is the most difficult object for the robot to pick up. Depending on the viewing angle, the hand-mounted Ensenso stereo camera sometimes does not provide reliable depth images so that the object model is only partially integrated or the integration contains large uncertainty. Noted that there are only a few viable pinch grasps which can be used to lift the 'PS3 Joystick'. These viable grasps are all located in the middle. When the contact regions for these viable grasps are not well modeled or have large uncertainty, our algorithm is not capable of finding them. These phenomena are also observed for the 'screw driver' and the 'stapler'. For these two objects, the region where the surface is black results in a large uncertainty in the model. Since these objects contain more feasible grasps, our robot still manages to pick up them successfully. However, the robot has a lower success rate on these objects if S.w.P criterion is used to assess the grasp outcome. Fig. 4.22 depicts one common grasp generated from our algorithm for the 'stapler'. Our algorithm tends to find grasps at the end side of the 'stapler', because that region are well modeled and certain to the robot. Since this grasp cannot counteract the wrench generated by the gravity, as soon as the

object is lifted up, an in-gripper object rotation may happen.

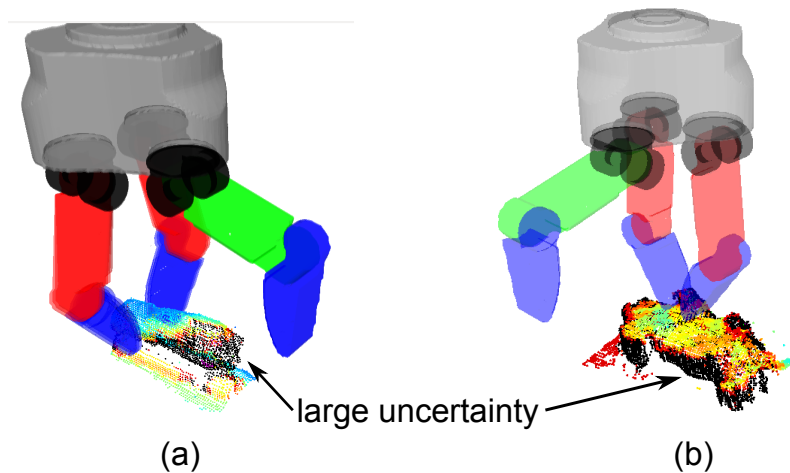


Fig. 4.22: (a) A grasp generated by our algorithm for the 'stapler'. Our algorithm tends to generate grasp in the region where the uncertainty is small, which leads to an non-optimal grasp for this case. This grasp may lead to an in-gripper rotation, because it cannot counteract the wrench of gravity. (b) The black points indicate that the side surface of the object has a large uncertainty, however these regions are the only good contact region for a stable grasp. Since our algorithm avoids to generate grasps on uncertainty surfaces, an non-stable grasp is found at one control axis of the 'PS3 Joystick'.

Object	S.	F.	S.w.P
Bunny	19/20	1/20	15/20
Big duck	19/20	1/20	16/20
Tape rectangular	19/20	1/20	19/20
Tape triangle	19/20	1/20	19/20
Stapler	19/20	1/20	14/20
Toy car	19/20	1/20	18/20
Duckie	20/20	0/20	20/20
Screwdrawer	19/20	1/20	11/20
PS3 Joystick	14/20	6/20	7/20
Correction fluid	20/20	0/20	20/20
Summary	93.5%	6.5%	79.5%

Tab. 4.1: (S. = success, F. = failure, S.w.P = Success with precision). Experimental result for a total of 200 grasping attempts. An outcome is classified as S.w.P when gripper successfully picked the object without displacing the object more than 1 cm or 30 degree. This criterion is strict comparing to a standard way to access grasping success.

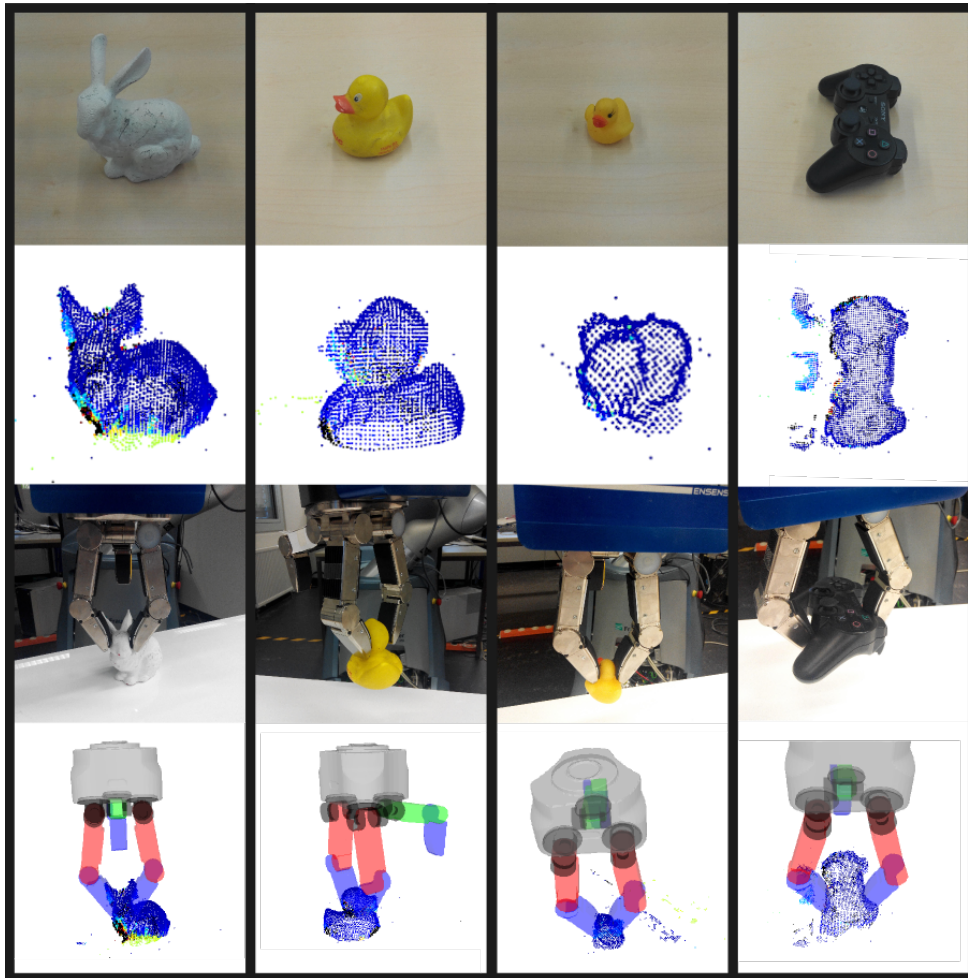


Fig. 4.23: Grasps generated by the algorithm with successful execution.

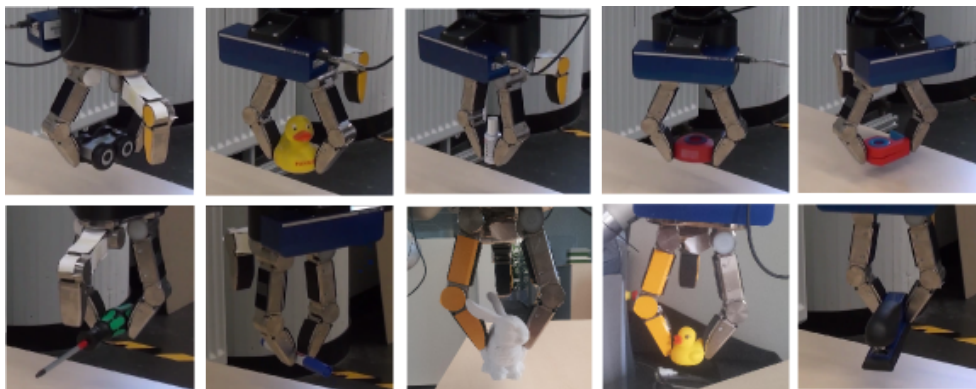
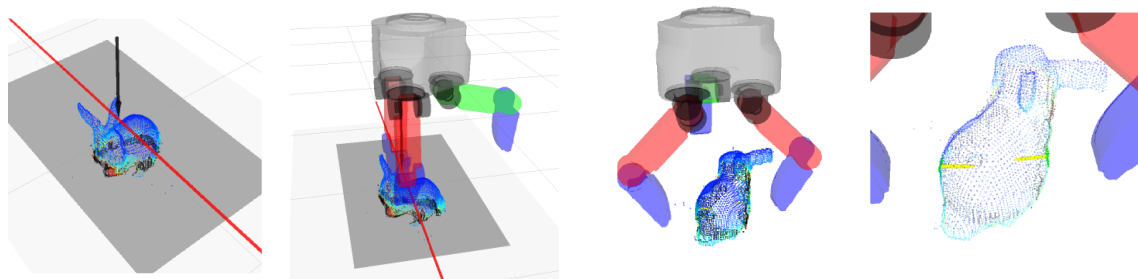


Fig. 4.24: Successful grasps executed by the robot.

4.8.4 Comparison with one baseline approach

We choose the approach proposed by Fischinger et al. [36] as the baseline for comparison. This method calculates grasps from a given point cloud using a so-called ‘*Height Accumulated Features*’(HAF). It has been shown that the approach was adapted to different robots and gripper systems for grasping a variety of unknown objects. In order to use their approach for grasping with our system, we need to adapt the calculated grasping points to our gripper. We use the implementation from http://wiki.ros.org/haf_grasping in our experiments. The default parameters are used to calculate the grasping points. The output of the method is 2D positions of two grasp points. Additionally, a grasp realization step is required to compute the height of a grasp for executing a real grasp. They implement the grasp realization step in OpenRAVE simulator, in which a mesh of an object is calculated for collision checking. Then they use a heuristic to compute a real grasp: setting a manipulator about 7 cm away from the object. As this implementation is not available for our system, we implement a realization step by setting the height of the grasping points at the middle of an object’s bounding box.

Fig. 4.25 depicts a grasp calculated by HAF and our approach on the bunny object. The grasp calculated by HAF may lead to a grasp failure since the region of contact points lies around the head of the bunny. In these regions, there is no flat surface to establish a stable contact. On the contrary, the best grasp calculated by our method is on the body of the bunny. This is more likely to succeed than the one calculated by HAF.



(a) Left: the best grasp point calculated by the baseline approach. The red line indicates the close direction. The black arrowed line shows the approach direction and the position of a gripper. Right: grasp realization for our gripper. (b) Left: the best grasp calculated by our method with weights ($w_1 = 0.6, w_3 = 0.2, w_2 = w_4 = 0.1$). Right: A zoom-in visualization of the contact points.

Fig. 4.25: A comparison of a grasp calculated by the baseline method and our method. The grasp generated by the baseline approach probably leads to grasping slippage because the method does not consider contact surface as an important factor.

In order to systematically compare both approaches, we conduct the same real grasp experiment on the test object set with HAF. We only use the hand-mounted stereo camera to obtain an input point cloud for HAF as the stereo camera provides more precise data than the Kinect. The input point cloud represents a top view of a test object. The experiment

result is summarized in Tab. 4.2. We exclude the object ‘duckie’ and ‘Correction fluid’ from the comparison because the point cloud which represents them can not be robustly segmented from the tabletop. This is related to the limitation of the camera we use. Due to the segmentation error, a lot of grasp failure occurs for these two objects when using the baseline method.

In summary, our method achieves a lower error rate than HAF based on 320 grasp trials in total. HAF can perform well on small regular objects, while it has difficulty to grasp objects with a complex surface distribution like ‘Bunny’. HAF also not performs well on ‘stapler’ or ‘screw driver’. One reason of failure for these two objects is that the input point cloud sometimes only contains a partial view of the object. The material of the objects may cause reflections, e.g. the input point cloud may not contain the shaft side of the ‘screw driver’, which may cause the calculated grasp being too far from the center of mass. Both approaches have difficulty in grasping ‘PS3 joystick’. The reason for that has been discussed previously.

	Our method	HAF
Bunny	19/20	16/20
Big duck	19/20	18/20
Tape rec.	19/20	19/20
Tape triang.	19/20	20/20
Stapler	19/20	12/20
Toy car	19/20	20/20
Screwdriver	19/20	7/20
PS3 Joystick	14/20	13/20
Nr. of failure	13	35

Tab. 4.2: Successful grasp executed by HAF (baseline method) and our method on a subset of the test objects. Our method achieves a lower error rate than the baseline method.

4.8.5 Discussion

Precise grasping is one of the advantages of the proposed approach. By explicitly modeling the surface geometry and its uncertainty, we can compute grasps with more precision so that during force closure the object is less probable to be displaced. Displacement of the object may result in grasp failure more likely, because the actual contact region are not identical to the one computed by the algorithm.

Most approaches address the problem of grasping an unknown object by finding grasp relevant feature from a single measurement. Single measurement, however, may contain large uncertainty. Partial and occlusion of the measurement usually result in a hard situation for grasp detection. Well defined grasp features such as surface normals can not be used because the region where contacts take place are usually not observable from a single measurement. Our approach handles this problem by integrating multiple measurements

so that we can use well-defined features, namely, the surface normals and the modeling uncertainty to assess the viability of grasps.

Recently, some approach exploits deep learning for grasp classification [109], however gathering training examples require enormous effort. The training data is gathered by running a robot over 700 hours. In addition, the training examples are basically images with a labeled grasp. The learned model may not be able to generalize to another environment with a different lighting condition. Our approach does not require any training data, and still gets a competitive result. The result indicates that for grasping problem choosing of a suitable representation can improve the system performance dramatically.

There are also some limitations in the current approach. As our approach requires a pre-segmentation of the object, applying this method to a cluttered scenario where lots of objects stacked to each other relies on a sophisticated segmentation algorithm. These algorithms may not be available at the moment. In addition, our approach can not handle on reflective objects such as glasses, coins, small screws. In these cases, exploiting other sensing modality such as a color camera may help. Another drawback of our approach is that the robot is required to observe the objects with different pre-defined view angles prior to grasping, this may increase the time of task execution. Regarding this problem, one may exploit active perception approaches to minimize the travel distance while still acquiring sufficient information for a stable grasp.

4.9 Summary

In this chapter, we present a probabilistic approach to address the problem of grasp synthesis for unknown objects. Our method explicitly models the uncertainty for the items which are presented to the robot for the first time. Our method finds the most promising grasp configuration which maximizes the grasp success probability. We especially choose challenging novel objects in our grasping experiment. The results demonstrate that our approach is capable of grasping a range of novel while unknown objects precisely and outperform a state-of-the-art method.

As the proposed method only considers using antipodal grasp for precision grasping of small objects relative to a gripper, it is not directly applicable to a multi-finger gripper to generate a grasp of more than two contacts. However, by configure the grasp synergy of a multi-finger gripper, it is still possible to use the method to construct antipodal grasps to grasp unknown objects. As the method is designed for handling surface uncertainty, applying the proposed method to a cluttered environment requires an additional object segmentation step.

One of the important aspects of the proposed approach is that without machine learning of grasping experience on similar objects, our system still achieves a high success rate in experiments. It may indicate that the choice of object representation and the model of grasp evaluation play a significant role for grasping. Our intuition behind this is that

learning from large scale experiments introduces semantics and high-level information for the choice of grasps, while for low-level precision grasping we still need a suitable object representation to handle uncertainty and achieve robustness. One direction of the future work is to combining other sensor modalities such as RGB information into the current approach so that we can benefit from the result of large scale learning to reinforce the approach with semantics and task level information. Another direction of the future work can be exploiting the environment constraints explicitly to allow contact-rich grasping even in crowded environments.

5 An Adaptive Grasping Control Architecture for Whole Body Mobile Manipulation

Humans use eye and touch to monitor the entire process of grasping. They can change the grasping strategy on the fly when needed. This is why humans are capable of grasping objects very robustly. In this chapter, we address the problem of grasp execution and demonstrate how the grasping can be executed robustly under limited sensing capability.

5.1 Introduction

Grasp execution is one of the most important steps for grasping success. Many earlier works use vision, tactile sensing, proximity sensing to improve the robustness of grasp execution ([32],[49],[53],[58]). Grasp execution on a real robot can be used to test whether the planned grasp configuration is feasible or not. Due to the error of object localization, sensor calibration and uncertain robot motion, the actual grasping configuration may differ from the planned one. Therefore, we need to handle these errors and keep the actual grasp configuration as close as possible to the intended one.

