KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT INGENIEURSWETENSCHAPPEN
DEPARTEMENT WERKTUIGKUNDE
AFDELING PRODUKTIETECHNIEKEN,
MACHINEBOUW EN AUTOMATISERING
Celestijnenlaan 300B, B-3001 Heverlee (Leuven), Belgium

COMPLIANT ROBOT MOTION: FROM PATH PLANNING OR HUMAN DEMONSTRATION TO FORCE CONTROLLED TASK EXECUTION.

Jury:
Prof. dr. ir. P. Van Houtte, voorzitter
Prof. dr. ir. J. De Schutter, promotor
Prof. dr. ir. H. Bruyninckx, promotor
Prof. dr. ir. P. Suetens
Prof. dr. ir. Y. Berbers
Prof. dr. J. Baeten
Prof. dr. J. Xiao,
University of North Carolina at Charlotte, NC, USA

Proefschrift voorgedragen tot het bekomen van de graad van Doctor in de Ingenieurswetenschappen

door

Wim Meeussen

ISBN 978-90-5682-753-3
D/2006/7515/87
UDC 681.3*I29

21 December 2006

D/2006/7515/87
ISBN 978-90-5682-753-3
UDC 681.3*I29

Preface

A man travels the world over in search of what he needs, and returns home to find it.

George Moore

After twenty-four years in the small city of Leuven, I could not have imagined how four years of research would take me all around the world. From the ever sunny and warm Charlotte North Carolina, I went to –the at that time still dry– New Orleans Louisiana, then from hectic Barcelona Spain to summer Heidelberg Germany to cold Madrid Spain all over to the soccer-loving Rio De Janeiro Brazil to mountain climbing Edmonton Canada and not to forget to the deserty Albuquerque New Mexico. In every place I got the opportunity to meet new people, share ideas, and receive valuable feedback on my research. Even more than on these many trips, back in Leuven there where many people who contributed in many ways to the development of my research. First of all there was the contagious enthusiasm and unflagging support of Herman and Joris. With Joris' eye for detail and Herman's ability to see connections between all fields of research, together they form the perfect team. I want to thank both of them for giving me the opportunity and freedom to do research and collaborate with many interesting people around the world. Joris and Herman surrounded themselves with an inspiring group of students: oldtimer Johan, finetuning Peter, Bayesian Klaas, realtime Peter, low-pressure Walter, nonminimal Tine, vacationing Kasper, bugreport Ruben, unremitting Tinne,

breakthrough Diederik and smiley Wilm. I want thank all of them for the many discussions and great times at the department. Special thanks go to Johan; ever since our first project together, now almost eight years ago, Johan and I had countless passionate and inspiring discussions that greatly improved the quality of my research.

Then of course there is my second home in Charlotte, with Jing and her research group. Jing invests an incredible amount of time and energy in every one of her students; she even traveled all the way to Leuven to jury my Ph.D. defense. Combining Jing's top-down approach to compliant motion with the bottom-up approach in Leuven proved to be effective. Working in her research group was a new and exciting experience, and helped me to find clear goals for my own research. I want to thank her for this unique opportunity to spend a whole year in a different lab in a different world. I will always remember the good old times with Qi, Peng, Yuli and of course Ernesto, Pedro, Christian, Slava, Valerie, Anna, Zennek, Marian and Mona. I also want to thank Ernesto for giving me the opportunity to visit him at his new university in Madrid, to continue our close collaboration together with Stefano, Maria Eugenia and Raquel.

I need to thank the people that worked hard to follow up my research during four years, and provide valuable feedback to make this dissertation clear and coherent. Thank you prof. Berbers and prof. Suetens. Also many thanks to prof. Van Houtte and Dr. Baeten to jury the defense of my research.

Last but not least I want to thank everyone in my family. My fiance Amy, who moved halfway across the globe, away from her family and friends, to support me throughout my work at the university. My mother, who has always done so much for me, put up with me and supported me in good and bad times. Also my three sisters Bieke, Marjolein and Loes, Erwin, Karl, and my new family in New Mexico Kent, Sandra and Amy's brothers and sisters. My family's continuing support, care and love allowed me to successfully complete this work. I dedicate this work to my father, who passed away too early.

Wim Meeussen.
Leuven, December 21, 2006.

Abstract

Still today, industrial robot assembly tasks are executed by replaying a pre-programmed trajectory, making them vulnerable to geometric uncertainties in the environment. Therefore, industrial assembly tasks are executed in expensive structured environments, which limits the use of robots to high volume and repetitive tasks, where the cost of the structured environment becomes relatively small. By equipping the robot with sensors, it can observe its environment and interact with its environment, allowing it to operate in less expensive, unstructured environments. Assembly tasks are only one example of compliant motion tasks, where a robot manipulates an object in contact with an environmental object. The force interaction between the contacting objects guides the manipulated object along the environmental object to overcome geometric uncertainties associated with the task.

Aiming towards more intelligent and flexible robots, this thesis presents two high level approaches for sensor based compliant motion task specification. The first approach is based on a compliant motion path planner that generates a compliant path given the geometric models of the contacting objects. The path is expressed by the relative positions between the objects, and the desired contact formations. The second task specification method exploits the advanced manipulation skills of a human, to obtain a compliant path. While a human operator uses a demonstration tool to demonstrate the targeted compliant motion task, sensors on the demonstration tool measure contact forces, positions and velocities. Applying state of the art Bayesian sequential Monte Carlo methods (also known as particle filters), the sensor measurements are combined to simultaneously estimate continuous geometrical parameters and recognize the discrete contact formation between the objects. The simultaneous estimation is helped by the availability of a contact state graph of all possible contact formations.

This thesis also presents the compliant task generator, an approach to automatically convert the output from the compliant path planner, or the output from a human demonstration, into a task specification for a hybrid robot controller. The planner or demonstration output is a geometric descrip-

tion of a compliant path, while the hybrid controller requires instantaneous force and velocity setpoints. The compliant task generator automatically converts a geometric compliant path into controller setpoints, allowing to plan or demonstrate a compliant motion task and immediately execute it on a real robot manipulator under active force control.

Finally, this thesis uses the presented Bayesian estimators to monitor discrete contact formation transitions online, during the execution of a compliant motion task. This provides a feedback to the generator and controller components about transitions between different sub-tasks, and allows the system to select the most appropriate controller strategy for the current contact formation between the manipulated object and its environment.

Beknopte samenvatting

De huidige industriële robots voeren assemblagetaken uit door een vooraf geprogrammeerd traject af te spelen, wat de uitvoering van de taak gevoelig maakt voor onzekerheden in de omgeving. Daarom worden assemblagetaken uitgevoerd in dure, gestructureerde omgevingen, wat het gebruik van robots beperkt tot grote series en zich herhalende taken, waar de kost van de gestructureerde omgeving relatief klein wordt. Door de robot met sensoren uit te rusten, kan hij zijn omgeving waarnemen, en kan hij interageren met zijn omgeving, waardoor het mogelijk wordt de robot in te zetten in goedkopere, ongestructureerde omgevingen. Assemblagetaken zijn slechts één voorbeeld van taken waarbij een robot een object manipuleert in contact met de omgeving. De krachtinteractie tussen de objecten in contact leidt het gemanipuleerde object langs het oppervlak van het omgevingsobject; op deze manier gaat de robot om met de geometrische onzekerheden in de taak.

Met het doel robots intelligenter en flexibeler te maken, stelt dit proefschrift twee hoogniveau methodes voor om sensorgebaseerde taken in contact te specificeren. De eerste methode is gebaseerd op een padplanner voor beweging in contact, die zelf gebruikt maakt van het geometrisch model van de objecten in contact. Het pad in contact is uitgedrukt aan de hand van de relatieve posities van de objecten, en de gewenste contacten tussen de objecten. De tweede methode voor het specificeren van taken in contact gebruikt de vaardigheden van mensen om objecten te manipuleren, om een pad in contact te bekomen. Terwijl een mens met een demonstratiehulpmiddel een taak in contact demonstreert, meten verschillende sensoren op het demonstratiehulpmiddel de contactkrachten, de posities en de snelheden. Door moderne Bayesiaanse sequentiële Monte Carlo-methodes toe te passen (ook gekend als deeltjesfilters), worden de sensorgegevens gecombineerd om gelijktijdig de continue geometrische parameters te schatten, en de discrete contacten tussen de objecten te herkennen. De gelijktijdige schatting wordt geholpen door de beschikbaarheid van een contactgrafe, die alle mogelijke contacten tussen twee objecten bevat.

Dit proefschrift stelt ook de contacttaakgenerator voor, een methode om

de uitvoer van een contactplanner, of de uitvoer van een menselijke demonstratie, om te zetten in een taakspecificatie voor een hybride robotcontrole. De uitvoer van de planner en de demonstratie is een geometrische beschrijving van een pad in contact, maar de hybride controle heeft ogenblikkelijke kracht- en snelheidwaardes nodig. De contacttaakgenerator zet een geometrisch pad automatisch om in waardes voor de controle, wat het mogelijk maakt om een contactpad te plannen of te demonstreren, en dit pad dan onmiddellijk uit te voeren op een echte robot met actieve krachtcontrole.

Ten slotte zet dit proefschrift Bayesiaanse schatters in om discrete veranderingen in contacten tijdens de uitvoering van een taak in contact te herkennen. De contactveranderingen worden teruggekoppeld naar de generator- en controlecomponent, wat hen in staat stelt de overgangen tussen verschillende deeltaken te herkennen, en de meest gepaste controlestrategie te selecteren voor de huidige contacten tussen het gemanipuleerde object en zijn omgeving.

Symbols, definitions and abbreviations

General abbreviations

1D, 2D, 3D	: 1-, 2-, or 3-dimensional
ACM	: Active Compliant Motion
BFL	: Bayesian Filtering Library
BN	: Bayesian Network
CF	: Contact Formation
CAD	: Computer Aided Design
C-space	: Configuration space
DOF	: Degrees of Freedom
EC	: Elementary Contact
EE	: End-Effector
EKF	: Extended Kalman Filter
GCR	: Goal Contact Relaxation
HCP	: Hybrid Control Paradigm
HMM	: Hidden Markov Model
IEKF	: Iterated Extended Kalman Filter
IMM	: Interacting Multiple Model
KF	: Kalman Filter
NC	: Numerically Controlled
NMSKF	: Non-Minimal State Kalman Filter
OROCOS	: Open Robot Control Software
PbD	: Programming by Demonstration
PC	: Principal Contact
PID	: Proportional Integral Derivative
PF	: Partile Filter
PRM	: Probabilistic Roadmap
SLAM	: Simultaneous Localization and Map Building
SNIS	: Summed Normalized Innovation Squared
SVD	: Singular Value Decomposition

SVM	: Support Vector Machine
TFF	: Task Frame Formalism

General symbols and definitions

a	: scalar (unbold lower case)
$\boldsymbol{a}$	: column vector (bold lower case)
$\boldsymbol{A}$	: matrix (bold upper case)
$\boldsymbol{A}^T$	: transpose of a matrix
$\boldsymbol{A}^{-1}$	: inverse of a matrix
$\boldsymbol{A}^\dagger$	: Moore Penrose pseudo inverse of a matrix
$\boldsymbol{A}^{\dagger\boldsymbol{\kappa}}$	: weighted Moore Penrose pseudo inverse of a matrix
$\tilde{\boldsymbol{a}}$	: predicted value of a vector
$\hat{\boldsymbol{a}}$	: estimated value of a vector
$\dot{\boldsymbol{a}}$	: first time derivative of a vector
$\ddot{\boldsymbol{a}}$	: second time derivative of a vector
$\boldsymbol{a}_m$	: measured value of a vector
$\boldsymbol{a}_d$	: desired value of a vector
$\boldsymbol{a}_c$	: controlled value of a vector
$\boldsymbol{0}$	: zero matrix
$\boldsymbol{I}$	: ones diagonal matrix

Objects names

$\boldsymbol{X}_w$	: pose of world frame
$\boldsymbol{X}_t$	: pose of demonstration tool frame
$\boldsymbol{X}_k$	: pose of Krypton camera frame
$\boldsymbol{X}_s$	: pose of wrench sensor frame
$\boldsymbol{p}_c$	: position of center of gravity
$\boldsymbol{w}_o$	: offset on wrench measurement

Kinematics

f	: force component
$\boldsymbol{f}$	: force vector
$\boldsymbol{X}$	: pose, containing a position and an orientation
$\boldsymbol{R}$	: rotation matrix
ω	: rotational velocity component
$\boldsymbol{\omega}$	: rotational velocity vector
$\boldsymbol{s}$	: screw vector
$\boldsymbol{S}$	: screw transformation, transforming both reference frame and reference point

$\boldsymbol{P}$	: screw projection matrix, transforming refrence frame
$\boldsymbol{M}$	: reference point transformation matrix
τ	: torque component
$\boldsymbol{\tau}$	: torque vector
v	: translational velcity component
$\boldsymbol{v}$	: translational velocity vector
$\boldsymbol{t}$	: twist vector (translational and rotational velocity)
$\boldsymbol{T}$	: set of vectors spanning the twist space
$\mathcal{T}$	: twist space
$\boldsymbol{w}$	: wrench vector (force and torque)
$\boldsymbol{W}$	: set of vectors spanning the wrench space
$\mathcal{W}$	: wrench space
$\boldsymbol{C}$	: compliance matrix
$\boldsymbol{K}$	: stiffness matrix
$\boldsymbol{D}$	: viscous damping matrix
$\boldsymbol{M}$	: inertia matrix

Estimation

$\boldsymbol{\Theta}$	: parameter
$\boldsymbol{\theta}$	: parameter value
$\mathcal{CF}_k$	: state at time step k
j	: state value
$\boldsymbol{Z}$	: measurement
$\boldsymbol{z}$	: measurement value
$\boldsymbol{r}$	: residue vector
d	: distance
$\boldsymbol{d}$	: distance vector

Number

n	: number of setpoints on compliant path
m	: number of contact formations
p	: total number of possible elementary contacts between two objects
r	: number of elementary contacts at a contact formation
s	: number of twist degrees of freedom
v	: number of visible LED markers

Table of contents

List of Figures

List of Tables

Chapter 1

Introduction

In the fifties, it was predicted that in 5 years robots would be everywhere. In the sixties, it was predicted that in 10 years robots would be everywhere. In the seventies, it was predicted that in 20 years robots would be everywhere. In the eighties, it was predicted that in 40 years robots would be everywhere...

Marvin Minsky

1.1 Motivation

Industrial robots have been successfully used in manufacturing settings as flexible and re-programmable positioning devices. Robots save costs, increase productivity, raise quality, or are used for dangerous and laborious work. The total worldwide stock of operational industrial robots at the end of 2004 reached 848,000 units. Industrial robots mainly execute relatively simple position-based tasks such as spot welding and material handling. These tasks are executed in very structured environments, where all obstacles are

known and all workpieces are in their exact position and orientation, as assumed by the operator that programmed the task. This is required because the robot operates blindly in its environment, only using its joint encoders for precise positioning, and simple binary sensors to be notified when a workpiece is positioned in its fixture. Figure 1.1 shows an example of an industrial robot operating in a structured environment. For each new task, the robot may need a different tool, a different fixture or an adapted environment. The high cost of a structured environment with specific fixtures and tools for each new product line limits the use of industrial robots to repetitive tasks in high volume series, common in the automotive industry.

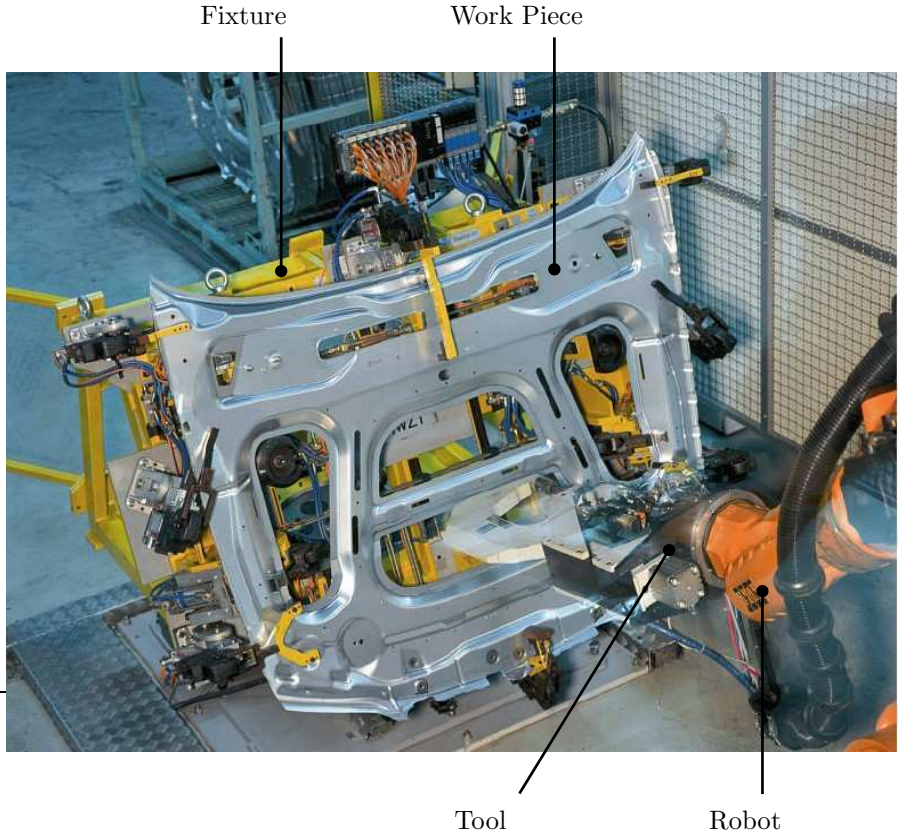


Figure 1.1: A structured environment allows an industrial robot to blindly replay a preprogrammed sequence of positions.

Robots outperform humans in speed and accuracy, but are no match for the intelligence and flexibility of humans. This research aims towards more in-

telligent and flexible robots that observe and interact with their environment. Equipped with various sensors such as a camera, a force/torque sensor or a laser distance sensor, a robot can see and feel its environment. This allows a robot to operate and perform useful tasks in an unstructured, unknown or even dynamic environment. By eliminating the need of a structured robot-specific environment, robots slowly find their way into warehouses, laboratories, exploration sites, energy plants, hospitals and even outer space. However, the level of autonomy of a robot and its ability to interact with objects in its environment still remain some of the main obstacles for a broader applicability of industrial robots in unstructured environments. This is true for most industrial applications of robots, but even more so for assembly tasks, where contact interactions with objects in the environment are part of the task.

1.2 Active compliant motion

This thesis focuses on compliant motion tasks (De Schutter and Van Brussel 1988a), such as assembly, manipulation, deburring, milling, etc. These are contact tasks where an object held by a robot manipulator is manipulated while it maintains contact with the environment. The force interaction at the contact is used to guide the manipulated object along the surface of the environmental object, to help overcome geometric uncertainties associated with the task. Almost all current industrial implementations of compliant motion tasks are still fully position controlled, and use a passive compliance between the robot and the manipulated object. With this passive compliance, a small positioning error of the robot or a small inaccuracy on the geometry of the contacting objects, does not result in high contact forces between the objects in contact; hence the passive compliance allows a robot to cope with small uncertainties on its position or on the geometry of the objects in contact. In contrast to this passive type of compliant motion, *active compliant motion* (ACM) measures and actively controls the forces and torques between the objects in contact. In other words, the system actively emulates a desired compliance between the robot and the manipulated object. Obviously, the active force/torque control allows a robot to cope with larger uncertainties on its position or on the geometry of the objects in contact. Moreover, the force/torque measurements can be used to monitor the contact task and detect possible problems in an early stage. Nevertheless, the robustness of the active compliant motion approach remains limited, compared to that of human operators, because it lacks the ability *to interpret* the cause of problems and to suggest appropriate remedies.

Figure 1.2 shows an experimental setup to verify active compliant motion strategies.

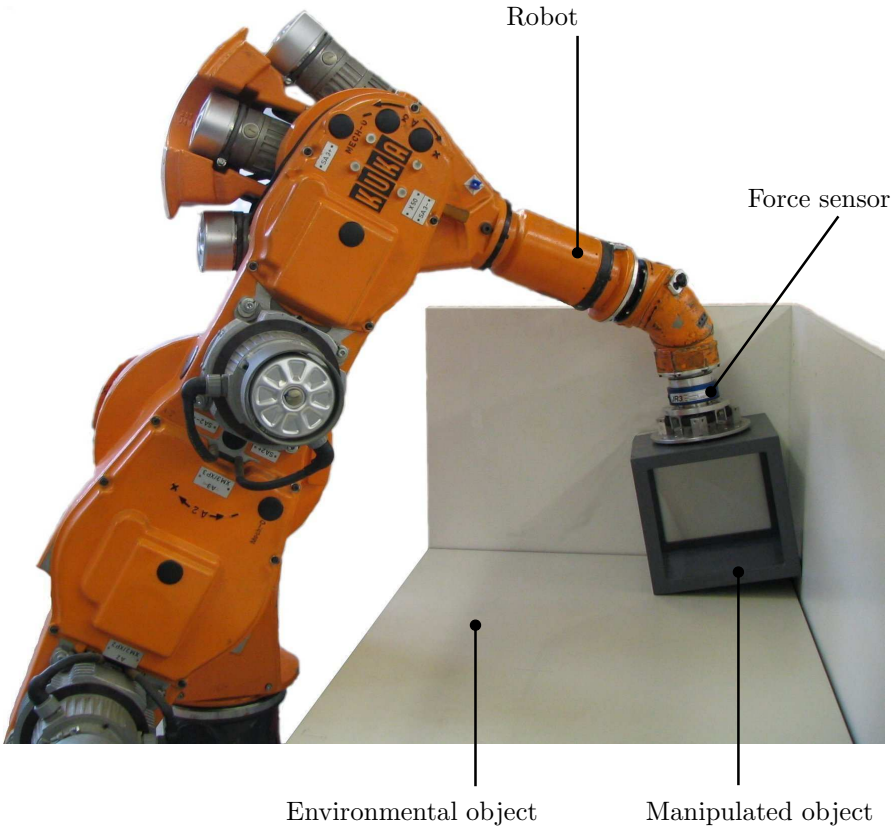


Figure 1.2: Active compliant motion: the Kuka 361 six degree of freedom industrial robot, manipulating a cube under active force control in contact with a corner.

1.2.1 Components

A general active compliant motion system consists of four main components: task specification, setpoint generation, control and estimation. Figure 1.3 gives a schematic overview of these components and the data flow between the components.

Specification The specification component allows an operator to specify a compliant motion task. Ideally the operator specifies as few details as possible, for example *put peg into hole*. A full specification is automatically generated from the operator input. Alternatively (but equally ideally!), an

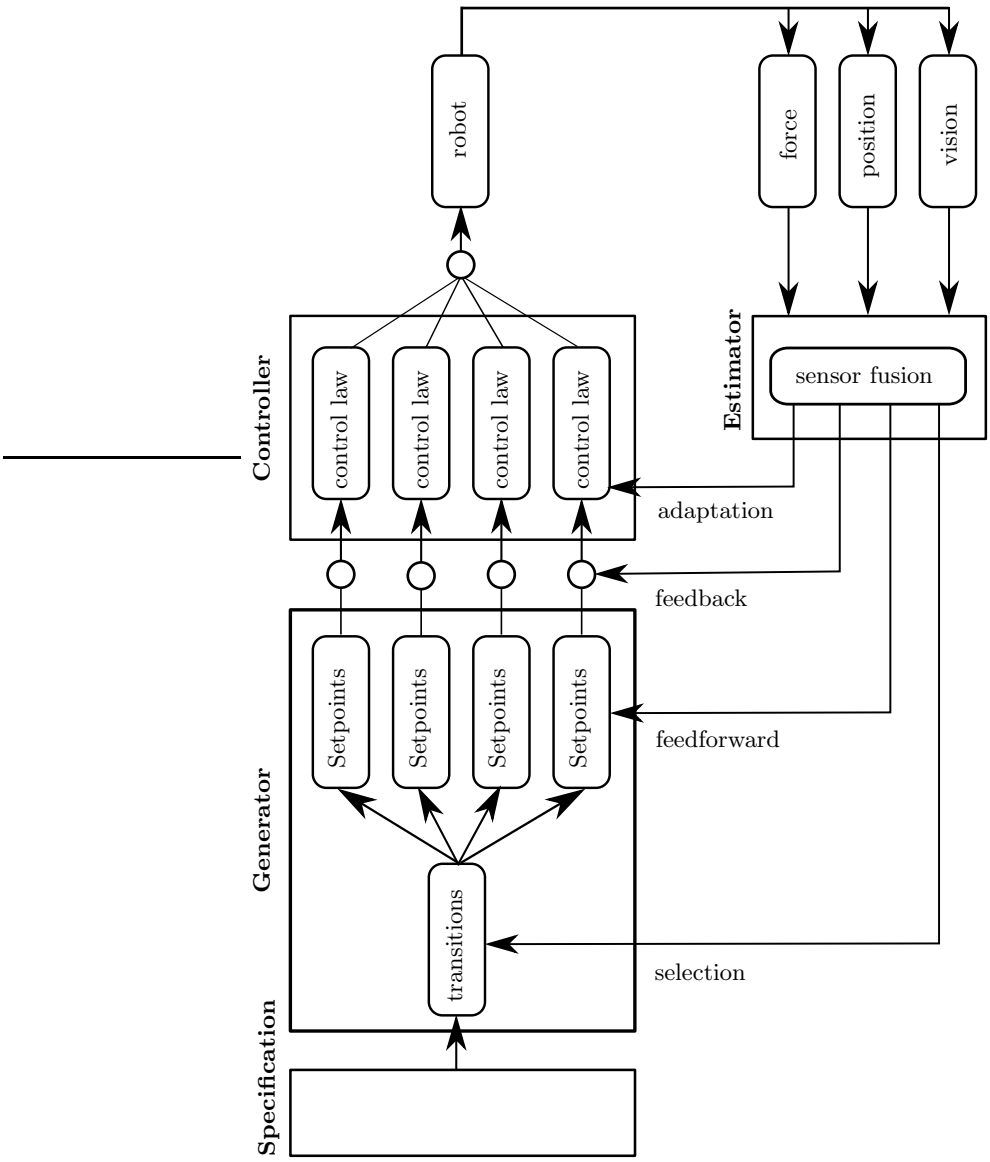


FIGURE 1.3: Overview of the control architecture for sensor based active compliant motion tasks.

operator demonstrates the desired task to obtain a task specification. In compliant motion, a task specification contains multiple sub-tasks, typically one sub-task for each contact state in the targeted task. Chapter 3 presents a compliant motion task planner that automatically generates a task specification based on the geometric models of two polyhedral objects, while Chapter 4 presents a method to obtain a task specification using programming by human demonstration.

Generator The generator component calculates instantaneous setpoints for the controller, based on a given task specification. For each sub-task of a task specification, a different *continuous* setpoint generation is used. When the estimator detects a transition to a different sub-task, a *discrete* transition to a different setpoint generator occurs. Chapter 5 presents a task generator that generates setpoints for a hybrid controller, based on a compliant motion task specification.

Estimator The estimator component combines various homogeneous and/or heterogeneous sensor measurements to estimate the probability of task specific parameters:

- the probability of the discrete parameters of the task, for example the probability on each possible discrete contact state. The generator and controller component apply this probability to select the appropriate setpoint generator and control algorithm.
- the probability of the continuous geometric parameters of the task. A better estimate of the geometric parameters decreases the uncertainty on the task model and therefore increases the overall performance of the system.
- the probability of other continuous task parameters, for example the velocity of a conveyer belt, the slip of a wheel, ... These parameters are provided to the controller component as feedback and feedforward signals.

Chapter 6 explains how the estimator based on a particle filter (presented in Chapter 4) is used to estimate the probability on discrete contact formations and continuous geometric task parameters.

Controller Controlling the motion of an object in contact with its environment requires a different control strategy for each type of contact state. A specific *continuous* control law is used for each contact state. Depending on the sensor measurements, the control law can be adapted continuously. Transitions between contact states are characterized by a *discrete* transition

between control laws. Chapter 6 presents the hybrid force/velocity controller for the force controlled execution of compliant motion tasks.

1.2.2 Work at our research group

Previous work at our research group first focused on the generator and controller components for active compliant motion tasks. In the last decade however, the research focus shifted more and more towards the task specification and estimation components.

Task specification in active compliant motion

In our research group, work on task specification in active compliant motion started with the task frame formalism (TFF), an intuitive interface to compliance and force control for the specification of force controlled robot tasks. In the TFF, the six degrees of freedom of an object in space are divided into force and velocity controlled directions along the orthogonal axes of a chosen frame, the *task frame*. Along each of the three axes of the task frame a rotational and a translational component of the object motion or interaction force is specified. The TFF relies on human intelligence and intuition to *manually* provide a task description compatible with the TFF. Current efforts are being made to create a task specification framework which is more broadly applicable and which offers better software support for the human programmer. This framework focuses not just on force controlled robot tasks, but on the integration of various other sensors into the robot controller.

The approach of programming by human demonstration (PbD) was first applied in a threshold-based system to *automatically* generate a task description. At that time the new task specification framework did not exist, nor were stochastic methods applied in compliant motion. This limited the PbD approach to ad-hoc methods and compliant motion tasks compatible with the TFF.

In close collaboration with Jing Xiao and her research group at the University of North Carolina at Charlotte, our group came in contact with task level programming methods for compliant motion tasks. The task level programming method is based on a compliant motion planner that converts a task level command (for example *put peg into hole*) into a compliant motion path through a sequence of contact states. The path description however, is not compatible with any task specification framework, and therefore it was not possible to directly execute a task specified with the task level programming method.

Coping with uncertainty

Once a robot manipulator operates outside an expensive, structured and well known environment which is adapted to the robot, it is confronted with both uncertain *geometry* and uncertain *positions* of the manipulated and environmental objects. As a result the *state* of the robot in its environment becomes uncertain. Therefore the robot is equipped with sensors, so it can observe its environment and learn about the uncertain geometrical parameters of the chosen environmental model and know its state. In the domain of active compliant motion, our research group applied Bayesian probability techniques for the integration of both estimation of geometrical parameters and state recognition. Over time, ever more advanced techniques were used to tackle this highly nonlinear estimation problem, starting with the extended Kalman filter (EKF), and then evolving towards the non-minimal state Kalman filter (NMSKF). Because Kalman filter approaches do not scale to multiple discrete states, particle filters (PF) were used in the next step. A particle filter can estimate both the geometrical parameters and the discrete state simultaneously in one single hybrid model. Every step in the process from EKF to NMSKF and finally to PF, allowed a larger uncertainty on the geometrical parameter, and more possible discrete states. However, the capability to cope with *all* possible discrete states, or the capability to apply the filter online during an execution with the robot manipulator, was not reached yet.

Open software framework

Our research group invested and still invests much time and effort into a modular and architecture-independent software framework for robot control, Open RObot COntrol Software or Orocos (www.orocos.org). Orocos includes three decoupled but integrated sub-projects:

- the open realtime control services, a hard realtime software framework for all possible machine control applications, fully independent of the project's original robotics focus,
- the kinematics and dynamics library (KDL), supporting different kinematic families, and
- the Bayesian filtering library (BFL), containing support for different filters (in particular Sequential Monte Carlo methods and Kalman filters, but also for example grid based methods) and easily extendible towards other Bayesian methods.

These components help a researcher to focus on the design of a robot application, rather than on the implementation details, and allow fast and efficient design of experiments. All components of Orocos are available as open source software under the LGPL license.