Many approaches ([61],[111],[116]) which address the grasping problem usually assume perfect executions. They usually follow a sense-plan-action paradigm to execute a grasp. In the sensing phase, a robot recognizes, localizes and segments a desired object from the environment. In the planning phase, grasping points and grasping trajectory are calculated. In the action phase, the robot executes the planned grasping trajectory to perform the grasp, usually in open-loop. One problem related to this paradigm is, sensing of object's state is missing in the action phase. If the action phase takes more than several seconds without updating the object's state, the robot is unable to react to actuation error or external perturbation. Humans also use this paradigm to perform grasping but the sense-plan-action cycling time is much shorter. They maintain eye contact to the object until the object is fetched. In addition, they can adapt their grasp motion when object is displaced. In order to achieve maximal efficiency, humans seamlessly combine their locomotion, body, and arm movement for the entire grasping process.

In this chapter, we propose a flexible and adaptive grasping control architecture to imitate human grasping. The flexibility is reflected in the freedom of choice and combination of required actuators in different grasping phases. We define three grasping phases for the entire grasp execution process as shown in Fig. 5.1. During the initial phase, the gripper is far away from the object, so the robot can execute an arm and platform coordinated

motion for approaching the object. In grasp motion phase, the gripper is closer to the object, the robot can stop the coordinated motion and use arm-only movement to reach the pre-touch configuration. In the force-closure phase, the robot can control the gripper and arm simultaneously so that grasping forces are applied in the normal direction. The control architecture enables the robot being adaptive during the grasp execution. The robot can continuously update an object's state and changes the corresponding grasp trajectory. We evaluate the proposed control architecture on our robot in a challenging grasping task: grasping paper folded cylinder. To increase difficulty, we specially choose the sensor which has significant noise. This challenge can test whether the proposed architecture can handle light objects with limited perception capability. The experimental result shows that our approach can handle the challenging grasping tasks, and outperforms a traditional approach.

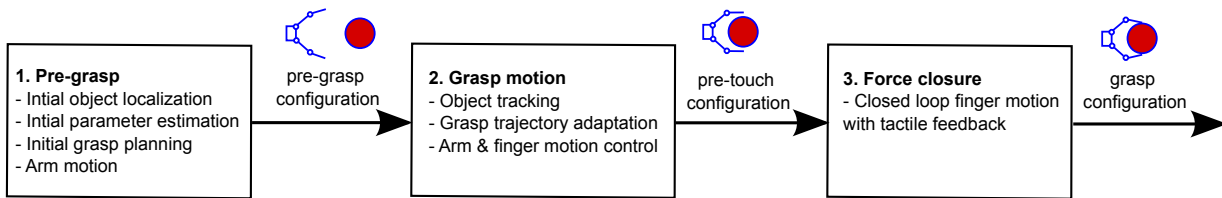


Fig. 5.1: A grasp execution process contains three intermediate grasping phases.

5.2 Related work

A robust grasp execution requires continuous sensing. In general, two types of sensing can be found in the earlier works which address the grasp execution problem. During the reach-to-grasp phase, visual servoing is the primary method to guide the movement of gripper. Many works use the method ([72],[44],[37]) for this task. During the force-closure phase, force and tactile sensing is the promising method to apply sufficient grasping forces to establish a stable grasp. Several works ([86],[21],[20]) use force or tactile sensing for grasp stability assessment and in-gripper manipulation. In addition to the sensor modality, we also give a short review on the control architecture and focus on the methods which allow whole body manipulation.

5.2.1 Methods using visual servoing

Visual servoing is primarily used in the reach-to-grasp phase, in which the gripper moves towards an object yet not in contact with it. Vahrenkamp et al. [128] proposed a hybrid method that combines visual estimations with kinematically determined orientations to control the movement of a humanoid arm. They demonstrated how a robust visual perception is used to control complex robots without hand-eye calibration. Furthermore, they improve the robustness of their system by estimating the hand position in case of failed visual hand tracking due to lightning artifacts or occlusions.

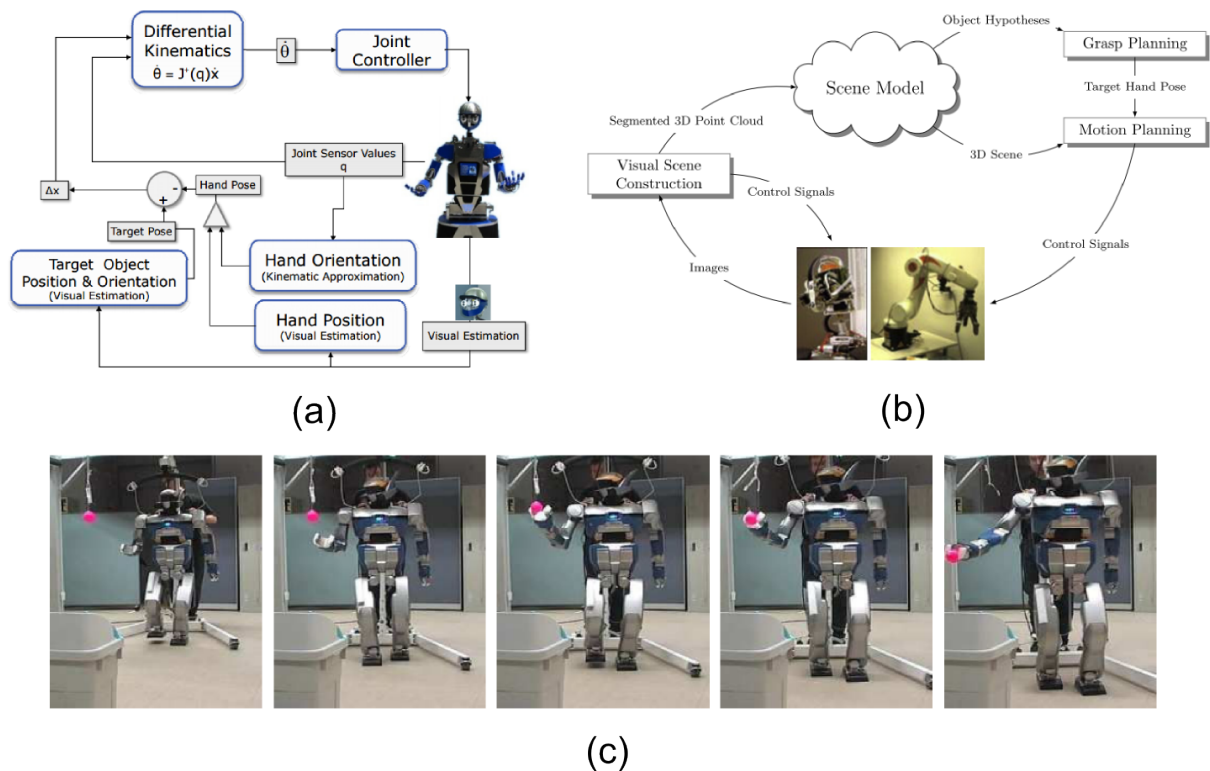


Fig. 5.2: Methods using vision for grasp execution. (a) A method [128] which combines visual perception and robot kinematics for determining motion of grasping. The approach was evaluated with the humanoid robot using the stereo system of the active head for perception as well as the torso and arms equipped with five finger hands for actuation. (b) A method [43] which uses visual servoing on grasping unknown objects. The grasp motion is executed with open loop. (c) A method [89] which allows grasping while walking for a biped humanoid robot.

Recatala et al. [113] employed a stereo camera and a two finger gripper with an eye-in-hand configuration to study visual guided grasping. Their system tracks a pre-defined grasping points in image space and use that for the control law. The tracking is based on using an invariant description with respect to the object.

Different from the above methods which only consider visual servoing of one robot, Muis et al. [96] exploit two robots and demonstrate how a robot with eye-to-hand configuration and another robot with eye-in-hand configuration can improve the tracking performance. The drawback of each configuration is resolved by the benefit of the other.

Traditionally visual servoing can only also be applied to tracking objects with an existing model. Gratal et al. [43] proposed a grasp pipeline which exploits visual servoing also on unknown objects. They demonstrated how visual servoing can be both applied for offline calibration and online grasp execution.

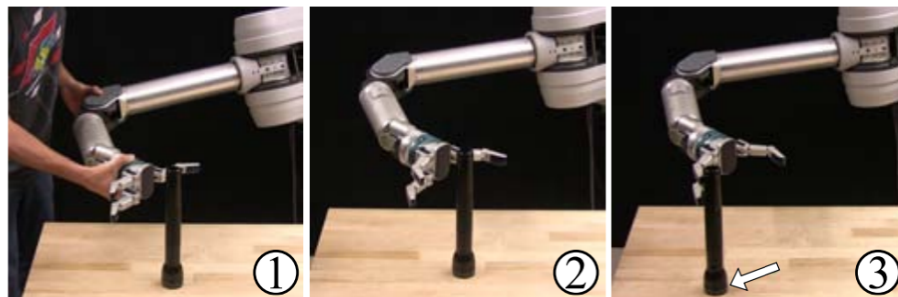
Visual servoing has also been applied to humanoid robots for grasping on the move. Mansard et al. [89] proposed a framework for building complex whole-body control and realized visually-guided grasping while walking on a humanoid robot. They divide the control into several sensor-based control tasks. This structure enables a very simple access for task sequencing and task-level control.

We also use visual sensing for guiding the gripper towards objects. The methods which discussed so far do not consider the uncertainty of a tracking result, so their methods can not react to the tracking uncertainty. Different from these methods, our architecture employs uncertainty of tracking to generate new grasps for adaptation. Therefore, our approach is able to generate the grasp which maximizes the success probability at each time cycle.

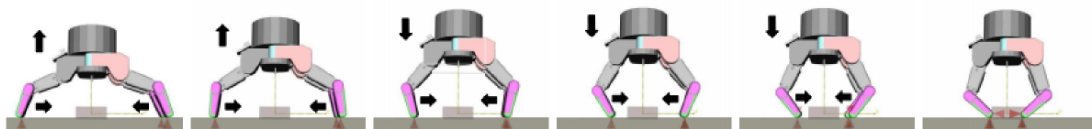
5.2.2 Methods using force sensing

As long as the robot touches an object, force sensing becomes a promising way for a robust grasp execution. Typically, force sensing refers to measuring joint torque or fingertip pressure. Kazemi et al. [63] proposed force compliant motion primitives to handle small objects. They exploited strain gauges on the joint of a gripper to detect collision with the support plane of the object. After collision is detected, fingers are closed using a control scheme that enables fingertips to slide on the support surface. Pastor et al. [105] presented an approach for generating contact reactive grasp motion using previous sensor experiences. These knowledge are acquired from human demonstrations. They exploit DMPs [57] to generate adaptive trajectories based on tactile and force torque sensing.

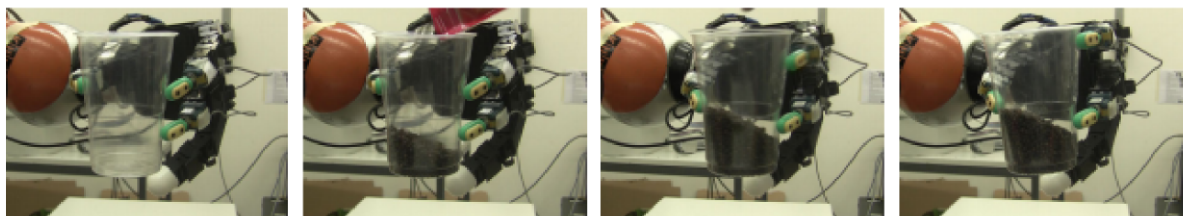
Tactile sensing is another commonly used strategies for establishing the final grasp ([19],[28],[75],[58]). Romano et al. [117] proposed a grasp controller based on tactile pressure and accelerometer to generate robotic tactile signals to mimic human SA-I, FA-I, and FA-II channels. These signals are used to indicate a set of event transitions between Close, Load, Lift and Hold actions of the gripper. The controller selects appropriate initial forces and increases or decreases as needed to prevent an object from slipping or determine when



(a)



(b)



(c)

Fig. 5.3: Methods using force sensing for grasp execution. (a) In [105], the robot is first demonstrated how to grasp an object using kinesthetic teaching. At test time, the robot detects object displacement with strain gauge mounted on the gripper and adapts its grasp motion to a new object position. (b) A force compliant motion primitives is proposed in [63]. A coordinated lift motion and close motion ensures the gripper always maintains contact with a support plane. This method works well for flat objects lying on a horizontal plane. (c) A method for in-hand grasp adaptation proposed in [79]. A three-finger initial grasp is first established on an object. The position of one finger can be changed online to maintain the stability of the grasp, when the weight of the object is increased.

to set it down. The proposed controller is implemented on a parallel gripper.

Tactile sensing can be also used for grasp adaptation with multi-finger grippers. Li et al. [79] proposed a grasp adaptation strategy for a multi-finger gripper. The method is able to handle the uncertainty of object's physical properties. A stability estimator was proposed to judge when to apply a grasp adaptation of a new grasp. The new grasp is chosen by the similarity of training examples.

In this chapter, force sensing is not considered as the primary sensing modality, because the stiffness of the target objects that we choose are extremely small. The tactile sensor is not sensitive enough to measure the grasping force. Instead we focus us in the reach-to-grasp stage, and use vision as our primary sensing modality.

5.2.3 Methods considering whole body coordination

In addition to the sensing modalities we reviewed so far, how to combine the sensing modality into a control architecture has also been addressed by many researchers. Here, we focus on the work for tasks requiring whole body coordination.

Back in the middle of the 90's, Khatib et al. [65] proposed a task-oriented control method for dynamic mobile manipulator coordination, in which a decentralized control structure was used to perform cooperative tasks with multiple mobile manipulators. Their focus is not on performing manipulation tasks like e.g. grasping.

Many work exploit arm-platform coordination in order to execute tasks such as 'opening doors', because a static robot can not perform such tasks due to the limited arm workspace. Ott et al. [103] used a mobile manipulator which has a 7-DOF arm and a omni-directional base to perform the door opening task. Based on impedance control, they generated the required arm-base coordinated motion for door opening, without knowing the door size and an explicit opening trajectory. During the operation, the robot arm kept the door at a certain distance while the base moved through the door. In [15], a framework for planning door opening trajectory with arm-base coordination was presented. They used a graph-based planning approach that allows a robot can open various doors both by pushing or pulling. Both of the work focused on how to use arm-base coordination to perform door opening trajectory. How to establish the grasp to the handle of the door is however not addressed.

Collision avoidance for mobile manipulators is also a common topic addressed by many authors. Omrcen et al. [102] proposed a method for real time obstacle avoidance using a torque and velocity controlled mobile manipulation robot. High compliance of the arm ensures safety without using any sensors. Dietrich et al. [23] proposed a reactive whole-body control mechanism for controlling a mobile manipulation robot with very high degree of freedom. They integrate several control requirements into a task hierarchy. These approaches demonstrate how to use arm-base coordinated motion to expand the robots' workspace.

Our control architecture not only covers many aspects which are also existed in the

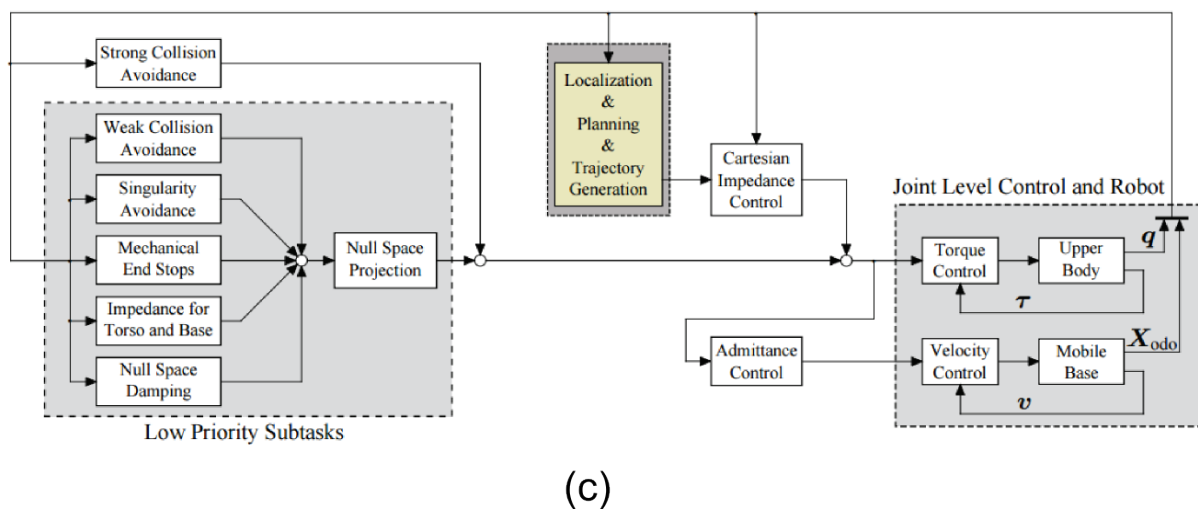
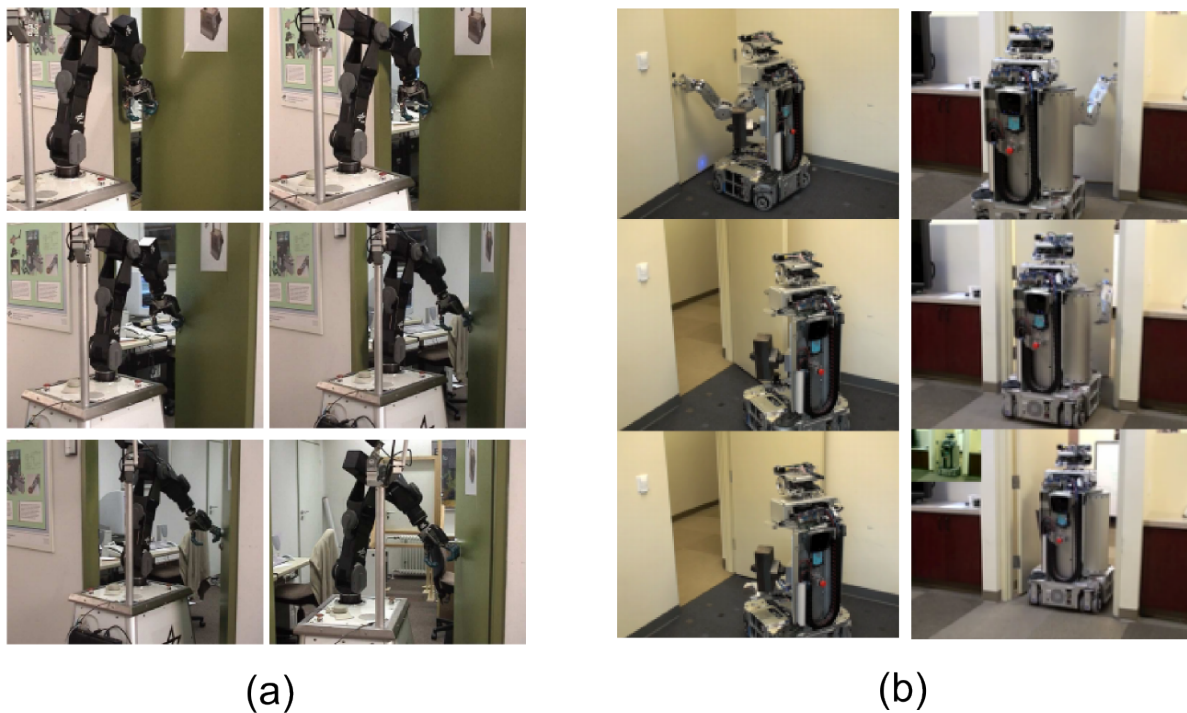


Fig. 5.4: Tasks that require tight coordination in between the motion of the base and the motion of the arm. (a) In [103], a mobile manipulator equipped with a single arm performs door opening task. Cartesian impedance controller was proposed to control the elastic joints of the arm. (b) A graph search-based motion planning method [15] is proposed to plan trajectories for door opening. The method is able to handle a variety of door types. (c) A reactive whole-body control mechanism [23] for a torque controlled robot. The control method can be used for performing tasks such as catching a flying ball [5].

previous work such as collision avoidance, coordination of actuators, but also seamlessly integrates vision feedback which may contain information of perception uncertainty. In addition, our architecture also monitors and controls switching of grasping phases to guarantee subgoals of grasping phases are subsequently reached for ensuring the final success.

5.3 Contributions

The main contribution of this chapter is the adaptive grasp control architecture. Different from the previous work which usually only address a sub-process of grasp execution. The control architecture monitors and controls the entire grasping process, from the phase that an object is detected to the phase the final grasp is established. The entire grasping process is decomposed into three phases. The robot has the freedom to choose a different combination of actuators to execute the motion in each phase. A tight coordination in between arm, base, and gripper imitates how a human performs grasping.

Different from previous work, vision feedback as well as the uncertainty of the feedback can be integrated into the system so that our system can adapt to uncertainty and external perturbation. An uncertainty-aware grasp configuration can be updated in each control cycle according to the vision feedback. We evaluate the complete system with a challenging grasping tasks. By comparing with two traditional approaches, our method achieved the highest grasping accuracy of the experiment.

5.4 System architecture

We propose a system architecture depicted in Fig. 5.5 for adaptive grasping control. The system architecture is designed to close the perception-action loop. The entire loop begins with the processing of sensor data. The component ‘Tracker’ estimates the state of an object based on the sensor data. The output of this component is a belief distribution of the object state. Then, the component ‘grasp synthesis’ takes the perception result and generates a set of grasp configurations. These include the afore-defined pre-grasp, pre-touch and force-closure poses. The component ‘Grasp phase control’ monitors the current state of the robot and generates goal configurations where the gripper of the robot must reach. The motion adaptation takes the TCP and the grasp posture configurations as input and generates trajectories for a gripper. The output of Motion adaptation consists a TCP trajectory and a grasp trajectory. The TCP trajectory defines the 6D path and velocity of the gripper, while the grasp trajectory defines the postures of the gripper. Since we represent the posture in a low dimensional space of a gripper, the actual finger joint to be executed has to be calculated by the ‘Gripper posture generation.’ To execute a TCP trajectory, the component ‘Redundancy resolution’ is used. It converts the velocity in task space to that in joint space. The output of the ‘Redundancy resolution’ and the ‘Gripper posture generation’ are both joint velocities. The component ‘whole body

coordinated control' takes the joint velocities as input and generate the corresponding hardware control signals to the robot.

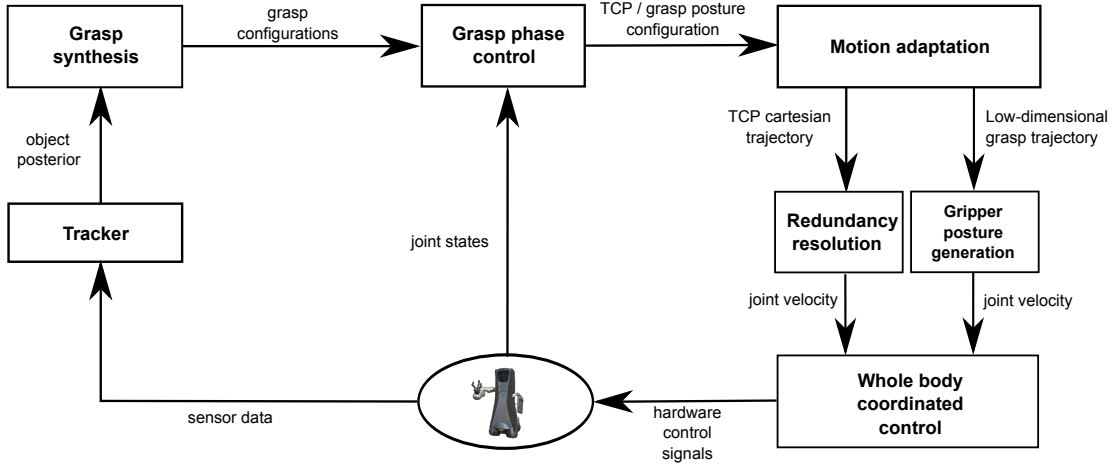


Fig. 5.5: System architecture

5.4.1 Motion adaptation

Adaptation of TCP Cartesian trajectory In this grasping scenario, the robot has to adapt its grasp to the measurement online. In order to achieve online motion adaptation, we choose dynamic movement primitives (DMPs) to generate the grasp motion. DMPs were originally proposed by Ijspeert et al. [57] and Pastor et al. [105] to learn trajectories from human demonstration. After learning, DMPs allow generalizing the learned path to new goal configurations. We represent the grasp motion using Cartesian trajectory. Two steps are required to generate the trajectory. The first step is learning the characteristic of a reference trajectory ¹. The second step is the adaptation from the reference trajectory. The reference trajectory is represented by a set of discrete points, which can be obtained using e.g. kinesthetic teaching [45]. As there is no human demonstration in our setup, we generate the reference trajectory using a cubic spine. The spine is generated by taking a current robot pose, a pre-grasp and a pre-touch configuration as input parameters. A DMP in one-dimensional (1-D) space is formulated by two differential equations which are equivalent to a non-linear mass-damper system:

$$\tau \dot{v} = k \cdot (g - x^{\text{dmp}}) - d \cdot v + (g - x_0^{\text{dmp}}) f(s) \quad (5.1)$$

$$\tau \dot{x}^{\text{dmp}} = v, \quad (5.2)$$

where x^{dmp} denotes the position of a 1-D DMP system. x_0^{dmp} is the start position and g is the position in which the DMP system converges. k and d denote stiffness and damping.

¹also called demonstration trajectory in the context of learning by demonstration

$f(s)$ is a non-linear function for approximating arbitrary convergent characteristics. $f(s)$ can be realized by a linear combination of a set of basis functions. s is a phase variable with $s \in [0, 1]$ and is determined by the following canonical system

$$\tau \dot{s} = -\alpha s. \tag{5.3}$$

τ is a time-scaling parameter used to scale the length of the trajectory to be generated. The process of learning a reference trajectory include: (1) Compute the velocity of the reference trajectory for each time stamp. (2) Compute the phase parameter s of the i -th point by

$$s_i = e^{-\alpha/\tau \cdot t} \tag{5.4}$$

(3) Compute the target value $f_{\text{target}}(s_i)$ for each point on the reference trajectory by plugging the points and velocities into Equ. 5.1 and Equ.5.2. (4) Use linear regression to estimate the parameters of $f(s)$.

For learning the position of a reference TCP trajectory, three instances of a 1-D DMP system are required to represent each dimension in a cartesian space. However, the orientation of the reference TCP trajectory can not be directly represented by three 1-D DMP system if we choose Euler angles to represent the orientation, as the Euler angles contain singularity at 2π . Instead, we use quaternion to represent the orientation so that a single quaternion-based DMP system proposed in [105] can be used to learn the orientation characteristics of the TCP reference trajectory.

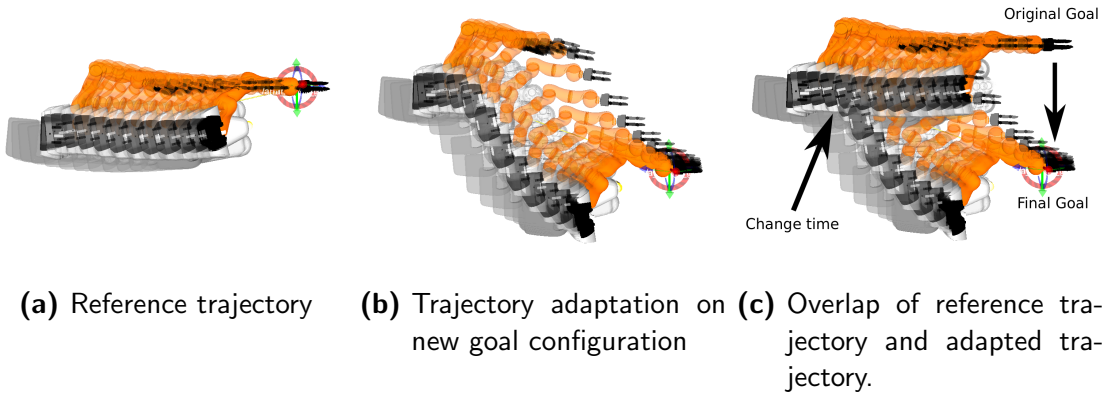


Fig. 5.6: An example of trajectory adaptation using DMPs

Generate a new trajectory from a learned DMP system include following steps: (1) Calculate the phase parameter s using Equ. 5.4 for a given time stamp t . (2) Evaluate the function approximator $f(s)$ and compute \dot{v} and \dot{x}^{dmp} . (3) Compute a new trajectory point by numeric integration with

$$x^{\text{dmp}}(t + \delta t) = x^{\text{dmp}}(t) + \dot{x}^{\text{dmp}} \cdot \delta t \tag{5.5}$$

$$v(t + \delta t) = v(t) + \dot{v} \cdot \delta t \tag{5.6}$$

where δt is a very small time duration. We choose δt to be 1 ms to reach a sufficient accuracy. To generate the whole trajectory, Step 1 to 4 is then repeated until $g - x^{\text{dmp}}$ is under a threshold. At an arbitrary time stamp, if a new goal configuration of TCP is available, we can plug it into Equ. 5.1 to force the DMP system converge to the new configuration. In this way, the grasp motion to be executed is adapted to new sensor measurements. Fig. 5.6 explains the adaptation of cartesian trajectory on a new goal configuration.

Adaptation of low-dimensional grasp trajectory Similar to adaptation of a cartesian trajectory in the ‘grasp-motion’ phase, we also employ DMPs to generate trajectories for the ‘force closure’ phase. An example force closure trajectory is shown in Fig. 5.7. The force closure trajectory contains a closing movement of fingers and a lifting movement of the gripper. In this way, the gripper realizes the same closing behavior of a parallel gripper to ensure the desired contact region of the finger moves anti-parallel towards each other.

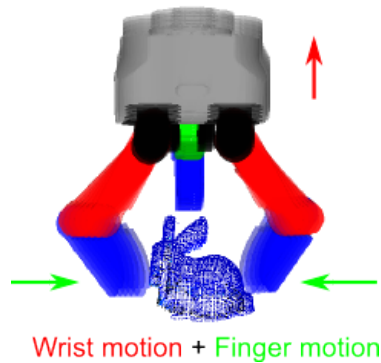


Fig. 5.7: A reference trajectory for learning coordinated force closure motion.

We realize the grasp trajectory with two DMP subsystems which are synchronized using the same DMP phase parameter s . One DMP subsystem generates the desired lifting trajectory while the other DMP subsystem generates a low-dimensional finger trajectory of the gripper. In section 4.6.1, we propose a method for parameterizing a high DOF gripper by using the ‘Eigengrasp’ concept. The method reduces the total DOF of a gripper to the number of joint groups that we defined for that gripper. For the gripper we use in this work, we need totally four one-dimensional DMPs to represent the closing trajectory, because four joint groups are defined for the Schunk gripper. The entire joint movement of the gripper is calculated according to Equ. 4.18 by the module ‘Gripper posture generation.’

5.4.2 Arm platform redundancy resolution

In order to control motion in cartesian space, the Cartesian trajectory generated by DMPs must be converted to the motion controls in joint space. For robots which have more than six DOF, the joint redundancy can be exploited. We apply the idea of nullspace optimization [97] to resolve the redundancy. Let \underline{x}_d be the desired position and orientation

of tool center point (TCP), and let \underline{q}_d denote the desired joint positions. The desired joint velocity $\underline{\dot{q}}_d$ can be expressed by

$$\underline{\dot{q}}_d = \underline{J}^+ \underline{\dot{x}}_d + (\underline{I} - \underline{J}^+ \underline{J}) \underline{\xi}, \quad (5.7)$$

where \underline{J}^+ represents the pseudo-inverse of Jacobian \underline{J} that is defined by $\underline{J}^+ = \underline{J}^T (\underline{J} \underline{J}^T)^{-1}$. The term $(\underline{I} - \underline{J}^+ \underline{J})$ projects an arbitrary vector $\underline{\xi}$ onto the nullspace of Jacobian. In this work, we exploit three criteria to resolve redundancy, which are generally applicable to any kind of mobile manipulation robots. These criteria are self collision avoidance, joint limit avoidance and singularity avoidance. We define each criterion as a unique cost function of joint position $E^i(\underline{Q})$. By setting $\underline{\xi} = -\nabla(\sum_{i=1}^3 E^i(\underline{Q}))$, nullspace velocity reduces the total cost in the gradient direction.