1.3 Applications

Broaden field of application In many manipulation tasks, from industrial environments to household environments, contacts between the manipulated object and the environment cannot be avoided; the contacts are inherently a part of the task. Due to the complexity of contact manipulation tasks, only few contact tasks have been successfully automated. Rather than aiming at fully automating one specific practical application, this thesis continues on the ambitious path followed by our research group to build solid foundations for a more robust and generally applicable approach in ACM, and in this way broaden the field of possible applications:

- This thesis presents an approach that allows *all possible contact states* between two rigid polyhedral objects. The presented work is not limited by the number of simultaneous contacts nor by the total number of possible contacts, and therefore envisions more potential applications than previously presented approaches. Many household tasks such as putting a book on a shelf in between other books, or putting a drawer in its place, require multiple simultaneous contacts to complete the task successfully. Also in the industry, many applications require multiple contacts, such as the assembly of a cell phone battery inside a cell phone, or piling up boxes against a wall. The presented approach did not focus on a single application, but instead is generally applicable; therefore it can be used to implement all these applications, although each application could pose some specific challenges such as high friction or hyperstatic contact situations.

- This thesis also allows *geometric uncertainty* on both the manipulated object and the environmental object. Therefore it is not needed to calibrate the pose of the manipulated object and the environmental object prior to the execution of the task. The absence of a calibration phase makes it economically more interesting to automate assembly lines for very limited series, which cannot afford a calibration procedure for each new product or product line.

- The presented *high level task specification* methods in this work hide the complexity of the task for the robot operator. This makes the task specification process more accessible to non-technical users, and makes it possible to specify contact tasks that are too complex to be specified using low level manual specification methods.

These improvements that broaden the field of possible applications are implemented and verified in an experiment where a cube is manipulated in contact with the three perpendicular faces of a corner. The experiment includes a

high-dimensional estimation problem with uncertainty on many geometrical parameters and many hundreds of possible contact states linked by the discrete *topological information* in a contact state graph. Force and position measurements provide feedback about the state of the task execution. The experiment is a proof of concept that is it possible to successfully solve such a complex problem in *realtime*.

Sensors for application This thesis uses two types of sensors: position sensors such as the Krypton camera system or the robot joint encoders, and a force sensor between the manipulated object and the manipulator. For the presented application these force and position measurements provide enough information to estimate all unknown geometrical parameters and distinguish between hundreds of contact states. Moreover, force and position measurements are expressed by only a few parameters, making the sensor processing very fast. The extension of the approach with optical sensors such as a camera or a laser distance sensor is a logical but challenging next step. However for some applications optical sensors are not functional, and then force and position sensors are an obvious choice. Optical sensors cannot be used in task such as underwater tasks in cloudy water, or tasks in a dusty environment. In some tasks, such as welding, the process would "blind" the optical sensors, rendering them useless.

Related application fields The proof of concept presented in this thesis shows that it is possible to solve a complex *high-dimensional* estimation problem with uncertainty on many geometrical parameters and many hundreds of possible contact states linked by the discrete *topological information* in a contact state graph, in *realtime*. This estimation problem is already solved for fields with lower-dimensional estimation problems, such as the simultaneous localization and mapping (SLAM) for mobile robots, which consists of many independent low-dimensional estimation problems. Many other fields of research are still challenged with the same complex high-dimensional estimation problem where discrete topological information about the task is available. An example of this is the motion capturing of the human body, where a camera tracks multiple markers on the human body. Similar to associating the force measurements with a contact state, the marker measurements are associated with the different parts of the human body. Another example is found in vision-based model building. In this field the visual features detected with a camera, are associated with the geometric features (vertex, edge, face, ...) of the geometric model of an object. Therefore the achievement of this thesis, which is the approach to solve a complex high-dimensional estimation problem in realtime, based on topological information about the task, is also relevant for many other fields of research.

1.4 Contributions of this thesis

The work in this thesis presents contributions to the state of the art in the task specification component, the generator component and the estimation component, as shown in Figure 1.3. The contributions are categorized in four major parts:

- The contact state graph, on which the compliant motion planner is based, considers free moving polyhedral objects in contact. In a real world application however, a robot manipulates one of the objects, and therefore limits the motion degrees of freedom of the object. In this thesis the contact state graph is revised to incorporate the constraints of a robot manipulator. This ensures that all contact states in the compliant path are feasible for the manipulator holding the manipulated object.

- This thesis presents a stochastic programming by human demonstration approach, based on sequential Monte Carlo methods or particle filters. Starting from the particle filter approach developed at our department, this thesis generalizes and scales this previously presented approach for simultaneous estimation of contact states and contact state recognition, to cope with *all possible contacts* between two polyhedral objects. To cope with this increased complexity, a more accurate prediction step is used, based on the *topological information* contained in a contact state graph, and the pose of the contacting objects. This thesis also presents efficient algorithms for the pose and consistency measurement equations, allowing the estimators to be used in *realtime*.

- An approach to automatically convert the output of the compliant motion planner or the human demonstration, into a task specification for a hybrid controller is presented in this thesis. This automatic conversion allows an off-line planned or demonstrated compliant motion task to be executed immediately under active force control.

- This thesis uses the presented Bayesian estimators to monitor discrete contact state transitions online, during the execution of an active compliant motion task. A discrete contact state transitions indicates the transition to a different sub-task of the compliant motion task, called a state transition. The information about the state transitions is provided to the controller component, allowing it to recognize and select the optimal control law for the given state.

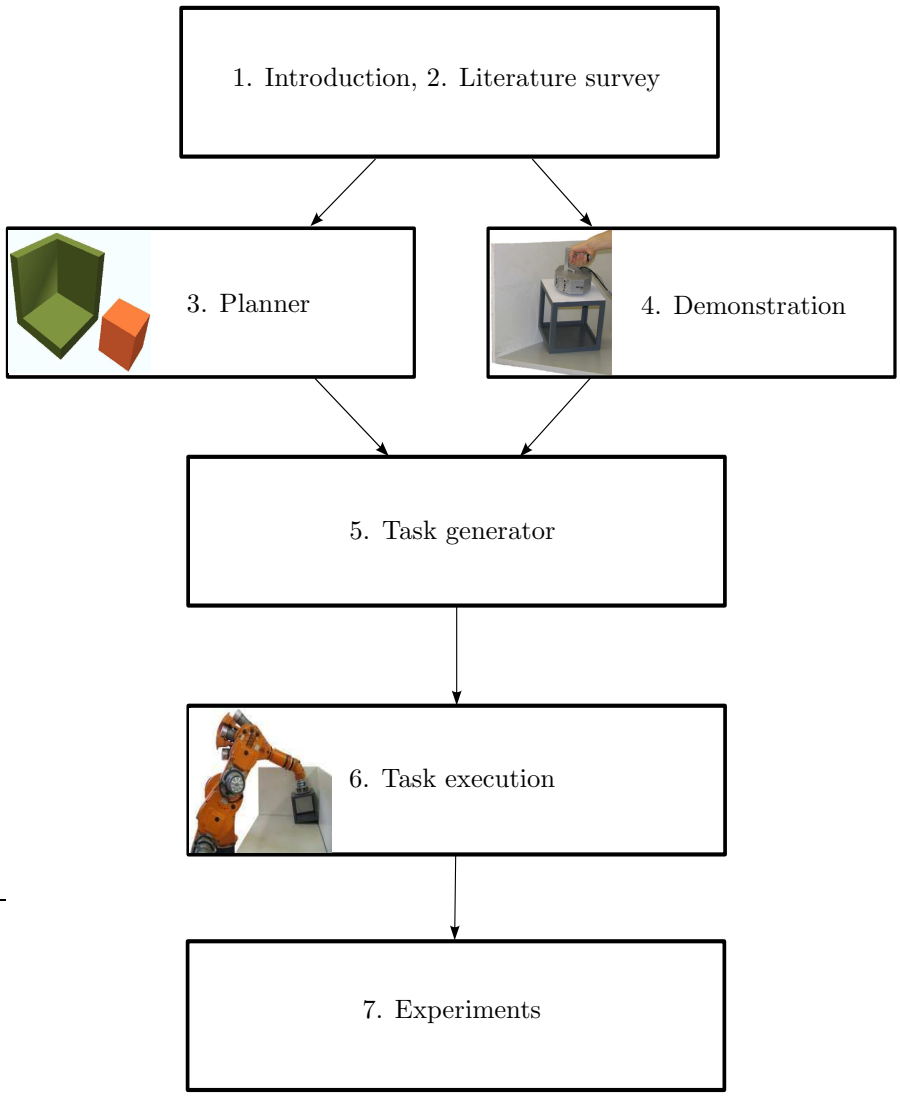


Figure 1.4: Overview of the chapters in this thesis, and the relationships between the chapters.

1.5 Outline of the thesis

The thesis is organized as follows. Chapter 2 gives an overview of historic and recent research progress in robotics literature, with a focus on sensor

based robotics tasks in active compliant motion. The chapter presents various approaches for robot programming, and approaches for the interpretation of sensor measurements in compliant motion tasks. Chapter 3 first reviews the concepts of contact formations and the contact state graph. Then the integration of manipulator constraints into the contact state graph is presented. Finally the off-line compliant path planner for automatic compliant motion task specification is discussed. Chapter 4 presents an approach for task specification by human demonstration. The chapter covers the specifications of the demonstration tool which is used to collect sensor data during human demonstration in compliant motion, and the interpretation of these sensor data using Bayesian estimation techniques. The chapter concludes with the experimental demonstration results. Chapter 5 presents the conversion of a compliant path generated by the compliant planner or human demonstration, into a task specification for the hybrid controller. This hybrid controller is presented in Chapter 6, together with the online estimation of contact states. The real world experiments that validate the presented approaches are discussed in Chapter 7. Chapter 8 presents the general conclusions of this work, discusses the contribution to the state of the art, and presents topics for future research. Figure 1.4 gives an overview of all chapters and schematically shows the relationships between the chapters.

Chapter 2

Literature survey

The next best thing to being clever is being able to quote someone who is.

Mary Pettibone Poole

2.1 Introduction

This chapter gives an overview of historic and recent research progress in robotics literature, with a focus on sensor based robotics tasks in active compliant motion, such as assembly, deburring, grinding, etc. The first section gives an overview of approaches to robot programming presented in literature in the last three decades. The robot programming approaches applied in this thesis, the automatic compliant planner and human demonstration approach, are situated within this research field. The second section deals with the processing and interpretation of sensor measurements in compliant motion tasks. It gives an overview of the research efforts for the estimation of geometric task parameters, contact formation segmentation, and contact formation recognition.

2.2 Approaches to robot programming

Programming robots, the act of specifying actions for the robot to perform in order to carry out a useful task, has been a challenge from the earliest days. The first robot programming languages emerged three decades ago, in the mid 70s, to create a simple and powerful robot programming interface (Taylor 1976; Lozano-Pérez 1976a; Lieberman and Wesley 1977). Since that time, this research domain has been very active, and many different approaches to robot programming have been presented in literature. Each approach has its own scope, level and practical usability. Comprehensive surveys on the historical development of robot programming may be found in (Ránky 1985; Latombe 1991; Chen 2001). This section gives an overview of research in robot programming, organized in four approaches:

- teach methods,
- programming languages,
- task level programming, and
- human demonstration.

Both the teach methods and the human demonstration approach are based on *showing* a task, while both the programming languages and the task level programming are based on *specifying* a task, as shown in Table 2.1.

	specify task	**show task**
high level	task level programming	human demonstration
low level	programming languages	teach methods

Table 2.1: Task specification approaches.

2.2.1 Teach methods

In teach methods, an operator manually guides a robot through a sequence of positions and actions (for example opening/closing a gripper) to perform a certain task. During the teaching phase, low-level sensor signals are recorded and stored. In early systems the low-level sensor signals are the joint positions of the robot, while in later systems the Cartesian position of the robot's

end-effector (EE) can be stored. The robot later executes the same task by playing back the recorded sequence of low-level sensor signals in "open loop", with only position feedback. In walk-through teaching, the operator uses a teach pendant as shown in Figure 2.1 to control the joint space motion or Cartesian motion of the robot (Marcelo, Lin, and Lim 1999). In lead-through teaching the operator physically interacts with the robot to move it towards a desired configuration (Groover, Weiss, Nagel, and Odrey 1986). On back-drivable robots that do not require gravity compensation, manual lead-through teaching is possible, where the actuators of the robot are not powered and the operator physically moves the robot links. On robots with a high gear ratio on their links, powered lead-through teaching is required, where the force interaction with the operator is measured and used to actively control the position of the robot links. Teach methods often require the operator to move inside the workspace of the robot, and therefore the operator risks serious injury.

FIGURE 2.1: A modern teach pendant equipped with an LCD screen is used to manually move an industrial robot in joint or Cartesian space.

Teleoperation allows the operator to program a robot from outside the workspace of the robot. Using a master-slave system, the robot (the slave) is directly controlled by the low-level sensor signals of a joystick or haptic device (the master) (Raju, Verghese, and Sheridan 1989; Kato and Hirose 2000; Conti and Kathib 2006). In many cases, the kinematic design of the slave differs from the kinematic design of the master, making the control not

very intuitive and therefore the operator will need some amount of training. Using teleoperation, it is possible to perform useful tasks in hazardous environments (for example undersea mining or explosives defusion), to manipulate heavy loads, to work in outer space or even to perform surgery. During the task execution it is advisable that a supervisory algorithm analyzes the teleoperation data as an additional safety measure (Castellani et al. 2004). In some situations for example, when operating a robot in space from an earth-bound control center, the time delay between the operator's input and the received feedback about the robot's reaction poses performance and stability problems. In such situations, a virtual simulation environment is used to generate a predictive graphical or tactile feedback about the operator's actions (Noyes and Sheridan 1984; Hirzinger, Brunner, Dietrich, and Heindl 1993).

Both teaching and teleoperation rely entirely on the human operator for perception and control. The operator receives visual or tactile feedback, and generates appropriate commands to accomplish a task. These types of programming methods directly play back the recorded low-level sensor signals, hence they require no task models. As a consequence, the programmed tasks are not transferable to different robots or different environments, and are not easily adapted to execute a new task. For every new task the programming effort has to be repeated. Cartesian space teaching, stores the poses and robot configurations instead of the robot's joint angles; hence Cartesian space teaching lends itself better to transfer of a taught program to other robots with similar kinematics.

2.2.2 Programming languages

Developing a programming language for robotics applications has been a topic of debate and discussion since the earliest computer-controlled robot systems (Danthine and Geradin 1984). This led to the variety of robot programming languages available today. A comparative study of robot languages is found in (Bonner and Shin 1982; Pembeci and Hager 2002). Using a robot programming language, an operator hand-codes a program to enable a robot to execute a given task. The task is described by means of robot motion commands, desired sensor measurements, sensor events, etc. A robot program that describes a certain task is often transferable to different robots or different environments, and is easily adapted to execute a new task. The execution of the robot program is influenced by on-line sensor measurements. This approach is flexible, but it generally requires expertise and a long development time, especially for intelligent, sensor based tasks.

The various robot programming languages can be classified as on-line or off-line languages, based on the type of syntax used or based on the level of specification:

On-line and off-line

One possible criterion to classify robot programming languages is on-line programming versus off-line programming. In on-line programming, the programming system is part of the robot controller. Programming is done on the controller itself, while the robot is inactive. Most of the early commercial systems use this method. In off-line programming, the programming system is independent of the robot controller. Although this reduces the time the robot is inactive during (re-)programming, the elaborate testing and debugging of the program (caused by deviations between the real world and the simulated world in which the programming took place) is still done on-line.

Syntax

Robot programming languages can also be classified based on the type of syntax of the language. Early developments focussed on creating specialized robot programming languages, with an easy syntax using common robot-terminology. However, it became obvious that the specialized robot programming languages have to provide nearly all the capabilities of general purpose programming languages. Therefore it was more reasonable to extend a general purpose programming language such as BASIC, Pascal, C or C^{++} with robot specific capabilities.