Self collision avoidance This criterion ensures that a robot does not collide with itself. The gradient of cost function for self collision avoidance is defined by

$$\nabla E^{(1)}(\underline{Q}) = \sum_{(i,j) \in CP} \frac{dE^{(1)}}{dD_{(i,j)}} \cdot \frac{dD_{(i,j)}}{d\underline{Q}}, \quad (5.8)$$

where $D_{(i,j)}$ denotes the distance between two links, which can potentially collide with each other. We compute link distances by means of the approach proposed in [76]. The gradients of link distances with respect to joint positions are computed numerically. $\frac{dE}{dD_{(i,j)}}$ denotes the gradient of cost with respect to a link distance.

Joint limit avoidance This criterion ensures that the limits of the joints will not be overshoot. Each joint of arm must work in a specified joint range. Overshooting of joint limit can cause hardware defects. The gradient of joint limit avoidance is expressed by

$$\nabla E^{(2)}(\underline{Q}) = \left[\frac{dE^{(2)}}{dQ_1}, \dots, \frac{dE^{(2)}}{dQ_N} \right]^T. \quad (5.9)$$

The cost function of each joint $E(Q_i)$ is defined as follows

$$E^{(2)}(Q_i) = \alpha_i \cdot [Q_i - (Q_i^{\max} - Q_i^{\text{soft}})]^2, \quad (5.10)$$

where α_i is a parameter to scale cost function. Q_i^{\max} denotes the maximum limit that joint i can reach. Q_i^{soft} is the range in which cost is activated.

Singular configuration avoidance This criterion ensures that no singular configurations will occur during the whole grasping procedure. Near singular configuration small actuator torques will lead to a large end-effector wrench. This could cause jerk movement and

overshoot hardware limitation. An approach to avoid singular configurations is to maximize manipulability defined by $M(Q) = \sqrt{\det(\underline{J} \cdot \underline{J}^T)}$. Similar to the first cost function (Equ. 5.8), the gradient of cost for singularity avoidance is given by

$$\nabla E^{(3)}(\underline{Q}) = \frac{dE^{(3)}}{dM} \cdot \frac{dM}{d\underline{Q}}. \quad (5.11)$$

5.4.3 Whole-body coordinated control

A mobile manipulator is composed of individual hardware components. Each one has its interface for control. The purpose of this module is synchronization of devices for executing trajectories containing high DoF. The structure of whole body coordinated control is depicted in Fig. 5.8. The input of ‘Whole body coordinated control’ is ‘Redundancy resolution’, ‘Joint space motion planner’, and ‘Gripper posture generation’. One use case is to control an arm and a platform simultaneously, we can use either the ‘Joint space motion planner’ or the ‘Redundancy resolution’ to generate joint velocities. As we did in chapter 3, for all collision-free motions, we use joint space motion planner to move the robot. In this chapter, we concentrate in grasp-motion phase. Typically, the robot approaches a pre-touch configuration of an object linearly, in this case, we use ‘Redundancy resolution’ to convert a linear Cartesian trajectory to joint velocities. Another use case is to control an arm and a gripper simultaneously, we need ‘Gripper posture generation’ to convert a low-dimensional gripper trajectory to the trajectory containing the entire joint positions using Equ. 4.18. We can generate a Cartesian trajectory of a gripper and a posture trajectory of each gripper joint to realize a coordinated grasping trajectory during force closure. The coordinated grasping trajectory is specially designed for grasping from the top of a flat object, so that during force closure phase a contact force is always maintained between the fingers and a support surface.

The ‘Whole body coordinated control’ contains two layers. The first layer is command allocation. The command allocation first splits the output of a high DoF trajectory and then it assigns joint velocities of the trajectory to each component. The second layer contains a set of low-level controllers. The low-level controllers convert the joint velocities to the commands accepted by the devices. For example, in order to control a base platform, the base controller must convert the joint velocities to linear and rotational velocities using Equ. 3.12. To control an arm, ‘arm controller’ must first integrate the velocities to positions and then send the positions to the hardware. To control a gripper, a ‘Gripper controller’ is proposed to send the desired velocities to each joint and stop the joints if the pressure measured by tactile sensors is higher than a pre-defined threshold.

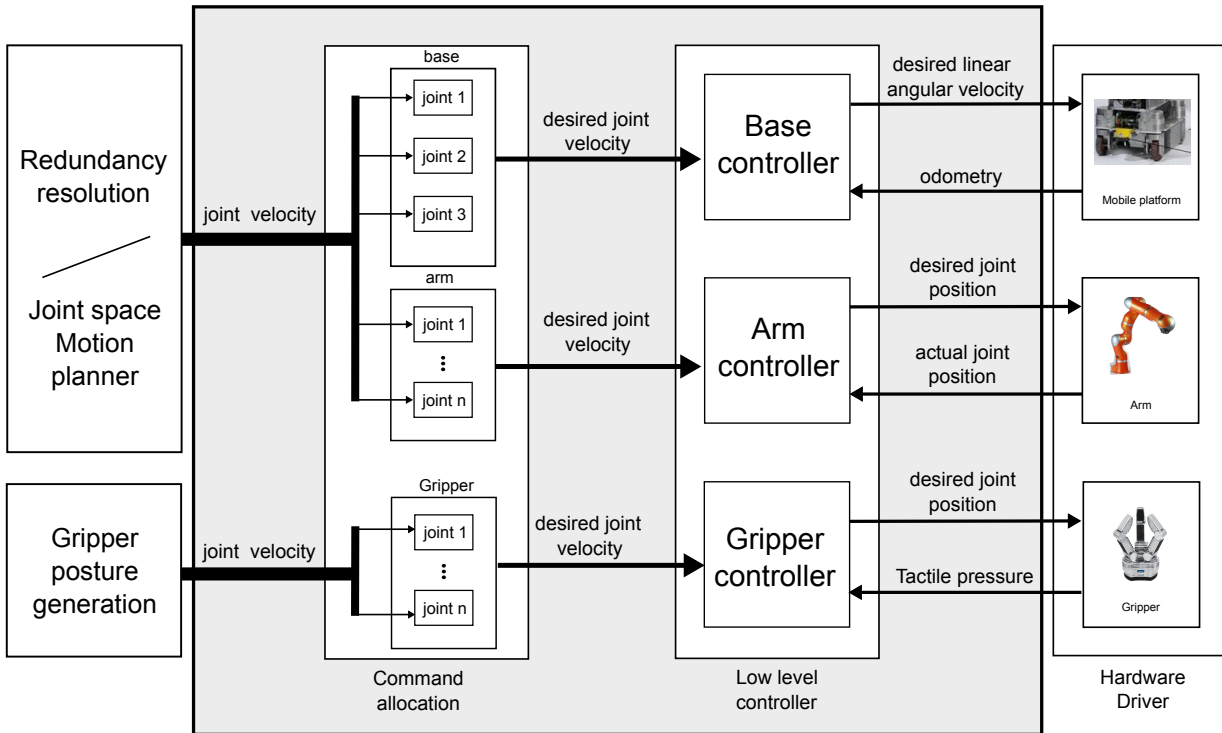


Fig. 5.8: Whole body control architecture

5.5 Use case: grasping cylindrical objects with unknown dimension

In this section, we choose another class of grasping problems to demonstrate the proposed system. How can a robot be able to conduct robust grasping, even the sensor used for perception has large noise. In chapter 2.4.3, we proposed a paradigm to deal with this situation. The main idea of this paradigm is to integrate the sensor measurement online and adapt control trajectories in an iterative fashion. The proposed system architecture provides the foundation to realize the paradigm. In order to demonstrate the proposed system architecture in this grasping scenario, we choose objects, the shape of which can be approximated by a cylinder, as target objects. We also assume that the target objects stand upright on a support plane. This assumption simulates the most stable orientation of the cylindrical objects such as mugs, glasses and etc in the real world. Based on this assumption, the grasp can be easily parameterized to satisfy a task constraint. For example, if the robot is desired to perform a pouring task, the grasp configuration has to be parameterized so that the robot approaches and grasps the object from the side. In the following, we elaborate two necessary components to use the proposed system architecture, the first is tracking of object state and the second is grasp synthesis for cylindrical objects.

5.5.1 Measurement model for object state estimation

The shape of a cylindrical object can be represented by two parameters: radius and height. Since objects are placed in upright positions in this grasping scenario, we define additional two parameters to represent the position of the object. Since we require the robot to grasp from the side of an object, the height of the gripper can be chosen at the half height of the object. Thus, the height parameter is not the crucial parameter which affects grasp success. In contrast, the radius and the position of an object are the critical parameters which must be estimated accurately. Therefore, we only need to online estimate the radius and the position in this grasping scenario. We propose to use an extended Kalman Filter (EKF) for this task. The EKF is a method for state estimation. In order to use the EKF, a system model and a measurement model must be provided. In our case, the system model defines the distance covered by a gripper during a grasping process. The distance traveled by the gripper can be obtained from the odometry and the joint positions of a robot's arm. The measurement model calculates desired measurements given the true state of a system. In the following we elaborate how the measurement model is defined for the given grasping scenario.

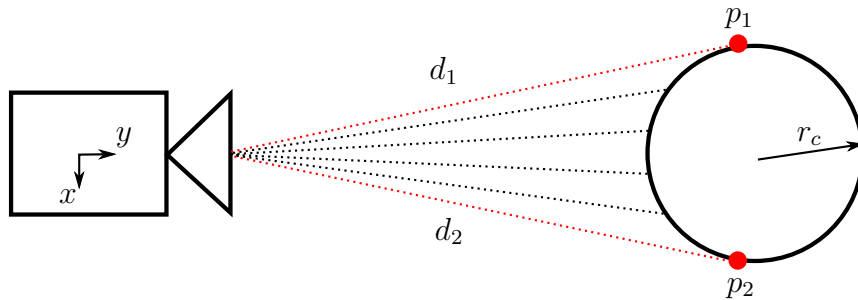


Fig. 5.9: A circular object measured by a sensor. p_1 and p_2 are the measurements used in EKF for estimating the position and the radius of the object.

Without constraining the actual sensor we use for the grasping task, we assume the input data of the sensor can be converted to a data type of a laser scan. A laser scan is defined by a set of points in the polar coordinate system $(\underline{d}, \underline{\theta})$, where \underline{d} is an array which gives a set of range measurements. $\underline{\theta}$ is also an array which has the same size as \underline{d} . Each θ_i in $\underline{\theta}$ defines the bearing angle for the corresponding range measurement d_i . If the real sensor outputs a stream of point clouds, we can extract the 3D coordinate of the points on the plane that we interested and then convert them to the polar coordinates. In order to estimate the radius and the position of an object, we only need two critical points as a measurement. These two points hit two tangential lines of a circular object as shown in Fig. 5.9. We define these two points as the measurement to be used in an EKF. By

considering the triangular relationship, the desired measurement can be formulated by

$$\underline{z}_{\text{EKF}} = \begin{bmatrix} \underline{z}_{p_1} \\ \underline{z}_{p_2} \end{bmatrix} \quad (5.12)$$

$$= \begin{bmatrix} d_1 \\ \theta_1 \\ d_2 \\ \theta_2 \end{bmatrix} \quad (5.13)$$

$$= \begin{bmatrix} \sqrt{x_c^2 + y_c^2 - r_c^2} \\ \text{atan2}(y_c, x_c) + \text{asin}\left(\frac{r_c}{\sqrt{x_c^2 + y_c^2}}\right) \\ \sqrt{x_c^2 + y_c^2 - r_c^2} \\ \text{atan2}(y_c, x_c) - \text{asin}\left(\frac{r_c}{\sqrt{x_c^2 + y_c^2}}\right) \end{bmatrix}, \quad (5.14)$$

where (d_1, θ_1) and (d_2, θ_2) are the desired measurement in polar coordinate of point p_1 and point p_2 . $\underline{x}_o = (x_c, y_c, r_c)$ denotes the position and the radius of the object in Cartesian coordinate of the sensor.

5.5.2 Grasp synthesis for circular objects

We use the same grasp success model proposed in 4.4.3 to evaluate the success probability of a grasp. The object is represented by its radius and position $\in \mathcal{R}^2$. A parallel gripper is chosen to grasp the object. The gripper configuration $\mathcal{G} \in \mathcal{R}^4$ contains the 2D positions of the left finger and the right finger. In order to validate the conditional grasp success model, we generate many random gripper configurations. For each configuration, we evaluate the conditional grasp success probability. Fig. 5.10(a) depicts 10 random gripper configurations, the grasp success probability of which are higher than 90 %. Fig. 5.10(b) shows 10 negative examples, the conditional grasp success probability of which are lower than 10 %.

The goal of grasp synthesis is to find an optimal pre-touch configuration. The optimal pre-touch configuration is found according to Equ. 4.2 by searching the maximal marginal grasp probability. Since the dimension of object state for a circular object is only three, we can integrate the Equ. 5.15 numerically. The posterior of the object is obtained from the EKF estimation. It is represented by a multi-variate Gaussian $p(\underline{x}_o | \mathcal{Z}) = \mathcal{N}(\mu, \Sigma)$ where μ is a mean and Σ is a covariance matrix. We integrate over the $\pm 3\sigma$ interval to approximate the integral of Equ. 5.15 by

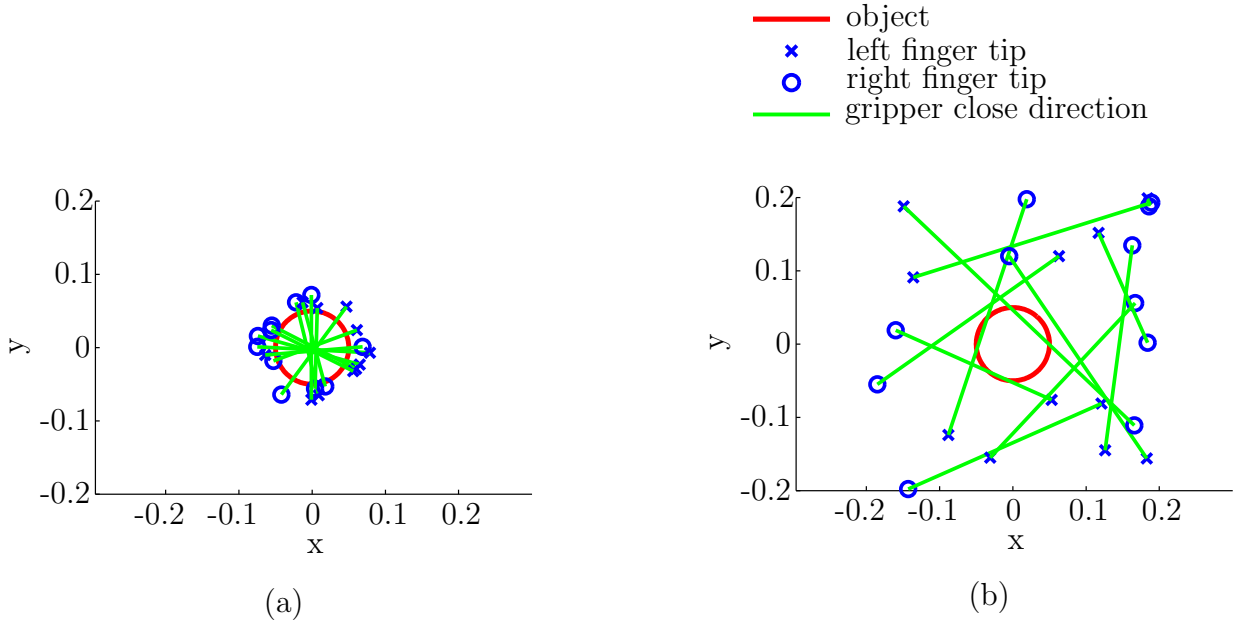


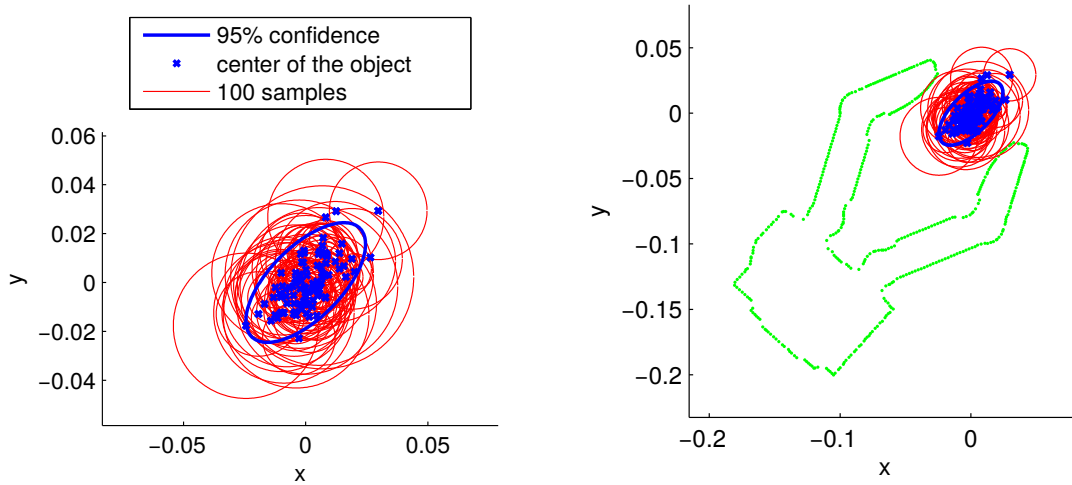
Fig. 5.10: (a) Ten random sampled good pre-touch configurations, where the grasp likelihood larger than 0.9, (b) Ten random sampled bad pre-touch configurations, where the grasp likelihood smaller than 0.1.