The first languages, such as Unimation's VAL-I (Unimation Inc. 1979), resembled BASIC in style and were implemented to facilitate on-line program development. Later languages have evolved and improved, and taken lessons from more modern structured programming languages. Examples are Pascal oriented languages such as PASRO: PASCAL for robots (Blume and Jakob 1985), GMF, KAREL for FANUC robots, LM (Latombe et al. 1984), or C oriented languages such as MRROC (Zieliński 1995) and ARCL (advanced robot control library) (Corke and Kirkham 1993). More recent C^{++} oriented languages include ZERO^{++} (Pelich and Wahl 1997), and MRROC^{++} (Zieliński 2002) and OROCOS (open robot control software) (Bruyninckx 2001; Soetens 2006).

Specification level

Robot programming languages also differ in the degree of abstraction achieved in the specification. In this thesis we distinguish between three levels of abstraction: joint level, end-effector level, and object level. These three specification levels are low-level with respect to the task level specification discussed in Section 2.2.3.

- *Joint level.* In a joint level programming language, the required position and velocity of the robot is specified in terms of the position and

velocity of the robot's joints. While a textual input is possible, many programming languages at this level use a teach pendant to program a sequence of required joint positions. During the execution of the program, this sequence is played back.

- *End-effector level.* End-effector (EE) programming languages describe the position and orientation of the robot's end-effector, in textual form or obtained by teaching with a teach pendant. Many robot vendors' proprietary languages shipped today are situated at this specification level. Examples of early structured robot programming languages at this level which already incorporate sophisticated data structures are Unimation's VAL-II (Geschke, Shimano, and Spalding 1984), MRL (MERL Research Lab, Mitsubishi), BAPS (Bosch Auto Parts Specialist), and AML (Taylor, Summers, and Meyer 1982).

  Specific programming languages have been developed for compliant motion tasks. Early languages are based on Mason's *Task Frame Formalism* (TFF) (Mason 1981), which is an intuitive interface to compliance and force control for the specification of force controlled robot tasks in the *Hybrid Control Paradigm* (HCP) (Raibert and Craig 1981; Mason 1981; Fisher and Mujtaba 1992; Duffy 1990). The HCP is one of the three major force control paradigms together with *Impedance Control* (Hogan 1985; Hogan 1987) and the *Parallel Force Control* (Chiaverini and Sciavicco 1993). The HCP assumes a *geometric* interaction model. In HCP terminology, an object in contact with its environment has s degrees of freedom (DOF) which are velocity controlled, and $(6 - s)$ DOF which are force controlled. An extensive catalog of TFF models and specifications can be found in (Bruyninckx and De Schutter 1996). One of the first task frame based compliant motion programming language is COMRADE (Compliant Motion Research and Development Environment) (Van de Poel et al. 1993). However, while limited to a linear program structure (that is without *if*... *then*... *else* and similar control flow language constructs), this programming language COMRADE allows an operator to specify complex contact tasks, for example involving vision and force servoing (Baeten 2001), with only a few commands. The Task-Net (Kröger, Finkemeyer, and Wahl 2004) is a more recent and extensive implementation of the TFF. It offers a programming language with multiple states, where sensor measurements trigger state transitions (Kröger et al. 2004). Although the TFF has applications in many contact tasks, it cannot model every *contact formation* (CF), such as CFs with multiple point contacts at different faces (Bruyninckx and De Schutter 1997).

- *Object level.* In object level programming languages, the operator speci-

fies spatial relationships between objects, based on a geometrical model of the objects. The system then automatically determines the required position and orientation of the workpiece (and thus the robot end-effector) in its environment to satisfy these spatial relationships (Ambler and Popplestone 1975; Popplestone, Weiss, and Liu 1988). Examples of programming languages that implement this approach are RAPT (Ambler and Corner 1982; Popplestone 1978) and LEO-GM. Note that these languages have no automatic planning capabilities such as obstacle avoidance or path planning, but rely entirely on the insight of the operator to manually specify feasible (low-level) commands based on spatial relationships. Recently, Rutgeerts et al. (2006) presented a new constraint-based framework that allows the specification of multiple constraints on spatial relationships between the objects, constraints on the links of the robot (for example for collision avoidance) or constraints on sensor measurements originating from for example a force sensor, a camera or a laser distance sensor. All these constraints are combined and translated into instantaneous joint positions and velocities of the robot. This approach overcomes the limitations of the TFF for complex CFs, and even allows the operator to over/under specify the robot's motion.

Today, robot vendors provide specialized packages for specific fields such as laser cutting, laser welding, arc welding, bending processes, palletizing, etc. These proprietary languages have very limited capabilities to communicate with third party software, to integrate external sensor signals other than the ones foreseen by the vendor, or to extend their functionality. Therefore many research laboratories develop their own robot programming languages.

2.2.3 Task level programming

In task level planning, a robot task is specified at a higher level. The operator uses natural high level commands such as *insert* one object into another object, *grasp* this object, *weld* these parts together, *screw* one part into another, or *move* to a certain position. The operator specifies *what* to do, but not *how* to realize the task. The underlying planning system comes up with a strategy to realize the high level specification, and converts it into robot level fine motion commands for automated execution. The goal in this research field is to develop flexible manufacturing systems that allow a manufacturer to bring new product designs into production rapidly, and easily adapt systems to a new product only by providing a description of the product. Although over time a lot of progress has been made in this field, this research goal still remains out of reach. An overview of research in planning can be found in (Latombe 1999; LaValle 2006).

Since the mid 70s a topic of research has been the development of task level planning systems for mechanical assembly. Many automatic motion planning system were proposed (Ambler and Popplestone 1975; Lozano-Pérez 1976a; Taylor 1976), including IBM's AUTOPASS (Lieberman and Wesley 1977) and MIT's LAMA (Lozano-Pérez and Winston 1977). Both LAMA and AUTOPASS focus on the task-level specification of collision free paths, but neither was completed.

Around the early 80s the concept of configuration space (C-space) was introduced by Lozano-Pérez (1976b), where a robot is reduced to a single point. Robot path planning is then "simplified" to the problem of planning a path for a single point in a space that has as many dimensions as the robot has DOF. This lead to the development of general-purpose path planning algorithms (Schwarz and Sharir 1983; Halperin and Sharir 1996), but none of them are used in practice. The complexity of the C-space for higher dimensional problems motivated the development of heuristic planners (Khatib 1986; Brooks and Lozano-Pérez 1983).

In the 90s the randomized planner was introduced (Barraquand and Latombe 1991) which is able to plan motions for 6+ DOF robots (Gupta 1995; Kavraki and Latombe 1994; Koga and Latombe 1994). Later the probabilistic roadmap (PRM) approach was presented, which connects random samples into a graph (Amato et al. 1998; Latombe and Motwani 1997). Ji and Xiao (2001a) presented a PRM based motion planner for complex contact tasks involving many simultaneous contacts, based on the random sampling of CFs (Ji and Xiao 2001b). This planner has been implemented for CFs with up to three simultaneous contacts.

Although the field of *geometric* motion planning has matured in the last three decades, motion planning for non-geometric interaction with the environment, where the *dynamics* of the interaction are important (for example deformable objects, deburring, human robot interaction, ...), still remains an open problem. This is one of the reasons why motion planners are not widely used in industrial robotics systems. Another explanation may be found in the poor integration between planning and control; in an integrated system, motion planners must eventually interact with a sensor based robot controller which executes fine motion commands and reacts to sensor measurements. Some interesting techniques for the integration of planning and control have been proposed. In (Tarn 1996) a motion plan is converted into an obstacle free trajectory using a path-based parameterization, rather than a time-based one. When an obstacle is detected, the execution is interrupted. Brock and Khatib (1999) present an elastic strips framework (Quinlan and Khatib 1993) for reactive obstacle avoidance in dynamic environments. In the field of active compliant motion, where contacts with obstacles are not avoided but required, (McCarragher and Asada 1992; McCarragher and Asada 1995)

specify (in)equality constraints at elementary contacts between two objects to maintain contacts, break existing contacts and create new contacts. Applied to two-dimensional objects, this work succeeds in assembling two objects in as little as 0.5 [*sec*]. This approach however, suffers from local minima when moving between CFs. This thesis presents a fully three-dimensional geometric approach that integrates both compliant motion planning and force controlled execution of compliant motion tasks.

2.2.4 Programming by human demonstration

To obtain a task specification for a complex robot task, programming by human demonstration (PbD) exploits the human intelligence and advanced manipulation skills. In PbD a human operator simply demonstrates the desired task. During the demonstration multiple sensors record various parameters. These parameters are not always the ones that directly describe a task such as contact forces or CFs, but measurable parameters such as positions, orientations, camera images, distances or interaction force. After the demonstration, in an interpretation step, the sensor data are used to estimate the parameters of a task model that describes the task. This task description is independent of the robot used to execute the task, and is transferable to a new environment. Even non-technical users, unfamiliar with computer technology, can use PbD to program complex robot tasks. However, also PbD suffers from drawbacks: the operator must demonstrate actions that are informative about the task's skills. A task description obtained by PbD can only cope with situations encountered during the demonstration, and humans have more and better sensors and sensor processing than what can currently be achieved with computers. The previously presented teach methods also start from a demonstration of the desired task. However, teach methods do not use a task model to interpret the sensor signals, but instead directly play back the low level sensor signals.

This section gives an overview of different demonstration and data acquisition methods, approaches to deal with demonstration noise, and skill models and skill acquisition techniques, presented in literature. An important step in skill acquisition for compliant motion tasks, the segmentation of the recorded sensor data in CFs, is more generally applicable than in PbD alone. Therefore, CF segmentation is discussed separately from human demonstration, in Section 2.3 which covers both the estimation of geometrical parameters and CF segmentation.

Demonstration

In robotics literature, three major approaches for an operator to demonstrate a task can be distinguished: task demonstration in a *virtual environment*,

demonstration by guiding a *real robot*, or demonstration by observing a *human* performing the task. Some less common and rudimentary demonstration approaches also use speech or gestures (Zhang, von Collani, and Knoll 1999; Steinhage and Bergener 2000).

- *Virtual environment.* A physical demonstration in a real world can be too time-consuming and may not be mandatory. Therefore some approaches rely on a virtual reality environment to demonstrate a task that will be executed on a real robot. The virtual environment offers the ability to create a programmer-friendly environment which is not bound to any specific location, and does not pose any risk of injury to the operator. Moreover, inaccuracies of sensor measurements do not have to be taken into account. The operator uses an interface device to transfer his motion intentions to the virtual environment, such as a simple 2D mouse (Lloyd et al. 1999), a 3D or 6D haptic device (Takeo and Fukuda 1995), a tactile glove (Harima and West 1992), an ultrasonic glove (Takahashi and Ogata 1992), or even natural language voice commands (Crangle, Michalowski, and Ling 1987). The operator most often receives visual feedback from his actions on a computer screen, showing the involved objects in a virtual 3D world. On haptic devices, the visual feedback is combined with force feedback for tactile sensing.

  In a virtual environment, the state of all objects is known at all time, and does not require incremental updates based on sensor measurements. However, the limitations of PbD in a virtual environment lie in the realism of the simulated environment. To obtain realistic "sensor measurements" from the demonstration, the dynamics in the virtual environment must be the same as in the real world (Lloyd et al. 1999). The virtual environment in (Xiao, Luo, and Song 2003; Luo and Xiao 2004) provides visual and tactile feedback to the operator through a haptic device. This environment models contact interactions between rigid bodies, allowing the operator to bump, slide and align objects, while demonstrating an assembly operation. In recent research the interaction with deformable objects is studied (Luo and Xiao 2005) based on physical laws.

- *Real robot.* A task can also be demonstrated using direct interaction with the robot that will execute the task later. In this approach the demonstration and execution obviously have the same dynamics, making the demonstration sensor measurements very informative. Also, the limitations of the robot, such as joint position limits, singular position or maximum load, are automatically incorporated in the demonstration. The operator can interact with the robot through a haptic device (Fukuda, Kosuge, and Asads 1991; Xu, Yang, and Chen 1994), a joy-

stick (Wang and De Schutter 1998), or by direct physical guidance of the robot (Asada and Izumi 1989; Kazerooni 1990) using force interaction.

- *Human.* The human operator can also directly manipulate the objects involved in a task. This is a convenient and intuitive approach that lets the demonstrator perform a task in a natural setting. Information about the task is gathered by sensors fixed in the environment and/or sensors mounted on a special demonstration tool attached to the manipulated object. Vision based system are very popular to track the position and orientation of the manipulated object. In (Dillmann, Kaiser, and Ude 1995) two cameras track the position of the manipulated object in space, in (Kuniyoshi, Inaba, and Inoue 1994) and (González-Linares et al. 1999) a camera tracks the motions of the operator's hand, and in (Knoop et al. 2006) and (Knoop, Vacek, and Dillman 2006) camera images are combined with an articulated body model to track the motion of a human body. Meeussen et al. (2006b) use a three-camera system to track the position of LED markers on a specially designed demonstration tool. Because resolution limitations of a camera make it difficult to extract a fine motion plan, Miura and Ikeuchi (1998) use a laser range finder to achieve adequate resolution. Sensors mounted on the demonstration tool ofter measure interaction forces and torques between the manipulated object and its environment (Delson and West 1993; Breidenbach, Koeppe, and Hirzinger 1996; Rutgeerts, Slaets, Schillebeeckx, Meeussen, Stallaert, Princen, Lefebvre, Bruyninckx, and De Schutter 2005). With a specific focus on programming robots for household tasks, Dillmann (2004) acquires as much information as needed to learn the performed tasks. He combines information from a camera that can be turned and tilted together with information from data gloves. The data gloves are equipped magnetic field trackers, tactile sensors and sensors that allow to acquire precise data about the finger joint angle in order to classify specific grasps. Typical household actions such as opening a door and filling a dishwasher are demonstrated and later executed by a robot (Ehrenmann, Zöllner, Rogalla, Vacek, and Dillmann 2002).

Demonstration noise

PbD will seldom result in a task execution that is optimal in terms of speed or energy consumption of the robot. Many publications identify that a demonstration often contains *demonstration noise*. This demonstration noise is caused by both sensor noise and operator noise. Different sources of operator noise in a human demonstration are recognized in (Kaiser, Friedrich, and Dillmann 1995): *unnecessary* actions that do not contribute to achieving the goal, *incorrect* actions, *unrecognizable* actions to the learning system, or

wrong intention when the operator does not know enough about the task he demonstrates.

Different approaches exist to identify and remove demonstration noise. Wang and De Schutter (1998) apply filtering techniques to the low level sensor signals, based on zero thresholds, magnitude thresholds and area thresholds. Stochastic based approaches use Kalman filters (KFs) (Kalman 1960; Sorenson 1985) to take into account sensor noise and system dynamics when filtering low level sensor signals (Slaets et al. 2006; Meeussen et al. 2006b). Many approaches combine the information in different demonstrations to identify operator noise (Delson and West 1996; Kaiser and Dillmann 1996; Eberman and Salisbury 1994; Eberman 1997). Chen (2005) identifies two levels of demonstration noise: noise on the (low) level continuous trajectory, and noise on (high) level discrete transitions between CFs. From multiple demonstrations of a paper towel holder into its support they generate a directed contact state graph. A graph search then finds a –for the robot– efficient high level contact state trajectory based on a combination of different cost functions.