$$\begin{aligned}
 P(S|\mathcal{G}, \mathcal{Z}) &= \int_{\underline{x}_o} P(S|\underline{x}_o, \mathcal{G}) \cdot p(\underline{x}_o|\mathcal{Z}) d\underline{x}_o \\
 &= \int_{\underline{x}_o} P(S|\underline{x}_o, \mathcal{G}) \cdot \mathcal{N}(\mu, \Sigma) d\underline{x}_o \\
 &\approx \int_{\mu_1-\sigma_1}^{\mu_1+\sigma_1} \int_{\mu_2-\sigma_2}^{\mu_2+\sigma_2} \int_{\mu_3-\sigma_3}^{\mu_3+\sigma_3} P(S|\underline{x}_o, \mathcal{G}) \cdot \mathcal{N}(\mu, \Sigma) dx_c dy_c dr_c,
 \end{aligned} \tag{5.15}$$

where x_1, x_2, x_3 represent the radius and the positions, μ_1, μ_2, μ_3 are the means of positions and the radius. $\sigma_1, \sigma_2, \sigma_3$ are the diagonal elements of the covariance matrix Σ . In this case, we use Subplex [119] implemented in NLOpt [60] to find the maximal probability of $P(S|\mathcal{G}, \mathcal{Z})$. Fig. 5.11a visualizes a synthetic posterior distribution with $\mu = (0, 0, 0.01)$

and $\Sigma = \begin{pmatrix} 0.01 & 0.008 & 0 \\ 0.008 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{pmatrix}$. The optimal pre-touch configuration for this posterior is

depicted in Fig. 5.11b. The marginal success probability of this object posterior by taking the optimal pre-touch configuration is equal to 44.9%. The absolute mass of probability depends on how the parameters are chosen for evaluating the conditional success grasp probability. Since the parameters we choose for calculating the conditional success grasp probability for this example is conservative, the success probability by taking the optimal pre-touch configuration is not high. However, the pre-touch configurations computed using our model are very close to the optimal in reality. From the figure we can see, that the



(a) A synthetic object posterior $p(\underline{x}_o|\mathcal{Z}) = \mathcal{N}(\underline{\mu}, \Sigma)$ where $\underline{\mu} = (0.01, 0, 0)$ and Σ is a covariance matrix with $\sigma_x = \sigma_y = \sigma_r = 0.01$, $\sigma_{xy} = 0.008$ and $\sigma_{rx} = \sigma_{ry} = 0$

(b) The optimal pre-touch configuration computed for a circular object with the posterior shown to the left. The marginal success probability is $P(S) = 0.449$.

Fig. 5.11: An optimal pre-touch configuration computed for a given multivariate Gaussian posterior.

grasp is explicitly computed for the given object state uncertainty.

5.5.3 Experimental evaluation

To evaluate the proposed system architecture, we performed 900 individual grasping experiments on a set of test objects. The experiments are conducted with a mobile manipulation robot. The robot's actuation system is composed of a 7 DOF KUKA LBR 4 and a 7 DOF SCHUNK SDH2 gripper mounted on top of an omnidirectional mobile base. The primary sensors of the robot are a head-mounted Kinect RGB-D camera and a hand-mounted laser time-of-flight camera (Creative Sens3D). We use the head-mounted sensor to generate the initial estimate of all objects in the scene and the hand-mounted camera for continuously tracking the target object during the grasping process.

Experimental setup

The robot is initially placed at a distance of about 1.5m from the table carrying the objects (see. Fig. 5.12). At this distance, the entire table-top is in the field of view of the robot's head camera. Our purposefully delicate test objects are rolls of paper of three different radii. These test objects are very light and are easily tipped over in the case of inaccurate grasp attempts.

We randomly select 4 test objects in each run of the experiment and place them upright on the table. The robot grasps the objects one by one, lifts them to a height of 10 cm above

table and then puts them back. A grasp is considered successful if the robot manages to successfully put the test object back, so that it stands upright on the table. The latter condition enforces that the object was accurately grasped and simulates a simple manipulation operation. It follows our conviction that grasping only makes sense in a manipulation context.

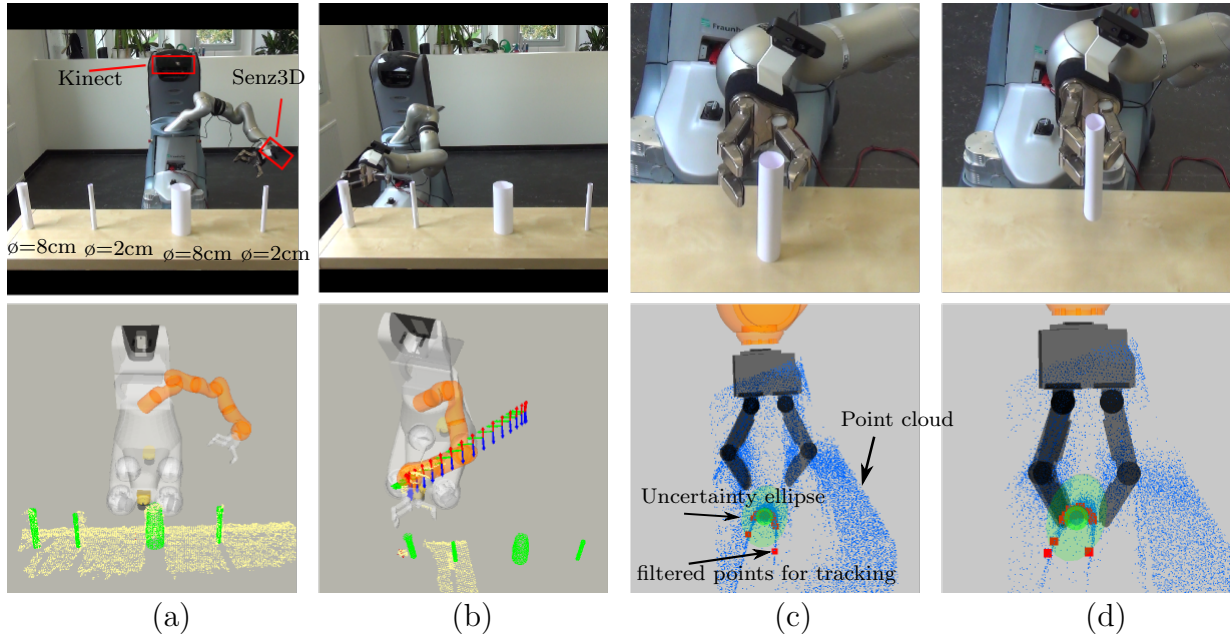


Fig. 5.12: (a) The robot generates an initial estimate of the observed objects and their configuration parameters with the head-mounted 3D camera. This initial estimate is used to determine a pre-grasp configuration only. (b) The robot moves to the pre-grasp configuration. (c) The robot approaches the object and uses the hand-camera to continuously re-estimate object pose and radius. (d) Object is lifted up after being successfully grasped.

Results

Fig. 5.13 shows the trajectory in the entire grasping process. From time 0–25s, the robot is in the pre-grasp phase in which the robot approaches the pre-grasp configuration. Approximately at time 25s, the pre-grasp configuration is reached and grasp motion phase begins. From then on, the *grasp phase control* sends updated TCP goal configurations to the *Motion adaptation* and switches from arm-platform motion to arm alone motion.

Fig. 5.14 compares the originally generated reference trajectory with the actual online trajectory for a successful grasp. New TCP goals are indicated by blue circles. Two clusters of blue circles indicate goal adaptations during the entire grasping process. Comparing the distribution of blue circles, the first cluster of circles has a bigger deviation than the second cluster, because the tracking uncertainty decreases with the distance between hand-mounted 3d camera and the target object. Besides that, the motion control errors of the mobile platform during the first phase also contribute to the deviation of the first cluster.

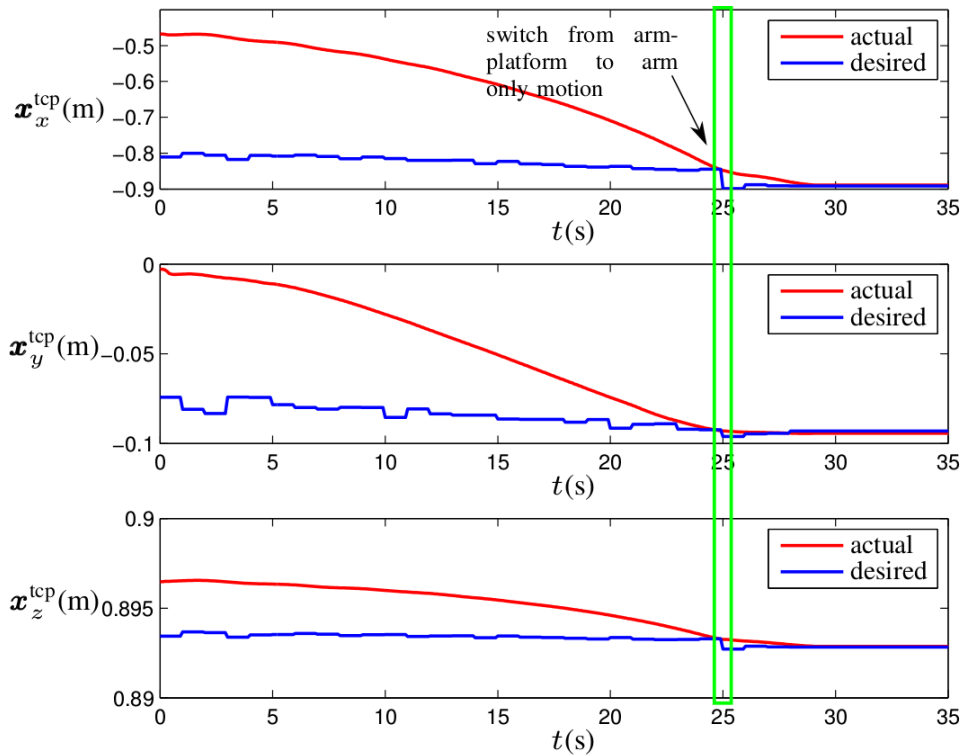


Fig. 5.13: Blue line is the desired TCP goal configuration. Red line is the actual trajectory of the TCP. At the 25s, the gripper reaches pre-grasp configuration. Grasp motion phase begins at 25s. In this phase the robot reactively approaches pre-touch configuration based on new sensor measurements.

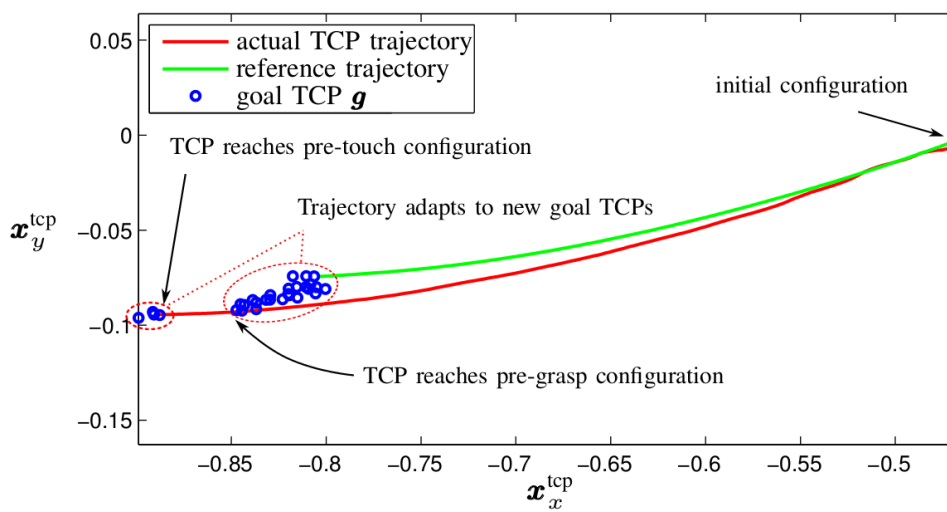


Fig. 5.14: Illustration of the trajectory of a successful grasp

We compare our approach with two other traditional approaches. Since our approach both considers uncertainty and grasp adaptation, we regard our approach as ‘closed-loop uncertainty-aware’. Both other approaches execute grasping in an open-loop fashion. The entire grasping trajectory is pre-computed at the pre-grasp configuration. One approach uses the same uncertainty-aware method proposed to compute the pre-touch configuration, while the other method compute the pre-touch configuration by sampling a random approach direction. The experiment is repeated 100 times for each object with different radius. In total, 900 grasp outcomes are recorded. Fig. 5.15 shows the experimental results. Our closed-loop uncertainty-aware approach achieves an average success rate of over 90 % regardless of actual object size, while the success rate of other two approaches are dependent on the size. For smaller objects ($\phi = 2\text{cm}$, $\phi = 4\text{cm}$) they often missed the objects or hit the objects while approaching the pre-touch configuration. Comparing both open-loop approaches, explicitly considering the uncertainty also contributes to the overall grasping success, because the uncertainty-aware approach generates the pre-touch configuration with higher marginal success probability. In summary, both closed-loop control and uncertainty-awareness are key factors to achieve a robust grasping performance.

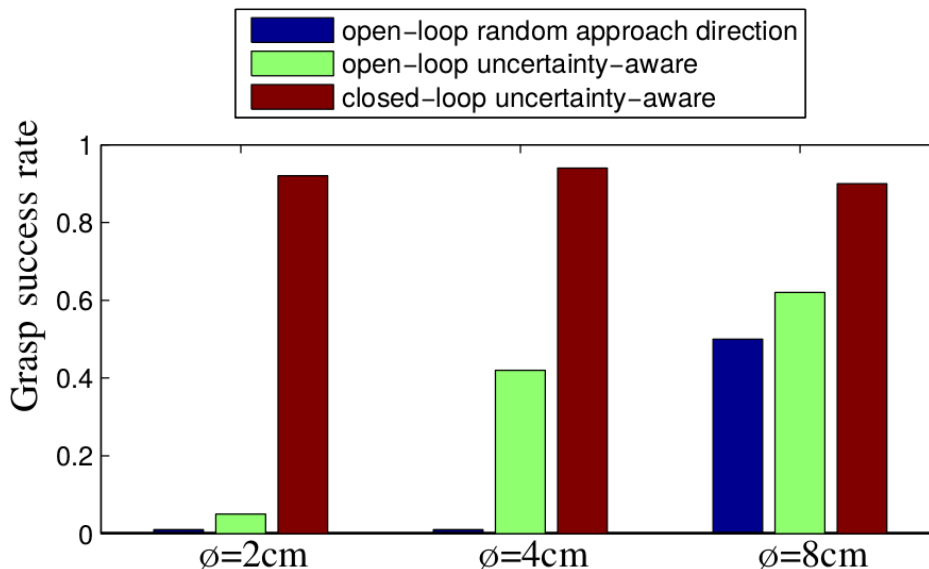
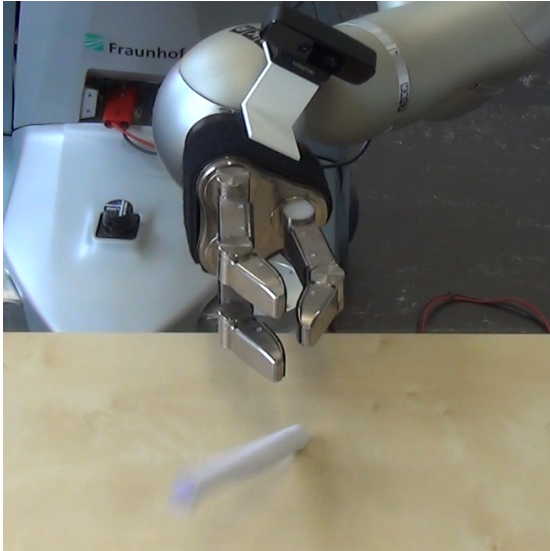


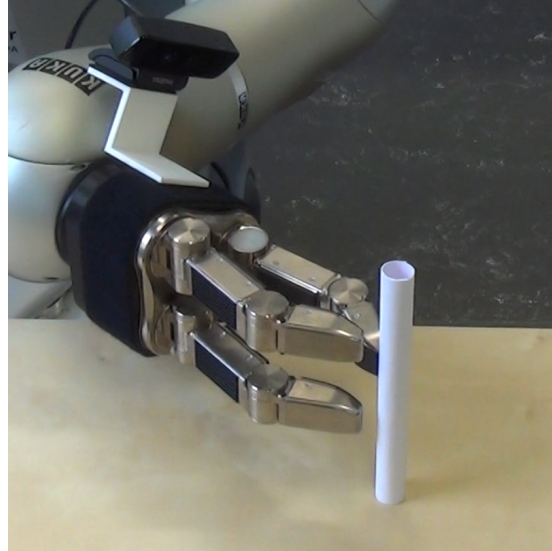
Fig. 5.15: Comparison of grasp success rate between methods. In total, 900 grasp trials were performed to obtain the data.

5.6 Summary

In this chapter, we address the problem of how to make grasp execution more robust. For this purpose, we propose an adaptive whole body control architecture specific tailored for mobile manipulators. Our control architecture contains seven major modules which close the entire loop from perception to control. We specifically focus on the reach-to-grasp



(a) This figure shows a typical grasping failure using the open-loop uncertainty-aware approach. The object is tipped over during the force-closure phase because the desired pre-touch configuration is not actually reached during the grasp motion phase. Uncertain perception and execution lead to this failure.



(b) This figure shows another grasping failure when using the open-loop random approach direction method. The grasp fails, because the fingers were closed at the wrong place. Randomly chosen approaching direction occasionally generates a longer trajectory which introduces uncertainty in actuation.

phase, in which the gripper of a robot moves from a ‘pre-grasp’ configuration towards the final grasp configuration. Different from the traditional methods which execute grasps in open loop, our method allows vision feedback to be continuously integrated. We exploit dynamic movement primitives for online trajectory adaptation. In this way, the robot continuously reduces the uncertainty during the grasping process. It allows a robust execution even the perceptual uncertainty could be large at the beginning of the process.

Another advantage of our architecture is the flexibility of coordinating different actuators to perform a certain motion. Our robot imitates how humans coordinate their locomotion, torso, arm and hand to grasp objects. We exploit arm-platform coordination and arm-gripper coordination for mobile manipulators. The arm-platform keeps the gripper in a ‘good’ configuration that the manipulability of the arm is maximized. In the case of grasping objects from the top, the arm-gripper coordination can keep the fingers of a gripper in contact with a support surface to increase the chance of success.

To transfer the proposed architecture to other robotic systems and tasks, the overall architecture remains the same, but the corresponding modules must be configured and adapted. For robots with a different kinematic of arm and mobile base, the module ‘Redundancy resolution’ and ‘Whole-body coordinated control’ need to be reconfigured so that the gripper of the robot maintains a good manipulability during the pre-grasp phase. For robots with a high DOF gripper, the ‘eigen-grasp’ space of the gripper must be defined. In addition, the ‘Gripper posture generation’ module must be adapted. For robots

equipped with another type of sensor, the accuracy and the measurement model must be specified for the ‘Object tracker’. For grasping other types of objects, the ‘Grasp synthesis’ module must be extended to handle other object type and perceptual uncertainty.

We evaluate the proposed architecture in a challenging grasp scenario. The object that we choose for the experiment is difficult to grasp, because of its weight and place orientation. The sensor we used to perform this task contains large noise which even more increase the difficulty. We combine the uncertainty-aware grasp synthesis which we proposed in the earlier chapter and demonstrate that considering uncertainty as well as closed-loop grasp execution significantly increase the grasp success.

6 Conclusion and Future Work

6.1 Concluding remarks

In this dissertation, an integrated and general system is developed to cover various aspects in robotic grasping. The system allows a mobile manipulator to perform robust grasping under different levels of prior knowledge, task requirements and sensing capabilities. In order to bring various grasping relevant aspects under the same framework, a probabilistic graphical model that based on the Bayesian network is proposed to formulate the grasping problem. This model contains a set of grasping relevant variables and represent the joint distribution of these variables. The advantage of using the proposed model is that it simplifies the modeling of complex joint distribution over many variables by small conditional probability distributions which are easy to design. Thus, the only difference between different grasping scenarios exist in that some variables are observable according to a task and prior knowledge. The goal of different grasping problems remains same: selecting the best action to maximize the probability of success. The problem of three challenging grasping scenarios are formulated using the model and explored in chapter 3,4 and 5 respectively.

In chapter 3, the problem of assembly is addressed. In this scenario, the robot is required to grasp known objects in a task oriented manner so that the assembly can be conducted. In this scenario, a skill framework is proposed to tackle the challenge of grasping under task constraint. Within the skill framework, two high- level skills are proposed to handle the complete sequence from the object perception, the reasoning of task oriented grasp to the coordinating of actuators for grasp execution. The reusable high level skills provide an intuitive way of parameterizing an assembly task. In addition, both intermediate base positions and arm base coordinated motions are planned during the execution, which contributes to the successful and efficient accomplishment of the task.

In chapter 4, we focus on the synthesis aspect of grasping. The robot is required to perceive and generate a suitable grasp configuration for an object which is presented to it for the first time. A new uncertainty-aware surface representation, as well as a sensor fusion method tailored for the representation are proposed for the object perception. This method allows fusing data from multiple sensors in real time. As a result, this reduces the measurement uncertainty and provides a detailed model and efficient surface query for grasp synthesis. To generate a feasible grasp configuration, an object representation independent model is proposed to predict the likelihood of grasp success for a given a grasp configuration. The final grasp is selected by a simulated annealing method which finds the most promising and precision grasp contact points.

In chapter 5, the control aspect of grasping is studied. In this scenario, the robot is required to grasp very light objects, however, the dimension of them are unknown. In addition, the sensor provided in this scenario has a limited sensing capability and produces very noisy data. An adaptive grasp control architecture is proposed to tackle this challenging scenario. The control architecture allows a robot to flexibly switch the combination of various actuators to mimic how a human performs grasping. In addition, the perceptual result can be fed back to grasp synthesis so that grasp can be executed in closed-loop.

In each scenario, a massive number of grasping experiments using real robot is performed. The results are compared to state-of-the-art methods, which proves the effectiveness of the proposed system.

6.2 Future directions

The work presented in this dissertation provide a solid contribution to bridging the gap between human and robotic grasping. The ideas demonstrated so far also motivate several future research directions:

Learning from grasp experience: As the computational power of processors increases and the cost of building robots reduces, nowadays it is feasible to perform large scale real world grasp experiments and train models with a large number of parameters. Therefore, more work should be done to exploit this advantage so that a robot can learn both from own experience and experience of other robots.

Exploit multi-modal information: in this work, we mainly use the depth information of sensors to determine the state of an object. For grasping, color information and tactile information are also important information source not only in predicting grasp success but also can be used in grasp control. How to combine multi- modal sensory input in a systematic way for a robust grasping control is an interesting research direction.

Handle environment constraints: A table-top environment is chosen for the grasping scenarios presented in this work. Most objects can be retrieved by grasping directly. However, for tasks like retrieving objects from narrow space, the current approach can not handle this task. Humans usually exploit the environment as a help rather than as obstacles. They use more sophisticated manipulation skills such as pushing, sliding to reconfigure the object prior to grasping. This requires a well understanding of environment dynamics and a compliant control mechanism. How to represent and retrieve the knowledge and combine them into a compliant robot controller can be an interesting direction.

List of Figures

1.1	Thesis outline	4
1.2	(a) A high-level assembly sequence provided to the system. P.R: The skill for picking and placing an object in another orientation. P.I: The skill for picking an object and inserting in another one. (b) A mobile manipulation robot performs the assembly task autonomously.	6
1.3	(a) The object is presented to the robot for the first time. (b) The robot reconstructs the object surface by fusing sensor measurement from two depth cameras. (c) An uncertainty-aware object model is constructed and a feasible grasp is found for grasping the object. (d) Successful execution of the grasp on a real robot.	7
1.4	(a) The robot is about to grasp light objects. The radius and position of the objects are estimated online during the reach-to-grasp phase. (b) The robot successfully pick up the object despite of large sensing uncertainty.	8
2.1	A Bayesian network for the grasping problem. Variables surrounded by a box are binary-valued discrete random variables. Variables surrounded by an circle can be either continuous or discrete.	11
2.2	Encapsulation of the original graphical model.	17
2.3	Chaining of the original graphical models.	17
3.1	Assembly of a 3D printed toy radio.	23
3.2	A grasp synthesis method based on bounding box representation proposed by [56]. Objects are first decomposed into minimum volume bounding box. The parameter of the bounding box as well as grasps generated by a simulator are fed into a Neural Network for learning the quality of a grasp.	24
3.3	A grasp pipeline for generating task-oriented grasp proposed by [8]. Objects are represented by their categories. The categorization of an object is based on their perception system. The result of perception as well as the user input are fed into the grasp prediction for generating grasps. The best grasp suited for the given task is selected by an offline-trained Baysian Network. Visual servoing is used to execute the selected grasp.	25

3.4	An example skill architecture proposed in [94]. The sensorimotor primitives are served as middle layer to connect low-level robot/sensor space to high level task space. The sensorimotor primitives can be reused within a D-connector skill, which is implemented by a Finite State Machine (FSM).	26
3.5	An example of skill architecture proposed in [129]. A knowledge-based system control module infers the required skills to execute a given task. The skill acquisition is based on multi-modal inputs. A skill is learned by teaching the correct sequence of atomic actions.	27
3.6	The structure of the framework	28
3.7	Assembly graph	29
3.8	Skill library	30
3.9	The assembly program	30
3.10	The diagram of state transitions for ‘Pick & Reconfigure’ skill. The low-level skill ‘Plan platform locations’ and ‘Grasp Object’ have an hierarchical structure, which includes further low-level skills organized by a state machine. In total seven low-level skills are used in the high-level skill ‘Pick & Reconfigure’. The low-level skill ‘Move Cartesian Goal’ is used twice for ‘PlaceDown’ and ‘MoveToPrePlaceLinear’.	32
3.11	An execution sequence of the Pick & Reconfigure skill	33
3.12	The state transition diagram of Pick & Insert skill.	35
3.13	An execution sequence of the Pick & Insert skill	36
3.14	The state transition diagram of ‘Move and locate’ skill.	41
3.15	The state transition diagram of ‘Grasp object’ skill.	42
3.16	Two common grasp situations	44
3.17	Grasps generated for verifying the proposed model	46
3.18	Grasping cube problem with different task contexts.	47
3.19	Probabilty of satisfying task constraints	48
3.20	Grasp realization on the SCHUNK-SDH gripper.	49
3.21	The virtual joints defined for a mobile platform.	50
3.22	The total time used in an assembly task by two methods. Planning arm-platform coordinated motion reduces the total time of assembly by 21%.	52
3.23	Assembly sequence parameterization for experiment I, II. & III.1	53
3.24	Performance evaluation in terms of different initial configurations	54
3.25	Assembly performance comparison in terms of different initial configurations	55
3.26	Performance evaluation in terms of different assembly sequence parameterization.	56
3.27	Evaluation in terms of different assembly sequence	57
3.28	Execution of the assembly task on a real robot. The parametrization of the assembly sequence corresponds to the one shown in Fig. 3.23.	58

3.29	A 2 m by 2 m by 2.6 m working space is discretized to voxels with the length of each voxel equaling 10 cm.	59
3.30	Poses generated in different density on the surface of an inscribed sphere. From left to right: sparse to dense	60
3.31	Visualization of a capability map at the height of 0.75 m. The magenta spheres indicate the region where the arm has the largest manipulability.	61
3.32	A cross-section visualization of the capability map from two perspectives.	61
3.33	Captability maps generated for two configurations of the robot	62
3.34	Capability map generated for picking-from-shelf scenario.	63
3.35	Start and goal configurations for benchmarking the motion planners in a limited working space.	64
3.36	An example planning result using arm-platform coordination	65
4.1	(a) A grasp synthesis method proposed by [66]. They consider the pose uncertainty by simulating object dynamics. (b) A method using pre-grasp manipulation to address the pose uncertainty [27]. They propose a planning framework which allows a robot to interact with objects by a library of actions such as push, slide or sweep operations. The pose uncertainty is addressed by the funneling effect of pushing.	71
4.2	(a) A method of representing object shape by Gaussian process implicit surface (GPIS) [131]. (b) A method for grasp synthesis considering shape uncertainty in 3D. They use GPIS to represent shape uncertainty of objects. The points for training the shape model is obtained from synthetic data [80]. (c) A method for grasp synthesis considering shape uncertainty in 2D [88]. The points for training the shape model is obtained from real sensor.	73
4.3	Learning based methods require a training set which include objects and associated grasps examples. Different methods can be used to acquire the training set. (a) Using human demonstration [46] (b) Execute random grasps with a single robot to acquire positive and negative grasp examples. [109] (c) Using a robot farm to acquire the training set. [78]	74
4.4	A method using hand-crafted feature for grasp classification proposed by [35]. The method discretizes input point cloud into height map. Features are extracted from height map for grasp classification. The best grasp is calculated by weighted grasp hypotheses and realized in a grasp planning simulator. Later, we use this method as baseline for comparison.	75
4.5	Illustration of a 1-D grasping problem.	78

4.6	Marginal grasp success probability in terms of two different object posteriors. The optimal gripper configurations are shown by red dots. In (a), $\mathcal{N}(0,0.3)$ is used to simulate the posterior of the object position. The marginal success probability $P(S)_{u=u_{\text{opt}}} = 0.13$. In (b) $\mathcal{N}(0,0.03)$ is used to simulate a more accurate posterior distribution of the object position with $P(S)_{u=u_{\text{opt}}} = 0.76$	79
4.7	Description of variables which represent a pre-touch configuration g_n on an bunny object. Fingertips are illustrated in blue. Dashed lines indicate variances of the representation. Arrows represent normals on the particular surfaces.	80
4.8	Description of the variables that we defined. The grid denotes the volume we consider for modelling an object. z^* is the true measurement from a depth camera. c denotes the pose of the depth camera. z_{sdf}^* is the true signed distance along the direction of the measurement.	83
4.9	Three grasp types are defined for SCHUNK-SDH gripper. The fingers of the gripper for an opening grasp posture P_b are marked in red, the ones for a closing grasp posture P_e are marked in green.	86
4.10	For each finger tip, 25 Frames are attached to the region, where contacts and forces are applied during grasping. Arrows in blue show the direction of operating ray castings.	87
4.11	Desired contact positions of a pre-touch configuration determined by ray casting. Magenta: contact points, yellow arrows: surface normals. For the purpose of visualization, this model is generated in Gazebo simulation. . .	87
4.12	Parametrization for refining a pre-touch configuration.	89
4.13	Simulation environment.	89
4.14	Synthetic noisy measurements with different standard deviation(sd). (a) original measurement (b) sd = 1 cm (c) sd = 2 cm (d) sd = 3 cm	90
4.15	Result of grasp synthesis and reconstructed objects based on measurement with different noise level. (a) original measurement (b) sd = 1 cm (c) sd = 2 cm (d) sd = 3 cm	91
4.16	Test objects we used in this experiment. From left to right: a 3-D printed ‘bunny’, ‘big duck’, ‘tape rectangular’, ‘tape triangle’, ‘stapler’, ‘toy car’, ‘duckie’, ‘screwdrawer’, ‘PS3 joystick’, ‘correction fluid’.	92
4.17	The process of modelling a ‘toy car’. Top: The robot moves its in-hand camera to several pre-defined poses to see the object from different viewing angles. Middle: Raycasted views of the model are shown alongside each robot pose. Bottom: A colored model of the object visualizes the uncertainty of the object during the modelling process. In the first three figures, the black points on the wheels indicate a large uncertainty on those regions, because the object was only observed from its top.	93