Skill model

Humans can quickly and efficiently demonstrate complex manipulation tasks using their fine manipulation skills, physical insight and experience. Humans have acquired these sophisticated manipulation skills through training over a long time period. PbD aims to exploit this expertise of the human to learn strategies that lead to a skilled execution of the desired task. However, humans are not aware of the exact nature of their advanced and complex skills. Therefore it is not easy to interpret a demonstrated task and find an appropriate strategy as well as to translate it into a set of robot commands. This section discusses *which* skill models for compliant motion tasks were presented in literature. *How* the parameters of these skill models are determined is discussed in Section 2.3 which discusses estimation and CF segmentation. Many of the skill models are also used in teleoperation. This research field is closely related to PbD, and faced with similar challenges. In teleoperation, a human "demonstrates" a task, and this task is immediately executed by a robot in a different environment. From the demonstration, different skill models are derived and sub-tasks are identified, to select an appropriate control strategy for the execution of the task by the robot (Castellani, Botturi, and Fiorini 2004; Ekvall, Aarno, and Kragic 2006). In teleoperation however, the human is still included in the feedback loop to the robot and provides the "intelligence" of the system, while in PbD all the task parameters and intelligence is learned from the demonstration of the task.

Neural networks have been a popular tool to create skill models that directly map sensor measurements to robot velocities (Kaiser and Dillmann 1996; Nuttin and Van Brussel 1996). These "black box" skill models avoid

the complex modelling of CFs and contact forces in assembly tasks, but have low physical foundation and the trained network is difficult to interpret afterwards and hence to transfer to other situations. Asada (1990) uses a multi layered neural network to learn non-linear compliance strategies. The strategy is verified on a peg-in-hole assembly task. Nuttin et al. (1995) performs a peg-in-hole experiment in simulation, to verify a learning controller that is trained in two phases: a knowledge acquisition phase and a reinforcement learning phase; he also applies the learning to a real force-controlled assembly task with a very complex geometry.

An alternative approach which also directly maps sensor measurements to robot velocities, is based on a *fuzzy logic* skill model (Dillmann, Kaiser, and Ude 1995). The forces and velocities, measured during the demonstration phase, are divided into discrete fuzzy groups. From the human demonstration, the approach extracts the fuzzy rules that link a measurement group to an action group, for example a high force along an axis can be linked to a high robot velocity in the opposite direction along the same axis. The fuzzy rules are created using cluster detection in the force-velocity space.

Early skill models with a physical foundation are based on the *hybrid control paradigm* implemented by the TFF, which allows a robot to conform to geometrical constraints. The control mode (force or velocity) along each axis of the task frame, together with the desired force or velocity controlled values, are learned from teaching data. This hybrid skill model was applied in (Asada and Izumi 1989) on a 3 DOF end-effector motion when assembling a cube into a corner. Most publications focussing on contour following tasks (see Figure 2.2) use the contact normal as a first order task model (Yoshikawa and Sudou 1993; De Schutter and Van Brussel 1988b), some use the contact point (Tsujimura and Yabuta 1989), while Demey (1996) not only adds the contact normal, but also adds the contact curvature to the task model.

More complex compliant motion tasks include multiple simultaneous CFs and changes in CF. Therefore, recent publications combine the previously discussed continuous skill models with discrete skill models. The result is a *hybrid* (both continuous and discrete) skill model (Finkemeyer, Kröger, and Wahl 2003). The discrete part of the hybrid skill model can be discrete C-space regions, as presented in (Ogata and Takahashi 1994), but most approaches choose discrete CFs (Ikeuchi and Suehiro 1994; Wang 1999; Debus, Dupont, and Howe 2002). The segmentation of a human demonstration into CFs, results in a *sequence* of discrete states. Chen (2005) combines the information from multiple demonstrations into a directed contact state *graph*. The hybrid skill model presented in (Meeussen et al. 2006b) combines discrete CFs with reciprocal (Klein 1871; Ohwovoriole and Roth 1981; Bruyninckx, De Schutter, and Dutré 1993b) force and velocity controlled directions.

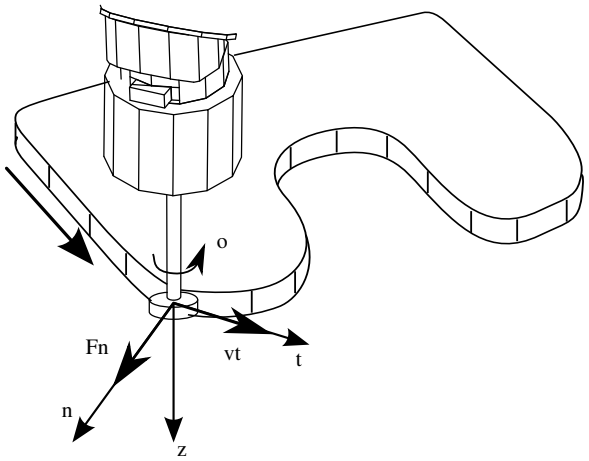


Figure 2.2: The task frame formalism applied to a contour following task. The task frame is positioned in the contact point, with one axis along the contact normal.

2.2.5 Conclusions

Robot programming methods in compliant motion aim to provide a powerful and flexible interface to specify contact tasks for robot manipulators. In unstructured or unknown environments, where a robot uses multiple sensors to obtain feedback from its actions, the process of task specification becomes increasingly complex. An easy interface to specify complex sensor based tasks would allow the use of robots for smaller product series in unstructured industrial environments, or could even help to introduce robots in households to assist humans in various tasks.

This section gives an overview of literature in the field of robot programming methods. Teaching methods and programming languages provide a low level tasks specification interface to an experienced operator. Most low level methods for compliant motion task specification are still limited to the TFF, which can only model a subset of all possible CFs. Although recently new methods have emerged that overcome the limitations of the TFF, the operator needs a thorough understanding of and insight in all aspects of the task, such as the geometry of the involved objects, the contact topology of the task, the kinematics and dynamics of the robot manipulator, etc. This thesis however, aims towards more accessible and easy-to-use high level programming methods, that do not require a lot of training and insight in the task.

Both task level programming and programming by human demonstration provide an intuitive and user friendly high level task specification interface

to an operator. While task level programming uses intelligent path planners, programming by human demonstration exploits the advanced human manipulation skills. However, over the last decades, these high level methods did not mature enough to meet industrial standards, and therefore typical industrial applications still mainly rely on low level task specification interfaces. The broader adoption of high level robot programming methods is prevented by a number of missing building blocks in the high level methods. One of the key missing building blocks is lack of integration between the high level methods and the low level controller. There is still a "gap" between the output of a high level robot programming method and the input to a low level robot controller. This lack of integration motivates the approach presented in this thesis to integrate both high level planning and low level control. The approach is applied to both the output of a task level compliant path planner and the output of a compliant human demonstration. While most authors only consider 2-3 DOF motion, with few possible contacts between the object manipulated by the robot and the environmental object, this thesis considers the full 3D space with all possible contacts between the involved polyhedral objects.

2.3 Estimation and contact formation segmentation

Compliant motion tasks are often performed in the presence of geometric uncertainty, for example an assembly operation executed in an unstructured or dynamic environment. The force interaction that arises from the contacts between the manipulated object and the environmental object is used to overcome the positioning uncertainty between the objects. The contact force guides the manipulated object along the surface of the environmental object. Passive, position controlled compliant motion, with a passive compliance between the manipulator (a robot manipulator or a human demonstrator) and the manipulated object, can only cope with small geometrical uncertainties, limited by the maximum deformation of the passive compliance. Active compliant motion (ACM) uses sensors to observe various task parameters, most often a force sensor to measure the interaction forces between the contacting objects, but also sensors such as a camera or a laser distance sensor can be used. When the sensor information from different sensors is correctly combined and interpreted, it allows a robot to actively react on events in its environment, to actively control contact forces or to actively maintain a distance to another object. This active interaction with the environment in combination with the interpretation of sensor information makes ACM (somewhat) more robust against large geometrical uncertainties

than the passive approach. Major challenges in the interpretation of sensor information from a compliant motion task execution are: (i) to recognize the CF to which the task is currently subjected, (ii) to estimate the *geometrical parameters* of that CF (that is the position of contact point(s), the direction of contact normal(s), etc.), and (iii) to detect when exactly the transition between two CFs occurs.

2.3.1 Estimation of geometrical parameters

Early work in the interpretation of sensor information from a compliant motion task execution assumes the topology of the contact, that is the CF, to be known. This reduces the estimation problem in compliant motion to only the estimation of continuous geometrical parameters. A popular application in much of the presented work is 2D contour tracking, where only one single vertex-face contact occurs.

Most approaches only use instantaneous sensor measurements, and apply ad-hoc, non-stochastic algorithms. Tsujimura and Yabuta (1989) only use wrench measurements to estimate the contact point, as the intersection point between the wrench screw vector and the probe. A similar approach in (Yoshikawa and Sudou 1993) estimates the contact normal as the vector perpendicular to the measured twist. To remove the friction component of the wrench screw, the measured wrench is projected onto the measured twist. De Schutter (1988) improves force controlled tracking by feeding forward the object motion parameters such as velocity and acceleration, in the force control law. Baeten (2001) presents a vision based contour tracking. This work combines both camera images, wrench measurements and twist measurements, to estimate the contact normal of an unknown contour. From the camera images the position of the contact point and the normal on the contour are extracted, and applied as a feed-forward to the controller.

More recent work is based on stochastic methods, which take into account uncertainty on the sensor measurements. The uncertainty consists of sensor noise and modelling uncertainty such as inaccuracy of position measurements due to elasticity at the contact, or inaccuracy of the wrench measurements due to contact friction. Fedele et al. (1993) apply a recursive-least-square method to contour following, based on position and force measurements. In (Mihaylova et al. 2002) an interacting multiple model (IMM) filter is implemented, where several iterated extended Kalman filters (IEKFs) (Tanizaki 1996) run in parallel. The contour curvature and angle estimate are the fusion from the estimates obtained from the separate EKFs. Lefebvre, Bruyninckx, and De Schutter (2005) use a non-minimal state Kalman filter (NMSKF) to identify the contour shape from a known set of shapes, and find the location of the contact point.

2.3.2 Contact formation segmentation

Even between two simple polyhedral objects, *hundreds* of CFs are possible (Xiao and Ji 2001), and, hence, hundreds of transitions between neighboring CFs. Recognizing the current CF and detecting when exactly the execution changes between two (discrete) CFs based on only continuous sensor measurements, has received a lot of attention from the active compliant motion research community. A comprehensive survey on CF recognition and CF transition detection may be found in (Skubic 1997).

Initial research on the identification of CFs mainly focused on ad hoc strategies that exploit geometrical knowledge of the contacting objects, but that have a poor stochastic foundation. Wang et al. (1996) apply filtering and thresholding techniques on human demonstration data (Wang 1999), while McCarragher and Asada (1993) use qualitative template matching in dynamic process models for state transition recognition. Both methods are sensitive for the chosen threshold values. In (McCarragher and Asada 1993; Dupont et al. 1999) rigid body dynamics are used to detect 2D CFs, using qualitative template matching of every measured force component. The approach needs threshold values for large, small or zero forces. Many force/torque based approaches were presented, such as (Desai and Volz 1989; Kitagaki, Ogasawara, and Suehiro 1993; Bicchi, Salisbury, and Brock 1993) or (Hirai and Asada 1990) where force information is interpreted using convex friction cones to indicate the unidirectional range of possible forces. The methods presented in (McCarragher and Asada 1993; Sikka and McCarragher 1996) observe the derivative of the measured contact forces to detect CF transitions, while Hovland and McCarragher (1996) observe the frequency content of the force signal. Dutré, Bruyninckx, and De Schutter (1996) use a systematic model-based approach to detect CF transitions. The CF monitoring is based on energy considerations and thus invariant with respect to changes in the mathematical representations.

Learned CF segmentation based on neural networks is applied in (Simons et al. 1982) and (Asada 1993; Nuttin and Van Brussel 1996). This approach produces good results, but the use of a neural network hides the use of prior information about the task. Hara and Yokogawa (1992) and Skubic and Volz (1996) use a fuzzy classifier where membership functions are obtained from training data. Skubic and Volz (2000) combines both approaches: first fuzzy sets are used to model patterns and sensor uncertainty membership functions, and second a neural network is used to generate confidence levels for each CF. In (Sikka and McCarragher 1996) discriminant functions are applied in combination with clustering techniques. The discriminant functions are learned from sensory data. Also stochastic methods based on hidden Markov models (HMMs) (Hannaford and Lee 1991; Eberman 1997; McCarragher and Hovland 1998; Hovland and McCarragher 1998a; Hovland and McCarragher

1998b; Ekvall, Aarno, and Kragic 2006) use learning techniques. The HMMs represent a stochastic, knowledge based system for discrete events, where the models are trained off-line with experimental data. HMM-based approaches require multiple datasets and are often applied in a 2D experimental setup, where they achieve accurate and very fast CF recognition. Recently (Castellani et al. 2004) applied HMMs in combination with *support vector machines* (SVMs) for the segmentation of a full three dimensional peg-in-hole based on force/torque measurements. The segmentation is performed online during a teleoperation task, and applied to select an appropriate control strategy for each of the recognized sub-tasks.

Farahat, Graves, and Trinkle (1995) *hypothesize* a CF, represented as a collection of elementary contacts (ECs). The feasibility of a hypothesized CF is tested using linear programming and force sensing. If more than one CF is feasible they use a geometric interpretation to determine the likelihood of each feasible CF. In (Xiao and Zhang 1996) the technique of growing a polyhedral object by its position uncertainty is applied to hypothesize CFs. The intersection of the grown regions obtains the set of all possible topological CFs due to the position uncertainties, reduced to the geometrically valid CFs. The precise CF can then be extracted by additional sensing such as force/torque sensing.

2.3.3 Simultaneous contact formation segmentation and geometrical parameter estimation

In a compliant motion task different contact constraints apply at each *discrete* CF. These contact constraints link the sensor measurements to the *continuous* geometrical parameters (for example the position and orientation of the manipulated object or the dimensions of the involved objects) of the task. In other words, the discrete CF is the model that links the sensor measurements to the continuous geometrical parameters. Therefore, to estimate uncertain geometrical parameters of the objects involved in a compliant motion task, the knowledge of the current CF model is required. On the other hand, the detection of CF transitions and the recognition of CFs improve when sensor measurements increase the knowledge about the geometrical parameters of the system. This shows how the estimation of continuous geometrical parameters and discrete CF recognition are two connected sub-problems of a single *hybrid state space* estimation problem. Both sub-problems are best solved simultaneously.

Mimura and Funahashi (1994) recognize vertex-face, edge-face and face-face contacts, based on twist, wrench and pose measurements. They first assume a vertex-face CF, and then build extra contact constraints which are tested from least to most constrained CF. Simultaneously unknown param-

eters such as contact positions and contact directions are estimated. Active force sensing is applied to distinguish different CFs.

Our research group has pioneered in applying state-of-the-art Bayesian probability techniques (in the domain of force-controlled compliant motion) for the integration of both estimation of geometrical parameters and CF segmentation. Although approaches based on HMMs also have a stochastic foundation, they only perform CF segmentation without the estimation of geometrical parameters. HMM based approaches also require multiple datasets for training purposes, while the presented work of our research group uses a *one shot* estimation based on a single dataset (Dillmann 2004). Our work applies to general CFs, not limited by the "orthogonal" contact models of (Mason 1981), and is based on pose, twist and wrench measurements in full 3D (6 DOF) experiments. De Schutter et al. (1999) present a recursive state estimation framework using Kalman filter (KF) techniques (Kalman 1960; Sorenson 1985). The sequence of CFs in the presented cube-in-corner experiment is assumed to be known. A summed normalized innovation squared (SNIS) value (Bar-Shalom and Li 1993) monitors the consistency between sensor measurements and the current contact model. A high SNIS value indicates an inconsistency and hence a CF transition to the next CF in the known sequence. Lefebvre et al. (2003) extended this approach using the non-minimal state Kalman filter (NMSKF), which allows them to cope with larger geometric uncertainties. Like all KF based approaches, the NMSKF also requires one filter per CF, which makes it hard to scale it to all possible CFs between contacting objects. Slaets et al. (2006) uses the same NMSKF to *build* a geometrical model of an unknown environment, from vertex and face primitives, and simultaneously recognizing the CF transitions from an unknown sequence of CFs. Their approach is valid if the estimation has converged to a unimodal Gaussian before each CF transition. They only allow new contact constraints to be added gradually (one vertex-face contact at a time), and no contact constraints to be removed.

Debus, Dupont, and Howe (2004) use deterministic multiple model estimation based on only pose measurements. The sequence of CFs is known in advance, but not when the CF transitions occur. They describe a CF as a parameterized position-based constraint equation, where the constraints are the distances at the contacts. The constraint error (that is the distance at the contacts) is presented to a hidden Markov model (HMM) for CF segmentation. The approach is applied to a peg in hole experiment with 3 DOF.

Recently Gadeyne et al. (2005) developed a sequential Monte Carlo or *particle filter* (PF) approach (Doucet, Gordon, and Krishnamurthy 2001) based on pose, twist and wrench measurements. Based on a *hybrid* (partly continuous, partly discrete) model, they estimate (continuous) geometrical parameters with a large initial uncertainty, and simultaneously recognize (discrete)

CF transitions in an experiment consisting of six initially known possible CFs. The low number of possible CFs allows them to use a simple prediction step in the PF, in combination with a small number of particles. In this thesis this PF approach is scaled and generalized to cope with all possible CFs between two polyhedral objects, in realtime. In contrast to approaches based on the KF or KF variants, the approaches based on a particle filter only require one single filter for all CFs.

Similarity to other application fields This hybrid state space estimation problem (also called *"data association"* in that it tackles the problem of assigning every sensor measurement to the appropriately corresponding discrete model) is similar to the estimation problems in other robotics domains, that must also match sensor measurements to available discrete "maps" of primitive geometrical building blocks to estimate continuous parameters. In mobile robotics navigation, the field of *SLAM* (Simultaneous Localization and Map Building) (Davison, Cid, and Kita 2004; Thrun, Burgard, and Fox 2005) tries to recognize discrete landmarks in camera images or laser scans. Only when the correct landmark is associated with the sensor measurements, the estimation of the continuous position of the mobile robot and position of the landmark can be improved. Because the continuous positions of each of the landmarks are independent, the SLAM problem benefits from efficient methods such as the rao-blackwellized particle filter (Montemerlo and Thrun 2003; Montemerlo et al. 2003), which combines a particle filter to estimate the robot position with Kalman filters to estimate the position of each landmark. In the compliant motion context of this thesis, the dependency between the continuous parameters prevents the use of these efficient methods. Vacek et al. (2006) use a particle filter for the tracking of multiple lanes on a street, to provide visual cues to assist in GPS navigation. They recognize discrete lanes and estimate the geometry of each lane based on a finite state machine in combination with a particle filter. Schulz, Burgard, and Fox (2003) apply sample-based joint probabilistic data association filters for vision-based object tracking. Also in this field, discrete objects (for example moving objects in a camera image) should be identified in a camera image in order to estimate the continuous parameters (for example the current position of the moving object). In data association for contact state recognition, sensor measurements are linked to one single contact state at each timestep, while in the work of Schulz, Burgard, and Fox (2003) the sensor measurements are associated with a varying number of discrete objects.

2.3.4 Conclusions

In "intelligent" compliant motion tasks, the information from various sensors is combined to estimate task parameters such as the contact topology, the task's geometrical parameters, dynamical contact parameters, etc. The estimators can be used in *task specification* using programming by human demonstration, but also for *task monitoring* during execution by a robot. In task specification by human demonstration, a high level task specification in the form of a sequence of CFs is identified by the estimators. In on-line task monitoring the estimators allow the robot to switch to a different controller when a CF transition is identified, re-plan a path when an undesired CF occurs, or operate at higher speed when the knowledge about the task's geometry increases.

In the presence of uncertainty, the estimation problem consists of two subproblems: the estimation of geometric parameters and the segmentation of the task in discrete CFs. Most approaches presented in literature only solve one of these subproblems, or solve both subproblems separately. However, both subproblems are not independent and when solved simultaneously (i) the CF segmentation improves when the knowledge about the task's geometrical parameters increases, and (ii) the estimation of geometrical parameters improves when the correctness of the CF segmentation increases.

Previous research in simultaneous estimation of geometrical parameters and CF segmentation was often limited to small geometrical uncertainties, required a small and known CF sequence, or was too slow to be used in realtime. This motivates the research efforts in this thesis to extend the state of the art estimators to cope with large geometric uncertainties, all possible CFs, and still work in realtime.

2.4 Summary

This chapter gives an overview of literature in two fields important to active compliant motion: robot programming methods and estimation methods. The work presented in this thesis improves the state of the art in these fields at three points.

- The integration between high level robot programming methods such as task level planning and programming by human demonstration, and the low level controller actions. This thesis presents an approach to automatically convert the output of a high level compliant motion path planner or the human demonstration of a compliant motion task, into a low level task specification for a hybrid force/velocity robot controller. This allows a robot to immediately execute a task that is specified at a high level.

- The simultaneous estimation of geometric parameters and CF segmentation. This thesis extends existing approaches to cope with large geometric uncertainties, all possible CFs between the involved objects, and still allow realtime operation.
- The online monitoring of compliant motions executed by a robot manipulator. This thesis uses the developed estimators to monitor CFs during the execution of a compliant motion task, allowing the robot to change its control strategy based on the current CF.

Chapter 3

Compliant motion planner

In preparing for battle I have always found that plans are useless, but planning is indispensable.

Dwight D. Eisenhower

3.1 Introduction

In compliant motion tasks, contacts between the manipulated objects are inevitable and even desired to obtain a higher precision than the absolute precision of the manipulator or to reduce the uncertainty associated with a task. Hence, the knowledge of the contact states and the relation between different contact states is extremely important in planning and executing such tasks. This chapter presents all the mathematical preliminaries, contact descriptions, contact models, etc. that are required for the automatic planning of compliant motion tasks. In the first section, this chapter describes different contact state representations for autonomous compliant motion. First the different topological contact descriptions are discussed, and then the mathematical representations and transformations of the contact models are pre-

sented. The second section covers the contact state graph, a simplified graph-representation of the relation between different contact states. The contact state graph is automatically generated for two free moving objects, and manipulator constraints are added afterwards. Finally this chapter discusses how a compliant path between a start and a goal configuration is calculated by an off-line compliant motion planner, as illustrated in Figure 3.1 (top left). Real world experimental results based on the presented compliant motion planner are presented in Chapter 7.

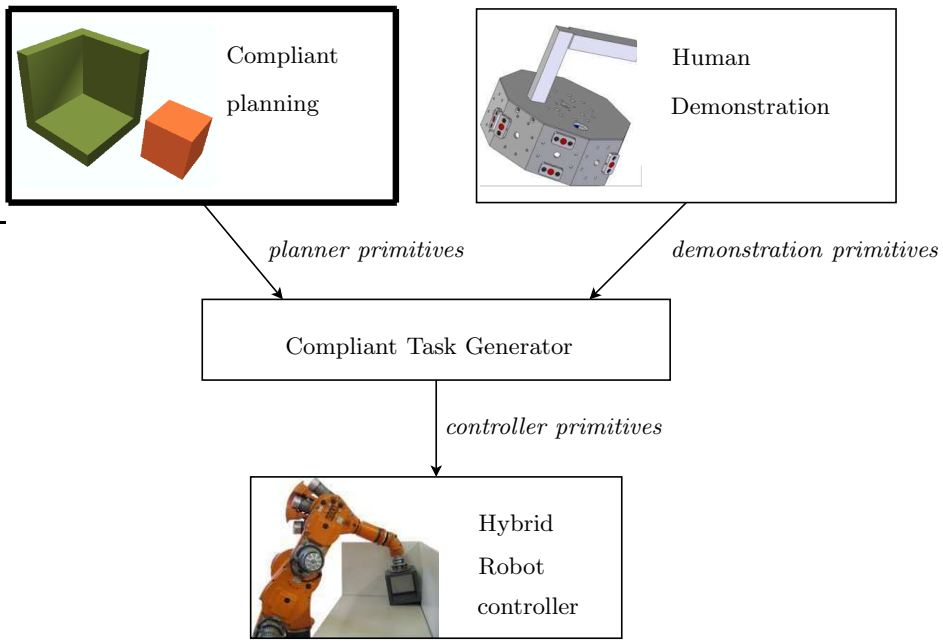


Figure 3.1: This thesis presents two high level approaches for task specification in active compliant motion: compliant path planning and programming by human demonstration. The compliant task generator converts the task specification into instantaneous setpoints for the hybrid robot controller. This section discusses the compliant path planner.

3.2 Contact modelling

When referring to a contact state between two objects, we mean a higher level representation than what the configuration of the objects describes. The contact state may remain the same while the configuration of the objects

varies, as illustrated in Figure 3.2. In the literature, different descriptions of contact states exist, and each description has its own granularity and field of application. Some descriptions provide a topological representation of the contact, while others describe the first or second order kinematics. Xiao (1993) presents a topological representation for *rigid* polyhedral objects in terms of contacting topological surface elements. For *articulated* polyhedral objects, Staffetti, Meeussen, and Xiao (2005) apply oriented matroid theory to characterize the proximity to a new contact for different configurations of an articulated object. A first order kinematic contact representation in terms of (reciprocal) twist and wrench spaces is presented in (Mason 1979). In (Bruyninckx, De Schutter, and Dutré 1993a) the contact kinematics between two contacting curved objects are represented by an equivalent kinematic chain, modelling second order kinematics.

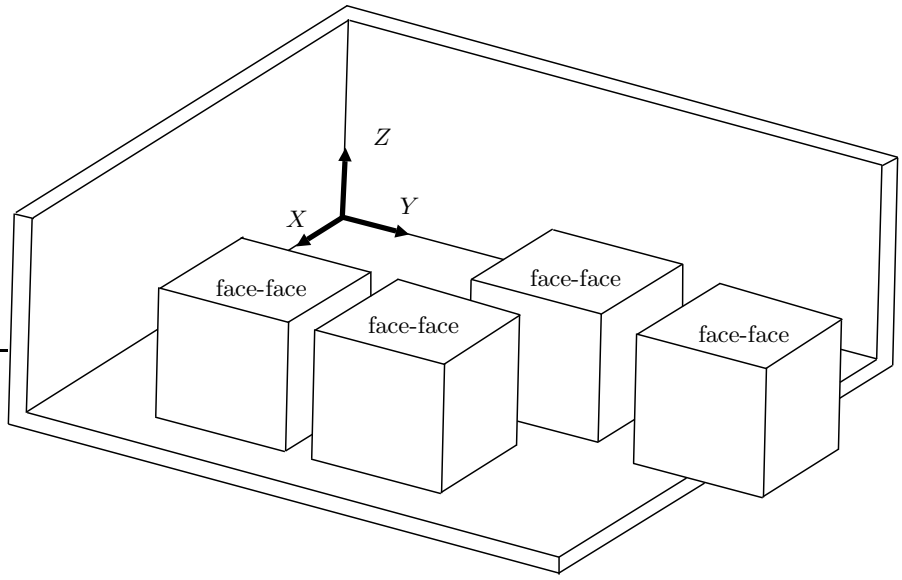


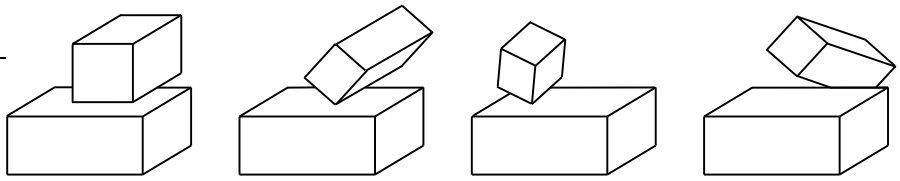
FIGURE 3.2: Different configurations of two contacting objects can have the same contact state.

In this thesis we start from a topological representation of contact states between rigid polyhedral objects, where a contact state can be described by contacts between the *surface elements* of the objects. A surface element can be a face, an edge or a vertex. The *boundary elements* of a face are the edges and vertices bounding it, and the boundary elements of an edge are the vertices bounding it. From this topological representation we then derive the first order kinematics in terms of a reciprocal twist and wrench space.

3.2.1 Contact topology

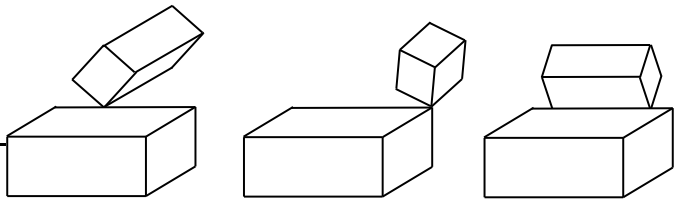
Principal contact

The notion of *Principal Contacts* (PCs) was introduced (Xiao 1993) to describe a contact primitive between two surface elements of two polyhedral objects in contact. Formally, a PC denotes the contact between a pair of surface elements which are not boundary elements of other contacting surface elements. The ten types of PCs that can be formed between two polyhedral objects are divided into six non-degenerate PCs and four degenerate PCs. Figure 3.3 shows the six non-degenerate PCs, the vertex-face, face-vertex, edge-face, face-edge, edge-edge and face-face. The degenerate PCs shown in Figure 3.4, the vertex-vertex, vertex-edge, edge-vertex and edge-edge-parallel are not considered in this thesis, as it is difficult to achieve a stable contact that includes one of these PCs. Each non-degenerate PC is associated with a *contact plane*, defined by a contacting face or the two contacting edges at an edge-edge PC.



face-face | face-edge, edge-face | face-vertex, vertex-face | edge-edge

Figure 3.3: The six possible non-degenerate principal contacts (PCs) between two polyhedral objects.

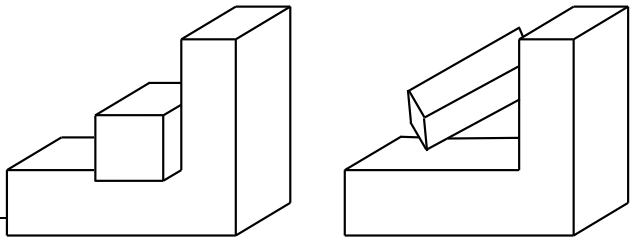


vertex-edge, edge-vertex | vertex-vertex | edge-edge parallel

Figure 3.4: The four possible degenerate principal contacts (PCs) between two polyhedral objects.

Contact formation

A general contact state between two objects can be characterized topologically by the set of PCs formed, called a *Contact Formation* (CF) (Xiao 1993; Desai 1989). Figure 3.5 shows an example of two CFs, each consisting of two PCs. Each configuration of two objects, this is their relative pose in space, compliant to the constraints of a CF, is called a *CF-compliant configuration*, denoted by a pose $\boldsymbol{X}$. Any motion formed by a sequence of CF-compliant configurations is called a *CF-compliant motion* .



2 × face-face 2 × vertex-face

FIGURE 3.5: A contact formation (CF) is characterized topologically by the set of PCs formed.

In this thesis we choose a homogeneous transformation matrix to represent the pose $\boldsymbol{X}_a^b$ of the reference frame on an object b relative to the reference frame on an object a:

$$\boldsymbol{X}_a^b = \begin{bmatrix} \boldsymbol{R}_a^b & \boldsymbol{p}_a^b \\ \boldsymbol{0} & 1 \end{bmatrix}, \tag{3.1}$$

where $\boldsymbol{p}_a^b$ represents the 1×3 position vector from the reference frame on a to the reference frame on b, and $\boldsymbol{R}_a^b$ represents a the 3×3 rotation matrix between the reference frame on a and the reference frame on b. While it is possible to represent a pose with a minimum of six elements, a homogeneous transformation matrix contains twelve elements, and is therefore called a non-minimal representation of a pose. A possible minimal representation of a pose using six elements is the combination of three position coordinates with three ZYX Euler angles. Non-minimal representations carry the extra cost of a set of constraints between the numbers in the non-minimal representation. However, minimal representations suffer from numerical singularities and from ambiguity in the representation as every representation of an orientation with only three parameters inevitably has coordinate singularities at a number of orientations. The numerical problems of minimal representations with coordinate singularities motivates the choice for a non-minimal representation of the pose in this thesis.