4.18	Number of frames used to model the 'toy car' (object see in Fig. 4.16). The time stamp of each bar corresponds to that of modelling result in Fig. 4.17.	93
4.19	The best grasp configuration found by the search algorithm when only one of the four criteria is activated. Green points indicate the contact regions. The yellow lines show the normal lines of each contact region.	95
4.20	The mean computation time of the grasp generation algorithm scales linearly with the number of samples. The variance is computed by evaluating 10 times for a given number of samples.	95
4.21	The algorithm converges approximately by evaluating 200 samples.	96
4.22	(a) A grasp generated by our algorithm for the 'stapler'. Our algorithm tends to generate grasp in the region where the uncertainty is small, which leads to a non-optimal grasp for this case. This grasp may lead to an in-gripper rotation, because it cannot counteract the wrench of gravity. (b) The black points indicate that the side surface of the object has a large uncertainty, however these regions are the only good contact region for a stable grasp. Since our algorithm avoids to generate grasps on uncertainty surfaces, a non-stable grasp is found at one control axis of the 'PS3 Joystick'.	97
4.23	Grasps generated by the algorithm with successful execution.	98
4.24	Successful grasps executed by the robot.	98
4.25	A comparison of a grasp calculated by the baseline method and our method. The grasp generated by the baseline approach probably leads to grasping slippage because the method does not consider contact surface as an important factor.	99
5.1	A grasp execution process contains three intermediate grasping phases.	104
5.2	Methods using vision for grasp execution. (a) A method [128] which combines visual perception and robot kinematics for determining motion of grasping. The approach was evaluated with the humanoid robot using the stereo system of the active head for perception as well as the torso and arms equipped with five finger hands for actuation. (b) A method [43] which uses visual servoing on grasping unknown objects. The grasp motion is executed with open loop. (c) A method [89] which allows grasping while walking for a biped humanoid robot.	105

5.3	Methods using force sensing for grasp execution. (a) In [105], the robot is first demonstrated how to grasp an object using kinesthetic teaching. At test time, the robot detects object displacement with strain gauge mounted on the gripper and adapts its grasp motion to a new object position. (b) A force compliant motion primitives is proposed in [63]. A coordinated lift motion and close motion ensures the gripper always maintains contact with a support plane. This method works well for flat objects lying on a horizontal plane. (c) A method for in-hand grasp adaptation proposed in [79]. A three-finger initial grasp is first established on an object. The position of one finger can be changed online to maintain the stability of the grasp, when the weight of the object is increased.	107
5.4	Tasks that require tight coordination in between the motion of the base and the motion of the arm. (a) In [103], a mobile manipulator equipped with a single arm performs door opening task. Cartesian impedance controller was proposed to control the elastic joints of the arm. (b) A graph search-based motion planning method [15] is proposed to plan trajectories for door opening. The method is able to handle a variety of door types. (c) A reactive whole-body control mechanism [23] for a torque controlled robot. The control method can be used for performing tasks such as catching a flying ball [5].	109
5.5	System architecture	111
5.6	An example of trajectory adaptation using DMPs	112
5.7	A reference trajectory for learning coordinated force closure motion.	113
5.8	Whole body control architecture	116
5.9	A circular object measured by a sensor. p_1 and p_2 are the measurements used in EKF for estimating the position and the radius of the object.	117
5.10	(a) Ten random sampled good pre-touch configurations, where the grasp likelihood larger than 0.9, (b) Ten random sampled bad pre-touch configurations, where the grasp likelihood smaller than 0.1.	119
5.11	An optimal pre-touch configuration computed for a given multivariate Gaussian posterior.	120
5.12	(a) The robot generates an initial estimate of the observed objects and their configuration parameters with the head-mounted 3D camera. This initial estimate is used to determine a pre-grasp configuration only. (b) The robot moves to the pre-grasp configuration. (c) The robot approaches the object and uses the hand-camera to continuously re-estimate object pose and radius. (d) Object is lifted up after being successfully grasped.	121
5.13	Blue line is the desired TCP goal configuration. Red line is the actual trajectory of the TCP. At the 25s, the gripper reaches pre-grasp configuration. Grasp motion phase begins at 25s. In this phase the robot reactively approaches pre-touch configuration based on new sensor measurements.	122

5.14 Illustration of the trajectory of a successful grasp	122
5.15 Comparison of grasp success rate between methods. In total, 900 grasp trials were performed to obtain the data.	123

List of Tables

2.1	Definition of variables	11
2.2	The conditional probability distribution for $P(S F, C)$. The superscript 1 and 0 indicate the variable equals success or failure respectively.	12
3.1	Planning success rate of three state-of-the-art motion planners in terms of 4 different goal configurations. 10 planning queries are conducted for each goal configuration to compute the statistic. In summary, 120 planning queries are conducted for the given scenario, only 26% overall success rate is achieved by the planners.	65
3.2	Arm-platform combined planning success rate of three state-of-the-art motion planners in terms of 4 different goal configurations. 10 planning queries are conducted for each goal configuration to compute the statistic. Comparing with the 26 % success rate of arm-only motion in Tab. 3.1, arm-platform combined planning achieves 83% overall success rate, which is a significant improvement.	65
4.1	(S. = success, F. = failure, S.w.P = Success with precision). Experimental result for a total of 200 grasping attempts. An outcome is classified as S.w.P when gripper successfully picked the object without displacing the object more than 1 cm or 30 degree. This criterion is strict comparing to a standard way to access grasping success.	97
4.2	Successful grasp executed by HAF (baseline method) and our method on a subset of the test objects. Our method achieves a lower error rate than the baseline method.	100

Bibliography

- [1] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuoka, “Physical human interactive guidance: Identifying grasping principles from human-planned grasps,” *Springer Tracts in Advanced Robotics*, vol. 95, pp. 477–500, 2014.
- [2] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, “Grasp planning in complex scenes,” in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*. IEEE, 2007, pp. 42–48.
- [3] D. Berenson, S. Srinivasa, and J. Kuffner, “Addressing pose uncertainty in manipulation planning using Task Space Regions,” in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 1419–1425.
- [4] A. Bicchi and V. Kumar, “Robotic grasping and contact: a review,” *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, pp. 348–353, 2000.
- [5] O. Birbach, U. Frese, and B. Bäuml, “Realtime perception for catching a flying ball with a mobile humanoid,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5955–5962.
- [6] J. Bohg and D. Kragic, “Learning grasping points with shape context,” *Robotics and Autonomous Systems*, vol. 58, no. 4, pp. 362–377, 2010.
- [7] J. Bohg, A. Morales, T. Asfour, and D. Kragic, “Data-Driven Grasp Synthesis - A Survey,” *Robotics, IEEE Transactions on*, vol. 30, no. 2, pp. 289–309, April 2014.
- [8] J. Bohg, K. Welke, B. León, M. Do, D. Song, W. Wohlkinger, M. Madry, A. Aldóma, M. Przybylski, T. Asfour *et al.*, “Task-based grasp adaptation on a humanoid robot,” *IFAC Proceedings Volumes*, vol. 45, no. 22, pp. 779–786, 2012.
- [9] C. Borst, M. Fischer, and G. Hirzinger, “Grasp planning: how to choose a suitable task wrench space,” in *IEEE International Conference on Robotics and Automation*, vol. 1, no. April, 2004, pp. 319–325.
- [10] L. Brignone and M. Howarth, “A geometrically validated approach to autonomous robotic assembly,” in *IEEE/RSJ International Conference on Intelligent Robots and System*, vol. 2, 2002, pp. 1626–1631.

- [11] R. C. Brost, “Automatic grasp planning in the presence of uncertainty,” *The International Journal of Robotics Research*, vol. 7, no. 1, pp. 3–17, 1988.
- [12] M. Buss, H. Hashimoto, and J. B. Moore, “Dextrous hand grasping force optimization,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 406–418, 1996.
- [13] L. Y. Chang, S. S. Srinivasa, and N. S. Pollard, “Planning pre-grasp manipulation for transport tasks,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2697–2704.
- [142] D. Chen and G. von Wichert, “An uncertainty-aware precision grasping process for objects with unknown dimensions,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015, pp. 4312–4317.
- [15] S. Chitta, B. Cohen, and M. Likhachev, “Planning for autonomous door opening with a mobile manipulator,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 1799–1806.
- [16] V. N. Christopoulos and P. Schrater, “Handling shape and contact location uncertainty in grasping two-dimensional planar objects,” in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 1557–1563.
- [17] M. T. Ciocarlie and P. K. Allen, “Hand Posture Subspaces for Dexterous Robotic Grasping,” *Int. J. Rob. Res.*, vol. 28, no. 7, pp. 851–867, July 2009.
- [18] H. Dang and P. Allen, “Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 1311–1317.
- [19] H. Dang and P. K. Allen, “Stable grasping under pose uncertainty using tactile feedback,” *Autonomous Robots*, vol. 36, no. 4, pp. 309–330, 2014.
- [20] —, “Grasp adjustment on novel objects using tactile experience from similar local geometry,” in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 4007–4012.
- [21] —, “Learning grasp stability,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2012, pp. 2392–2397.
- [22] R. Diankov and J. Kuffner, “Openrave: A planning architecture for autonomous robotics,” *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, p. 79, 2008.

-
- [23] A. Dietrich, T. Wimbock, and A. Albu-Schaffer, “Dynamic whole-body mobile manipulation with a torque controlled humanoid robot via impedance control laws,” in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, Sept 2011, pp. 3199–3206.
- [24] V. Dietrich, D. Chen, K. Wurm, G. v. Wichert, and P. Ennen, “Probabilistic Multi-Sensor Fusion based on Signed Distance Functions,” in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016.
- [25] R. Dillmann, O. Rogalla, M. Ehrenmann, R. Zöllner, and M. Bordegoni, “Learning Robot Behaviour and Skills Based on Human Demonstration and Advice: The Machine Learning Paradigm,” *Control*, vol. 9, pp. 229–238, 2000.
- [26] B. Dizioğlu and K. Lakshiminarayana, “Mechanics of form closure,” *Acta Mechanica*, vol. 52, no. 1-2, pp. 107–118, 1984.
- [27] M. Dogar and S. Srinivasa, “A Framework for Push-Grasping in Clutter,” in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.
- [28] S. Dragiev, M. Toussaint, and M. Gienger, “Uncertainty aware grasping and tactile exploration,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 113–119.
- [29] —, “Gaussian process implicit surfaces for shape estimation and grasping,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 2845–2850.
- [30] S. Ekvall and D. Kragic, “Learning and evaluation of the approach vector for automatic grasp generation and planning,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 4715–4720.
- [31] C. Eppner and O. Brock, “Grasping unknown objects by exploiting shape adaptability and environmental constraints,” in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 4000–4006.
- [32] J. Felip and A. Morales, “Robust sensor-based grasp primitive for a three-finger robot hand,” in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 1811–1816.
- [33] C. Ferrari and J. Canny, “Planning optimal grasps,” *Proceedings 1992 IEEE International Conference on Robotics and Automation*, 1992.
- [34] D. Fischinger and M. Vincze, “Empty the basket—a shape based learning approach for grasping piles of unknown objects,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 2051–2057.

- [35] D. Fischinger, A. Weiss, and M. Vincze, “Learning Grasps with Topographic Features,” *I. J. Robotic Res.*, vol. 34, no. 9, pp. 1167–1194, 2015. [Online]. Available: <http://dx.doi.org/10.1177/0278364915577105>
- [36] ———, “Learning Grasps with Topographic Features,” *I. J. Robotic Res.*, vol. 34, no. 9, pp. 1167–1194, 2015.
- [37] H. Fujimoto, L. C. Zhu, and K. Abdel-Malek, “Image-based visual servoing for grasping unknown objects,” in *IECON Proceedings (Industrial Electronics Conference)*, vol. 1, 2000, pp. 876–881.
- [38] S. Geidenstam, K. Huebner, D. Banksell, and D. Kragic, “Learning of 2D grasping strategies from box-based 3D object approximations.” in *Robotics: Science and Systems*, vol. 2008, 2009, pp. 9–16.
- [39] J. Glover, D. Rus, and N. Roy, “Probabilistic models of object geometry for grasp planning,” *Proceedings of Robotics: Science and Systems IV, Zurich, Switzerland*, pp. 278–285, 2008.
- [40] K. Y. Goldberg and M. T. Mason, “Bayesian grasping,” in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on.* IEEE, 1990, pp. 1264–1269.
- [41] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelosof, “Grasp planning via decomposition trees,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2007, pp. 4679–4684.
- [42] B. Graf, U. Reiser, M. Hagele, K. Mauz, and P. Klein, “Robotic home assistant Care-O-bot® 3-product vision and innovation platform,” in *Advanced Robotics and its Social Impacts (ARSO), 2009 IEEE Workshop on*, 2009, pp. 139–144.
- [43] X. Gratal, J. Romero, J. Bohg, and D. Kragic, “Visual servoing on unknown objects,” *Mechatronics*, vol. 22, no. 4, pp. 423–435, 2012.
- [44] M. Gridseth, K. Hertkorn, and M. Jagersand, “On Visual Servoing to Improve Performance of Robotic Grasping,” in *Proceedings -2015 12th Conference on Computer and Robot Vision, CRV 2015*, 2015, pp. 245–252.
- [45] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, and S. Schaal, “Template-based learning of grasp selection,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on.* IEEE, 2012, pp. 2379–2384.
- [46] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, J. Bohg, T. Asfour, and S. Schaal, “Learning of grasp selection based on shape-templates,” *Autonomous Robots*, vol. 36, no. 1-2, pp. 51–65, 2014.

-
- [47] D. Holz, A. Topalidou-Kyniazopoulou, F. Rovida, M. R. Pedersen, V. Krüger, and S. Behnke, “A skill-based system for object perception and manipulation for automating kitting tasks,” in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, vol. 2015-October, 2015.
- [48] G. Hovland, P. Sikka, and B. McCarragher, “Skill acquisition from human demonstration using a hidden Markov model,” *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2706–2711, 1996.
- [49] —, “Contact-reactive grasping of objects with partial shape information,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010.
- [50] K. Hsiao, M. Ciocarlie, and P. Brook, “Bayesian grasp planning,” in *ICRA 2011 Workshop on Mobile Manipulation: Integrating Perception and Manipulation*, 2011.
- [51] K. Hsiao and L. P. Kaelbling, “Task-Driven Tactile Exploration,” *Proceedings of Robotics Science and Systems*, 2010.
- [52] K. Hsiao, L. P. Kaelbling, and T. Lozano-Perez, “Grasping pomdps,” in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 4685–4692.
- [53] N. Hudson, T. Howard, J. Ma, A. Jain, M. Bajracharya, S. Myint, C. Kuo, L. Matthies, P. Backes, P. Hebert, T. Fuchs, and J. Burdick, “End-to-end dexterous manipulation with deliberate interactive estimation,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2012, pp. 2371–2378.
- [54] K. Huebner and D. Kragic, “Selection of robot pre-grasps using box-based shape approximation,” in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, Sept 2008, pp. 1765–1770.
- [55] K. Huebner, S. Ruthotto, and D. Kragic, “Minimum volume bounding box decomposition for shape approximation in robot grasping,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2008, pp. 1628–1633.
- [56] K. Huebner, K. Welke, M. Przybylski, N. Vahrenkamp, T. Asfour, D. Kragic, and R. Dillmann, “Grasping Known Objects with Humanoid Robots: A Box-Based Approach,” *International Conference on Advanced Robotics*, pp. 1–6, 2009.
- [57] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, “Dynamical movement primitives: learning attractor models for motor behaviors,” *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.
- [58] —, “Seashell effect pretouch sensing for robotic grasping,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2851–2858.