Principal Contacts	Elementary Contacts
vertex-face	vertex-face
edge-face	2 vertex-face
edge-face	vertex-face, edge-edge
face-face	vertex-face, 2 edge-edge
face-face	vertex-face, 2 edge-edge, face-vertex
face-face	6 edge-edge

Table 3.1: Decomposing the PCs of Figure 3.6 into ECs depends on the configuration of the contacting objects.

Elementary contact

A PC can be decomposed into one or more *Elementary Contacts* (ECs), providing a lower level description of the contact formation, as shown in Figure 3.6. An EC is a point contact and is associated with a *contact point* and a *contact normal* . The three types of ECs (face-vertex, vertex-face and edge-edge) are shown in the two examples at the right of Figure 3.3. For the decomposition of a PC into ECs, we use the *contacting area* of the PC, as shown by the gray areas in Figure 3.6. The contacting area can be a single point (for a vertex-face, face-vertex or edge-edge contact), a line (for a face-edge or edge-face contact) or a polygon (for a face-face contact). We position the ECs at the boundary points of the (polygonal) contacting area. A contact description based on ECs therefore contains information about the *physical* contacting area, where PCs only give information about the contacting features but not about which part of those features are actually in contact.

The number and type of ECs at a given PC depend on the compliant pose $\boldsymbol{X}$ of the contacting objects at the PC. This is illustrated by the examples in Figure 3.6. The second and third example show the same two objects in the same edge-face PC, while the last three examples all show the same two objects in the same face-face PC. Although the PC remains the same at the different poses, the number of ECs changes, as well as the types of EC. This change is listed in Table 3.1 which matches the six examples in Figure 3.6. This shows how an EC is a lower level and more detailed contact state representation than a PC or a CF. This higher granularity of the EC representation will prove useful to automatically convert the output of a compliant path planner into a low-level task specification for a robot manipulator.

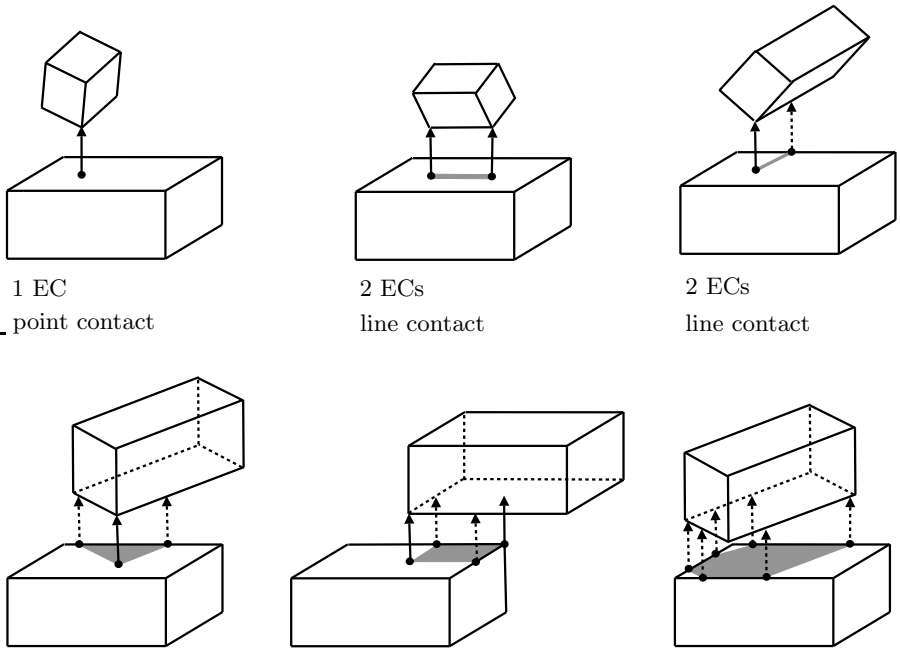


FIGURE 3.6: A principal contact (PC) can be decomposed into one or more elementary contacts (ECs), which are associated with a contact point and a contact normal. The dotted arrows indicate the edge-edge ECs, and the full arrows indicate the vertex-face or face-vertex ECs. Table 3.1 lists the ECs of each of the shown PCs.

3.2.2 Contact kinematics

Representation

The motion degrees of freedom of two objects in contact are limited by the contact constraints. When considering rigid bodies and frictionless contacts, these constraints can be approximated by the first order kinematics, in terms of a local twist space $\mathcal{T}$ and wrench space $\mathcal{W}$. A twist is a six-vector containing a translational and a rotational velocity:

$$\boldsymbol{t} = \begin{bmatrix} v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{v}^T & \boldsymbol{\omega}^T \end{bmatrix}^T, \tag{3.2}$$

while a wrench is a six-vector containing a force and a torque:

$$\boldsymbol{w} = \begin{bmatrix} f_x & f_y & f_z & \tau_x & \tau_y & \tau_z \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{f}^T & \boldsymbol{\tau}^T \end{bmatrix}^T. \tag{3.3}$$

The twist space $\mathcal{T}$ spans the s-dimensional subspace representing the s instantaneous twists degrees of freedom (DOF) that maintain the contact constraints, while the wrench space $\mathcal{W}$ spans the $(6-s)$-dimensional subspace representing the instantaneous wrench DOF where contact forces can be applied between the contacting objects. The wrench and twist space model the first order kinematic constraints of a contact between two objects at a pose $\boldsymbol{X}$. All possible wrenches of $\mathcal{W}$ are reciprocal to all possible twists of $\mathcal{T}$ (Lipkin and Duffy 1988; Duffy 1990). This means that the ideal (frictionless) contact wrenches $\boldsymbol{w}$ produce no work against the twists $\boldsymbol{t}$ allowed by the contact, and is given by:

$$\boldsymbol{w}^T\boldsymbol{t} = 0 \ [W], \tag{3.4}$$

with the work per time unit expressed in Watt $[W]$. To obtain bases $\boldsymbol{W}$ and $\boldsymbol{T}$ for the wrench space and the reciprocal twist space given the CF and the pose $\boldsymbol{X}$, we first break down the CF into its PCs. Say the CF is decomposed into q PCs. Next the wrench space at each of the q PCs is calculated. For a vertex-face or a face-vertex PC, the wrench space is spanned by:

$$\boldsymbol{W}_{PC} = \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_1 \times \boldsymbol{n} \end{bmatrix}, \tag{3.5}$$

with $\boldsymbol{n}$ the normal on the face, and $\boldsymbol{p}_1$ the position of the vertex. For an edge-edge PC, the wrench space is spanned by:

$$\boldsymbol{W}_{PC} = \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_1 \times \boldsymbol{n} \end{bmatrix}, \tag{3.6}$$

with $\boldsymbol{n}$ the common normal of the two edges, and $\boldsymbol{p}_1$ a point on the line that connects both edges along the common normal. For an edge-face or a face-edge PC the wrench space is spanned by:

$$\boldsymbol{W}_{PC} = \begin{bmatrix} \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_1 \times \boldsymbol{n} \end{bmatrix} & \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_2 \times \boldsymbol{n} \end{bmatrix} \end{bmatrix} \tag{3.7}$$

with $\boldsymbol{n}$ the normal on the face, $\boldsymbol{p}_1$ one boundary point of the edge, and $\boldsymbol{p}_2$ the other boundary point on the edge. For a face-face PC the wrench space is spanned by:

$$\boldsymbol{W}_{PC} = \begin{bmatrix} \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_1 \times \boldsymbol{n} \end{bmatrix} & \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_2 \times \boldsymbol{n} \end{bmatrix} & \begin{bmatrix} \boldsymbol{n} \\ \boldsymbol{p}_3 \times \boldsymbol{n} \end{bmatrix} \end{bmatrix} \tag{3.8}$$

with $\boldsymbol{n}$ the normal on the face, and $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, $\boldsymbol{p}_3$ three non-collinear boundary points of the face.

The wrench space of a whole CF is the union of the wrench spaces of each of the q individual PCs of the CF:

$$\boldsymbol{W}_{CF} = \left[\begin{array}{ccc} \boldsymbol{w}_{PC_1} & \ldots & \boldsymbol{w}_{PC_q} \end{array} \right]. \tag{3.9}$$

One way to obtain a base $\boldsymbol{W}$ and a base $\boldsymbol{T}$, representing the wrench and the twist space, respectively, is via a *singular value decomposition* (SVD) of $\boldsymbol{W}_{CF}$:

$$\boldsymbol{W}_{CF} = \boldsymbol{U}_{6\times 6}\boldsymbol{S}_{6\times 6}\boldsymbol{V}^T_{6\times r}, \tag{3.10}$$

where $\boldsymbol{V}$ and $\boldsymbol{U}$ are orthonormal, r is the number of wrench vectors at PCs that define the wrench space, and

$$\boldsymbol{U} = \left[\begin{array}{cc} \boldsymbol{W} & \boldsymbol{T} \end{array} \right] \tag{3.11}$$

$$= \left[\begin{array}{ccccccc} \boldsymbol{w}_1 & \ldots & \boldsymbol{w}_{(6-s)} & \boldsymbol{t}_1 & \ldots & \boldsymbol{t}_s \end{array} \right]. \tag{3.12}$$

$\boldsymbol{S}$ is a diagonal matrix containing the singular values $s_1 \ldots s_6$. The $(6-s)$ columns of $\boldsymbol{U}$ that correspond to singular values that are greater than a threshold $\epsilon \approx 0$ span the wrench space, while the s columns that correspond to smaller singular values span the twist space:

$$s_1 \geq \ldots \geq s_{6-s} > \epsilon > s_{6-s+1} \geq \ldots \geq s_6 \geq 0. \tag{3.13}$$

The columns of matrix $\boldsymbol{U} = [\boldsymbol{W}\ \boldsymbol{T}]$, calculated by the numerical SVD algorithm, are orthogonal to each other. The notion of orthogonality is often used to interpret the reciprocity condition, but is not applicable because orthogonality can only be defined between elements of the same space, and twist and wrench spaces are distinct vector spaces. However, all columns of the wrench space should be reciprocal to all columns of the twist space (Ball 1871). This means that any possible twist $\boldsymbol{t}$ of $\mathcal{T}$ produces no work in the interaction with any possible wrench $\boldsymbol{w}$ of $\mathcal{W}$:

$$\boldsymbol{W}^T\boldsymbol{T} = \boldsymbol{0}. \tag{3.14}$$

In order to interpret the orthogonal columns of $\boldsymbol{U}$ as reciprocal wrenches and twists, we assign compatible units to forces, torques, rotational velocities and translational velocities.

Transformation

At each time instant, an object has a unique translational and rotational velocity. Hence, the twist is unique and unambiguously defined, and as such a property of the objects in contact. Similarly, at each time instant, the force and torque applied to an object is unique, and hence the wrench applied to an object is unambiguously defined. However, the mathematical representation

of a twist or a wrench depends on the choice of a *reference point* and *reference frame*, as shown in Figure 3.7. Because of this, leading subscripts and superscripts are added to the notation of a twist: ${}^{a}_{f}\boldsymbol{t}$ expresses the twist reduced to a reference point a and expressed in a reference frame f. These leading subscripts and superscripts are omitted when it is clear from the context which reference point and reference frame are considered.

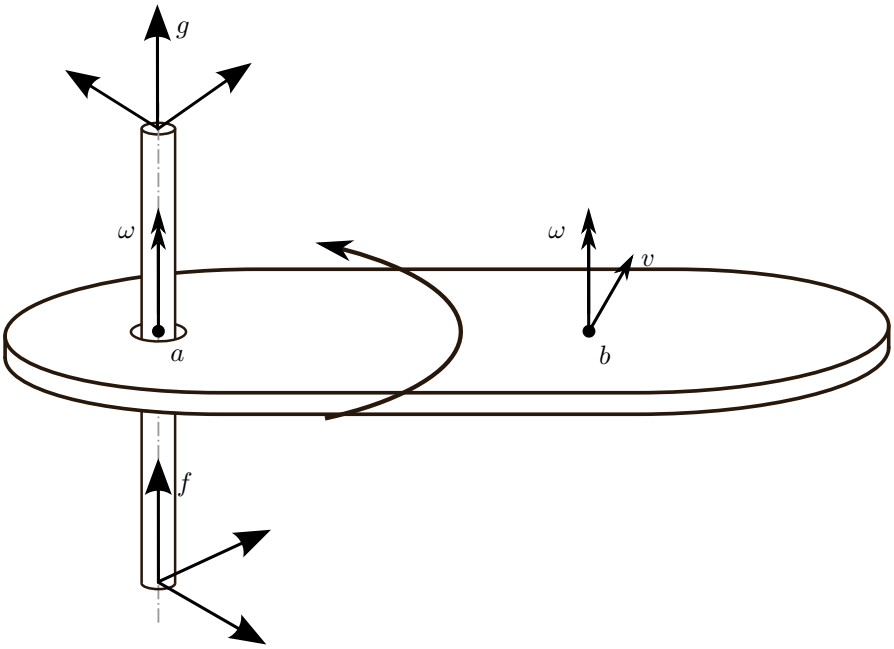


Figure 3.7: An object rotates around a vertical axis. Point a of this object, on the axis, purely rotates when expressed in frame f. Point b of this object, not on the axis, translates as well as rotates when expressed in frame f.

Both ${}^{a}_{f}\boldsymbol{t}$ and ${}^{b}_{f}\boldsymbol{t}$ express the same motion of the object and as such *the* twist of the object. However, the values of their coordinates differ because both twists are reduced to different reference points. Similarly, both ${}^{a}_{f}\boldsymbol{t}$ and ${}^{a}_{g}\boldsymbol{t}$ express the same motion of the object, however the values of their coordinates differ because both twists are expressed at different reference frames.

The 6×6 *screw projection matrix* $\boldsymbol{P}^{f}_{g}$ transforms the coordinate representation of a twist ${}^{a}_{f}\boldsymbol{t}$ from its current reference frame f to a new reference frame g, without changing the reference point:

$$ {}^{a}_{g}\boldsymbol{t} = \boldsymbol{P}^{f}_{g} \; {}^{a}_{f}\boldsymbol{t}, \tag{3.15} $$

with

$$\boldsymbol{P}_g^f = \begin{bmatrix} \boldsymbol{R}_g^f & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_g^f \end{bmatrix}. \tag{3.16}$$

The 6×6 *reference point transformation matrix* $\boldsymbol{M}_b^a$ transforms the coordinate representation of a twist ${}_f^a\boldsymbol{t}$ from its current reference point a to a new reference point b, without changing the reference frame:

$${}_f^b\boldsymbol{t} = \boldsymbol{M}_b^a \ {}_f^a\boldsymbol{t}, \tag{3.17}$$

with

$$\boldsymbol{M}_b^a = \begin{bmatrix} \boldsymbol{I} & [\boldsymbol{p}_b^a\times] \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}. \tag{3.18}$$

In this, $[\boldsymbol{p}_b^a\times]$ is the 3×3 skew-symmetric matrix representing the cross product with a 3×1 vector $\boldsymbol{p}$:

$$[\boldsymbol{p}\times] = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix}. \tag{3.19}$$

The 6×6 *screw transformation matrix* $\boldsymbol{S}_g^f$ transforms the coordinate representation of a twist ${}_f^a\boldsymbol{t}$ from its current reference point a at the origin of its reference frame f, to a new reference point b at the origin of the new reference frame g:

$${}_g^b\boldsymbol{t} = \boldsymbol{S}_g^f \ {}_f^a\boldsymbol{t}, \tag{3.20}$$

with

$$\boldsymbol{S}_g^f = \boldsymbol{M}_b^a \ \boldsymbol{P}_g^f. \tag{3.21}$$

Projection

Both twists and wrenches consist of elements from two distinct subspaces (translational and rotational velocities on one hand, and forces and torques on the other hand), and therefore common concepts such as orthogonality or Euclidean norm do not apply. Hence the projections of twists and wrenches onto their respective twist and wrench space are not invariant to changes of units. However, an invariant projection can be obtained when applying a weighted projection with a well chosen weighting matrix (Bruyninckx and De Schutter 1996). The weighted projection of a twist onto the twist space is invariant and obtains a physical meaning when the generalized 6×6 inertia matrix $\boldsymbol{K}_{\boldsymbol{M}}$ is chosen as a weighting matrix:

$$\boldsymbol{t}_p = \boldsymbol{T}\boldsymbol{T}^{\dagger_{\boldsymbol{K}_{\boldsymbol{M}}}}\boldsymbol{t}. \tag{3.22}$$

The operator ${}^{\dagger_{\boldsymbol{K_M}}}$ represents the weighted pseudo-inverse (Doty, Melchiorri, and Bonivento 1993; Nakamura 1991) of a matrix using a positive definite weighting matrix $\boldsymbol{K_M}$:

$$\boldsymbol{K_M} = \boldsymbol{L_M^T L_M}, \tag{3.23}$$

and is defined by:

$$\boldsymbol{T}^{\dagger_{\boldsymbol{K_M}}} = \left(\boldsymbol{L_M^T T}\right)^{\dagger} \boldsymbol{L_M}. \tag{3.24}$$

The † at the right-hand side now denotes the traditional Moore-Penrose pseudo-inverse (Penrose 1955). Using the inertia matrix as a weighting matrix, the weighted pseudo inverse minimizes the weighted norm of the projection error:

$$||\boldsymbol{t} - \boldsymbol{t}_p||_{\boldsymbol{K_M}} = \left|\left|(\boldsymbol{t} - \boldsymbol{t}_p)^T \boldsymbol{M} (\boldsymbol{t} - \boldsymbol{t}_p)\right|\right|. \tag{3.25}$$

This projection error has the physical meaning of the kinetic energy of the inertia that cannot be explained by the motion degrees of freedom represented by the twist space.