- [59] Y. Jiang, S. Moseson, and A. Saxena, “Efficient grasping from RGBD images: Learning using a new rectangle representation,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3304–3311, 2011.
- [60] S. G. Johnson, “The NLOpt nonlinear-optimization package,” 2010.
- [61] I. Kamon, T. Flash, and S. Edelman, “Learning to grasp using visual information,” *Proceedings of IEEE International Conference on Robotics and Automation*, no. April, pp. 2470–2476, 1996.
- [62] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.
- [63] M. K. Pollard, J.-S. Valois, J. A. Bagnell, and Nancy, “Robust Object Grasping using Force Compliant Motion Primitives,” in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.
- [64] B. Kehoe, D. Berenson, and K. Goldberg, “Toward cloud-based grasping with uncertainty in shape: Estimating lower bounds on achieving force closure with zero-slip push grasps,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 576–583.
- [65] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, and A. Casal, “Vehicle/arm coordination and multiple mobile manipulator decentralized cooperation,” in *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, vol. 2, 1996, pp. 546–553.
- [66] J. Kim, K. Iwamoto, J. J. Kuffner, Y. Ota, and N. S. Pollard, “Physically based grasp quality evaluation under pose uncertainty,” *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1424–1439, 2013.
- [67] D. Koller and N. Friedman, *Probabilistic Graphical Models- Principles and Techniques*, 1989, vol. 53.
- [68] M. Kopicki, R. Detry, F. Schmidt, C. Borst, R. Stolkin, and J. L. Wyatt, “Learning dexterous grasps that generalise to novel objects by combining hand and contact models,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 5358–5365.
- [69] K. B. Korb and Ann E. Nicholson, “Introducing Bayesian Networks,” *Bayesian artificial intelligence*, pp. 29–54, 2003.
- [70] T. Kröger, B. Finkemeyer, and F. M. Wahl, “Manipulation Primitives - A Universal Interface between Sensor-Based Motion Control and Robot Programming,” *Robotic Systems for Handling and Assembly*, pp. 293–313, 2010.

-
- [71] D. Kragić, A. T. Miller, and P. K. Allen, “Realtime tracking meets online grasp planning,” in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 3, 2001, pp. 2460–2465.
- [72] D. Kragic and H. I. Christensen, “Survey on Visual Servoing for Manipulation,” *Computer*, vol. 15, no. ISRN KTH/NA/P-02/01-SE CVAP259, p. 58, 2002.
- [73] O. B. Kroemer, R. Detry, J. Piater, and J. Peters, “Combining active learning and reactive control for robot grasping,” *Robotics and Autonomous Systems*, vol. 58, no. 9, pp. 1105–1116, 2010.
- [74] J. J. Kuffner and S. M. LaValle, “RRT-connect: An efficient approach to single-query path planning,” in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.
- [75] J. Laaksonen, E. Nikandrova, and V. Kyrki, “Probabilistic sensor-based grasping,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 2019–2026, 2012.
- [76] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, “Fast proximity queries with swept sphere volumes,” Technical Report TR99-018, Department of Computer Science, University of North Carolina, Tech. Rep., 1999.
- [77] I. Lenz, H. Lee, and A. Saxena, “Deep learning for detecting robotic grasps,” *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015.
- [78] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, “Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection,” *arXiv*, 2016.
- [79] M. Li, Y. Bekiroglu, D. Kragic, and A. Billard, “Learning of grasp adaptation through experience and tactile sensing,” in *IEEE International Conference on Intelligent Robots and Systems*, 2014, pp. 3339–3346.
- [80] M. Li, K. Hang, D. Kragic, and A. Billard, “Dexterous grasping under shape uncertainty,” *Robotics and Autonomous Systems*, vol. 75, pp. 352–364, 2016.
- [81] A. Liégeois, “Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.
- [82] Y. Lin and Y. Sun, “Grasp planning to maximize task coverage,” *The International Journal of Robotics Research*, vol. 34, no. 9, pp. 1–16, 2015.
- [83] ———, “Robot grasp planning based on demonstrated grasp strategies,” *The International Journal of Robotics Research*, vol. 34, no. 1, pp. 26–42, 2014.

- [84] G. Liu and Z. Li, “Real-time grasping-force optimization for multifingered manipulation: Theory and experiments,” *IEEE/ASME Transactions on Mechatronics*, vol. 9, no. 1, pp. 65–77, 2004.
- [85] I. Lopez-Juarez, J. Corona-Castuera, M. Peña-Cabrera, and K. Ordaz-Hernandez, “On the design of intelligent robotic agents for assembly,” *Information Sciences*, vol. 171, no. 4, pp. 377–402, 2005.
- [86] H. Maekawa, K. Tanie, and K. Komoriya, “Dynamic grasping force control using tactile feedback for grasp of multifingered hand,” *IEEE International Conference on Robotics and Automation*, vol. 3, no. April, pp. 2462–2469, 1996.
- [87] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, “Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics,” *CoRR*, vol. abs/1703.09312, 2017. [Online]. Available: <http://arxiv.org/abs/1703.09312>
- [88] J. Mahler, S. Patil, B. Kehoe, J. van den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg, “GP-GPIS-OPT: Grasp planning with shape uncertainty using Gaussian process implicit surfaces and Sequential Convex Programming,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, May 2015, pp. 4919–4926.
- [89] N. Mansard, O. Stasse, F. Chaumette, and K. Yokoi, “Visually-guided grasping while walking on a humanoid robot,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2007, pp. 3041–3047.
- [90] A. Miller and P. Allen, “Graspit! A versatile simulator for robotic grasping,” *Robotics Automation Magazine, IEEE*, vol. 11, no. 4, pp. 110–122, Dec 2004.
- [91] —, “Examples of 3D grasp quality computations,” *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2, no. May, pp. 1240–1246, 1999.
- [92] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, “Automatic grasp planning using shape primitives,” *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 2, pp. 1824–1829 vol.2, 2003.
- [93] J. Morrow and P. Khosla, “Manipulation task primitives for composing robot skills,” *Proceedings of International Conference on Robotics and Automation*, vol. 4, no. April, pp. 3354–3359, 1997.
- [94] —, “Sensorimotor primitives for robotic assembly skills,” *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, no. May, pp. 234–240, 1995.

-
- [95] H. Mosemann and F. M. Wahl, “Automatic decomposition of planned assembly sequences into skill primitives,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 5, pp. 709–718, 2001.
- [96] A. Muis and K. Ohnishi, “Eye-to-hand approach on eye-in-hand configuration within real-time visual servoing,” *IEEE/ASME Transactions on Mechatronics*, vol. 10, no. 4, pp. 404–410, 2005.
- [97] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, “Comparative experiments on task space control with redundancy resolution,” in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2005, pp. 1575–1582.
- [98] B. J. Nelson, N. P. Papanikolopoulos, and P. K. Khosla, “Robotic visual servoing and robotic assembly tasks,” *IEEE Robotics and Automation Magazine*, vol. 3, no. 2, pp. 23–31, 1996.
- [99] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, “KinectFusion: Real-time dense surface mapping and tracking,” in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE, 2011, pp. 127–136.
- [100] V.-D. Nguyen, “Constructing Force- Closure Grasps,” *The International Journal of Robotics Research*, vol. 7, no. 3, pp. 3–16, 1988.
- [101] T. Ogata, S. Sugano, and J. Tani, “Acquisition of Motion Primitives of Robot in Human-Navigation Task,” *Transactions of the Japanese Society for Artificial Intelligence*, vol. 20, pp. 188–196, 2005.
- [102] D. Omrcen, L. Lajpah, B. Nemec, and J. Babi, “Torque-Velocity Control of Mobile Manipulator in Unstructured Environment,” in *RAAD 03 12th International Workshop on Robotics in Alpe-Adria-Danube Region*, 2003.
- [103] C. Ott, B. Bäuml, C. Borst, and G. Hirzinger, “Employing cartesian impedance control for the opening of a door: A case study in mobile manipulation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop on mobile manipulators: Basic techniques, new trends & applications*, 2005.
- [104] R. Paolini, A. Rodriguez, S. S. Srinivasa, and M. T. Mason, “A data-driven statistical framework for post-grasp manipulation,” *The International Journal of Robotics Research*, vol. 33, no. 4, pp. 600–615, 2014.
- [105] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, “Online movement adaptation based on previous sensor experiences,” in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 365–371.

- [106] M. R. Pedersen, L. Nalpantidis, R. S. Andersen, C. Schou, S. Bøgh, V. Krüger, and O. Madsen, “Robot skills for manufacturing: From concept to industrial deployment,” *Robotics and Computer-Integrated Manufacturing*, vol. 37, pp. 282–291, 2016.
- [107] R. Pelossof, A. Miller, P. Allen, and T. Jebara, “An SVM learning approach to robotic grasping,” in *Robotics and Automation, 2004. Proceedings. ICRA’04. 2004 IEEE International Conference on*, vol. 4. IEEE, 2004, pp. 3512–3518.
- [108] J. M. Phillips, N. Bedrossian, and L. E. Kavraki, “Guided expansive spaces trees: A search strategy for motion-and cost-constrained state spaces,” in *Robotics and Automation, 2004. Proceedings. ICRA’04. 2004 IEEE International Conference on*, vol. 4. IEEE, 2004, pp. 3968–3973.
- [109] L. Pinto and A. Gupta, “Supersizing Self-supervision: Learning to Grasp from 50K Tries and 700 Robot Hours,” *arXiv preprint arXiv:1509.06825*, 2015.
- [110] M. Przybylski, T. Asfour, and R. Dillmann, “Planning grasps for robotic hands using a novel object representation based on the medial axis transform,” in *IEEE International Conference on Intelligent Robots and Systems*, 2011, pp. 1781–1788.
- [111] D. Rao, Q. V. Le, T. Phoka, M. Quigley, A. Sudsang, and A. Y. Ng, “Grasping novel objects with depth segmentation,” in *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 2010, pp. 2578–2585.
- [112] C. E. Rasmussen, *Gaussian processes for machine learning*. Citeseer, 2006.
- [113] G. Recatala, P. Sanz, E. Cervera, and A. del Pobil, “Grasp-based visual servoing for gripper-to-object positioning,” in *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1, 2004, pp. 118–123.
- [114] J. Redmon and A. Angelova, “Real-time grasp detection using convolutional neural networks,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1316–1322.
- [115] M. A. Roa and R. Suárez, “Grasp quality measures: review and performance,” *Autonomous Robots*, vol. 38, no. 1, pp. 65–88, 2015.
- [116] —, “Computation of independent contact regions for grasping 3-d objects,” *Robotics, IEEE Transactions on*, vol. 25, pp. 839–850, 2009.
- [117] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, “Human-inspired robotic grasp control with tactile sensing,” *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1067–1079, 2011.

-
- [118] J. Romero, H. Kjellstrom, and D. Kragic, “Modeling and evaluation of human-to-robot mapping of grasps,” *2009 International Conference on Advanced Robotics*, 2009.
- [119] T. H. Rowan, “Functional Stability Analysis of Numerical Algorithms,” Ph.D. dissertation, University of Texas at Austin, Austin, TX, USA, 1990, uMI Order No. GAX90-31702.
- [120] J. K. Salisbury and B. Roth, “Kinematic and Force Analysis of Articulated Mechanical Hands,” *Journal of Mechanisms Transmissions and Automation in Design*, vol. 105, no. 1, p. 35, 1983.
- [121] P. R. Sinha and J. M. Abel, “A Contact Stress Model for Multifingered Grasps of Rough Objects,” *IEEE Transactions on Robotics and Automation*, vol. 8, no. 1, pp. 7–22, 1992.
- [122] M. Skubic and R. A. Volz, “Acquiring robust, force-based assembly skills from human demonstration,” *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 772–781, 2000.
- [123] D. Song, C. Ek, K. Huebner, and D. Kragic, “Multivariate discretization for Bayesian Network structure learning in robot grasping,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 1944–1950.
- [124] D. Song, K. Huebner, V. Kyrki, and D. Kragic, “Learning task constraints for robot grasping using graphical models,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 1579–1585.
- [125] F. Stulp, E. Theodorou, J. Buchli, and S. Schaal, “Learning to grasp under uncertainty,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 5703–5708.
- [126] I. A. Sucas and S. Chitta, “Moveit!” *Online Available: <http://moveit.ros.org>*, 2013.
- [127] J. Tegin, S. Ekvall, D. Kragic, J. Wikander, and B. Iliev, “Demonstration-based learning and control for automatic grasping,” *Intelligent Service Robotics*, vol. 2, no. 1, pp. 23–30, 2009.
- [128] N. Vahrenkamp, S. Wieland, P. Azad, D. Gonzalez, T. Asfour, and R. Dillmann, “Visual servoing for humanoid grasping and manipulation tasks,” in *2008 8th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2008*, 2008, pp. 406–412.
- [129] F. Wallhoff, J. Blume, A. Bannat, W. Rösel, C. Lenz, and A. Knoll, “A skill-based approach towards hybrid assembly,” *Advanced Engineering Informatics*, vol. 24, no. 3, pp. 329–339, 2010.

- [130] J. Weisz and P. K. Allen, “Pose error robust grasping from contact wrench space metrics,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 557–562.
- [131] O. Williams and A. Fitzgibbon, “Gaussian process implicit surfaces,” *Gaussian Proc. in Practice*, 2007.
- [132] Z. Xue, A. Kasper, J. M. Zoellner, and R. Dillmann, “An automatic grasp planning system for service robots,” in *Advanced Robotics, 2009. ICAR 2009. International Conference on*. IEEE, 2009, pp. 1–6.
- [133] T. Yoshikawa, “Manipulability of robotic mechanisms,” *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.
- [134] F. Zacharias, C. Borst, and G. Hirzinger, “Capturing robot workspace structure: representing robot capabilities,” in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*. IEEE, 2007, pp. 3229–3236.
- [135] F. Zacharias, C. Borst, S. Wolf, and G. Hirzinger, “The Capability Map: A Tool to Analyze Robot Arm Workspaces,” *International Journal of Humanoid Robotics*, vol. 10, no. 04, p. 1350031, 2013.
- [136] M. Zuliani, “RANSAC for Dummies,” *Citeseer*, 2008.

List Of Own Publications And Collaborations

- [137] D. Chen, V. Dietrich, and G. von Wichert. Precision grasping based on probabilistic models of unknown objects. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2044–2051, May 2016.
- [138] D. Chen, Z. Liu, and G. von Wichert. Grasping on the move: A generic arm-base coordinated grasping pipeline for mobile manipulation. In *2013 European Conference on Mobile Robots*, pages 349–354, Sep. 2013.
- [139] D. Chen and G. v. Wichert. Uncertainty-aware arm-base coordinated object grasping with a mobile manipulation platform. In *ISR/Robotik 2014; 41st International Symposium on Robotics*, pages 1–6, June 2014.
- [140] D. Chen and G. von Wichert. An uncertainty-aware precision grasping process for objects with unknown dimensions. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4317, May 2015.
- [141] Dong Chen, Vincent Dietrich, Ziyuan Liu, and Georg von Wichert. A Probabilistic Framework for Uncertainty-Aware High-Accuracy Precision Grasping of Unknown Objects. *Journal of Intelligent and Robotic Systems*, 90(1):19–43, 2018.
- [142] Dong Chen, Ziyuan Liu, and Georg von Wichert. Uncertainty-Aware Arm-Base Coordinated Grasping Strategies for Mobile Manipulation. *Journal of Intelligent and Robotic Systems*, 80(1):205–223, 2015.
- [143] V. Dietrich, Dong Chen, K. M. Wurm, G. v. Wichert, and P. Ennen. Probabilistic multi-sensor fusion based on signed distance functions. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1873–1878, May 2016.
- [144] Z. Liu, D. Chen, and G. von Wichert. 2d semantic mapping on occupancy grids. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6, May 2012.
- [145] Z. Liu, D. Chen, and G. von Wichert. Online semantic exploration of indoor maps. In *2012 IEEE International Conference on Robotics and Automation*, pages 4361–4366, May 2012.
- [146] Z. Liu, D. Chen, K. M. Wurm, and G. von Wichert. Using rule-based context knowledge to model table-top scenes. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2646–2651, May 2014.
- [147] Z. Liu, W. Wang, D. Chen, and G. von Wichert. A coherent semantic mapping system based on parametric environment abstraction and 3d object localization. In *2013 European Conference on Mobile Robots*, pages 234–239, Sep. 2013.

- [148] Ziyuan Liu, Dong Chen, Kai M. Wurm, and Georg von Wichert. Table-top scene analysis using knowledge-supervised mcmc. *Robot. Comput.-Integr. Manuf.*, 33(C):110–123, June 2015.