In the same way the projection of a wrench onto the wrench space is invariant and obtains a physical meaning when the generalized 6×6 compliance matrix $\boldsymbol{K_C}$ is chosen as a weighting matrix:

$$\boldsymbol{w}_p = \boldsymbol{W}\boldsymbol{W}^{\dagger_{\boldsymbol{K_C}}} \boldsymbol{w}. \tag{3.26}$$

Using the compliance matrix as a weighting matrix, the weighted pseudo inverse minimizes the weighted norm of the projection error:

$$||\boldsymbol{w} - \boldsymbol{w}_p||_{\boldsymbol{K_C}} = \left|\left|(\boldsymbol{w} - \boldsymbol{w}_p)^T \boldsymbol{C} (\boldsymbol{w} - \boldsymbol{w}_p)\right|\right|. \tag{3.27}$$

This projection error has the physical meaning of a potential energy in the compliance that cannot be explained by the force degrees of freedom represented by the wrench space.

3.3 Contact state graph

Compliant motion planning and control requires the knowledge of contact state space and geometry of the objects in contact (Lefebvre, Bruyninckx, and De Schutter 2003; McCarragher and Asada 1992; Schimmels and Peshkin 1992; Sturges and Laowattana 1995). The contact state space between two objects is often reduced to a compact graph representation, a *contact state graph* G. Figure 3.8 shows an example of a contact state graph. In G a node represents a contact state, while an arc connecting two nodes represents the adjacency or neighboring relationship between the contact states of the nodes. Two contact states are adjacent when a compliant motion exists from

one contact state to the other contact state, without passing through any other contact state. Figure 3.9 shows an example, where a rotation around the dashed axis causes a transition from a face-face contact state to a neighboring edge-face contact state.

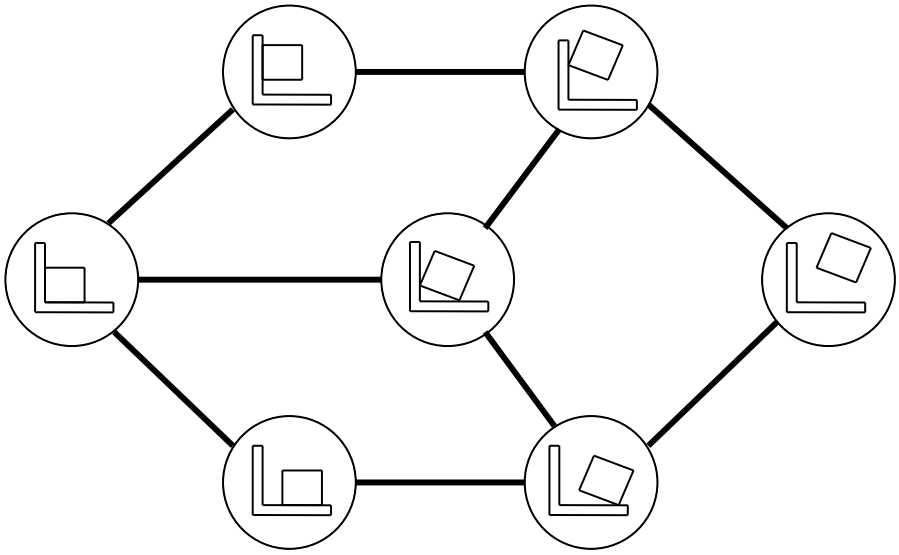


FIGURE 3.8: A contact state graph shows all possible contact states (nodes) and transitions between neighboring contact states (arcs). This figure shows a simplified example that only contains 7 CFs.

3.3.1 Automatic generation

The contact state graph is often manually extracted and fed into a system as input, which can be extremely tedious, incomplete, and error prone for even tasks of simple geometry (Sturges and Laowattana 1995). Hence a manual procedure is practically infeasible for complex tasks due to the huge number of complex contact states. To address the problem, Xiao and Ji developed a divide-and-merge approach (Xiao and Ji 2000; Xiao and Ji 2001) to automatically generate a contact state graph between two arbitrary polyhedral objects, where each node represents a possible CF between two rigid polyhedral objects. Specifically, the approach takes advantage of the fact that a contact state graph can be divided into special subgraphs called the *goal-contact relaxation* (GCR) graphs, where each GCR graph is defined by a locally most constrained CF, called the seed (for example a cube totally assembled in a

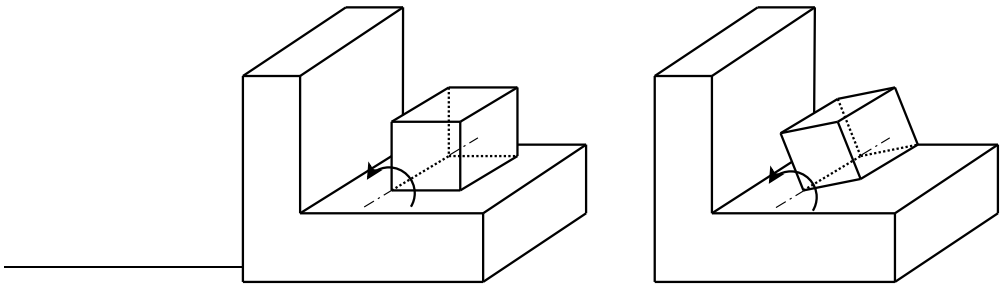


Figure 3.9: A rotation around the dashed axis causes a transition from a face-face contact state to a neighboring edge-face contact state.

corner, or a peg inserted in a hole), and its less-constrained neighboring CF. Such a subgraph is easier to generate because of several nice properties. The main properties include:

- Given a valid contact formation, CF_i , all of its less constrained neighboring CFs can be hypothesized topologically from the PCs in CF_i.
- A hypothesized less constrained neighboring contact formation, CF_j, is valid, if and only if there exists a compliant motion to relax certain constraints of CF_i to obtain CF_j that does not result in any other CF. Such a compliant motion is called a *neighboring contact relaxation motion.*
- Neighboring relaxation can be generally achieved by a CF_i-compliant motion, followed by an instantaneous compliant motion for state transition and a CF_j-compliant motion, and most neighboring relaxation can be achieved by instantaneous compliant transition motion alone.

The approach was implemented with algorithms to generate a complete GCR graph automatically, that can contain hundreds of contact formations. By combining multiple GCR graphs from multiple given locally most constrained CFs, a complete contact state graph is obtained. Figure 3.10 shows an automatically generated contact state graph for a cube in contact with a corner. The graph contains 245 nodes and is obtained from a single locally most constrained CF, namely the CF with face-face contacts where the cube is completely assembled into the corner. The research in this thesis is based on the implementation of this approach. In (Meeussen, Xiao, De Schutter, Bruyninckx, and Staffetti 2004), a method was proposed to add robot manipulator constraints to a given contact state graph. Further research on contact state graphs by the same research group applied the relaxation principle to

generate a contact state graph for 2-dimensional articulated objects (Staffetti, Meeussen, and Xiao 2005) and curved objects (Tang and Xiao 2006).

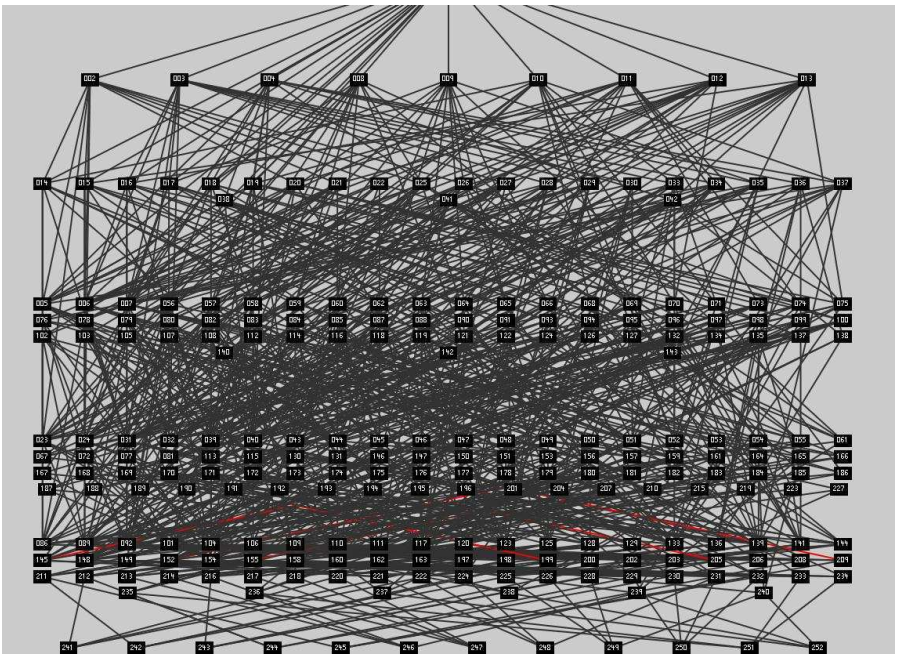


FIGURE 3.10: A contact state graph shows all possible CFs (nodes) and transitions between neighboring CFs (arcs). This figure shows an automatically generated contact state graph from a cube in contact with a corner. The graph contains 245 non-degenerated CFs.

Feng and Joseph M. (2003) present another approach to automatically generate the set of contact states that may occur during an assembly task. A contact state graph is constructed in two stages. In the first stage, all hypothetical ECs are evaluated for geometric feasibility with respect to part-imposed and robot-imposed restrictions on relative positioning. In the second stage, the feasibility of each of the various combinations of ECs is evaluated using topological existence and uniqueness criteria, and using part-imposed and robot-imposed geometric criteria.

Chen (2005) semi-automatically obtains a partial contact state graph from the human demonstration of an assembly task. All CFs that occur during a human demonstration are added to a graph, where all CFs that occurred subsequently in the demonstration are considered adjacent.

3.3.2 Manipulator constraints

The previous paragraph addressed the automatic generation of the contact state graph between rigid polyhedral objects, based on only the geometric model of the contacting objects. However, in many robotics tasks, it is a robot manipulator that moves an object and creates contacts between the object and the environment. Therefore, whether a CF of the contact state graph can be formed and whether a CF transition of the contact state graph is possible, is subject to the constraints of the manipulator. It is necessary to take into account such a manipulator in obtaining a contact state graph so that a compliant motion plan generated based on such a graph can be actually executed by the manipulator. This thesis presents an algorithm to find feasible contact states between a polyhedral part A held by a manipulator and a fixed polyhedral environment B, for a given contact state graph between an unattached part A and a fixed environment B. The approach combines the model of A and that of a manipulator with a fixed base to form the model that represents A being held by the manipulator end-effector. The combined model is called the *held A*. For this combined model, the approach first checks the reachability of each contact state of a give contact state graph, and then checks the connection between each two neighboring CFs of the contact state graph. The checks are performed by applying a virtual compliant controller to the manipulator to create possible compliant motions of A. The result of the checks is a revised contact state graph, as shown in Figure 3.11, that includes the manipulator constraints. Preliminary implementation results validate the effectiveness of the method.

First the method to verify the feasibility of a CF for the held A is presented, then the method to verify the feasibility of a CF transition for the held A is presented, and finally the implementation details are discussed.

Verification of contact formations

First, we need to check, for each CF in the given contact state graph G of the object A in contact with the environment B, whether a CF-compliant configuration of the held A can be found. Let s denote a contact state in G with a contact formation CF_s and a CF_s-compliant configuration C_s^0 of the single part A. If there exists an inverse kinematic solution for the robot manipulator such that the object A is in the CF_S-compliant configuration C_s^0, then the state s is re-expressed as $<CF_s, C_s^0>$ and is labeled as *feasible*. Otherwise, another CF_s-compliant configuration C_s^1 is sampled using the compliant random sampling algorithm presented in (Ji and Xiao 2001a; Ji and Xiao 2001b). For this new CF_S-compliant configuration C_s^1 the inverse kinematics is performed again. This process can continue until either (i) a CF_s-compliant configuration C_s^i of the held A is found and s is labeled feasi-

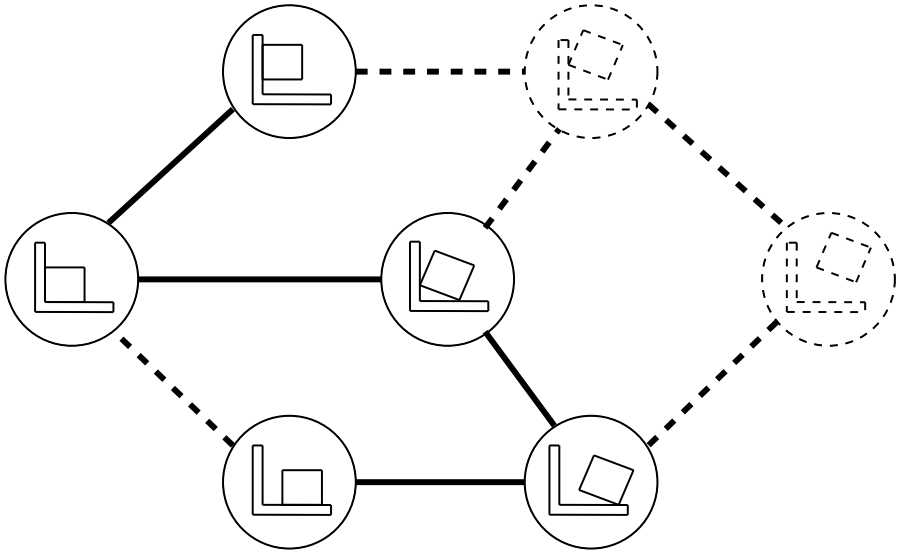


FIGURE 3.11: A revised contact state graph includes the manipulator constraints. Feasible nodes and arcs in are solid lines, infeasible nodes and arcs are in dashed lines. The resulting contact state graph is a subgraph of the original contact state graph.

ble, or (ii) no CF_s-compliant configuration of the held A can be found after a certain number of samples. In the latter case, the contact formation CF_s is considered difficult or impossible to reach, and the state s is labeled *infeasible*. This process is described in Algorithm 3.1.

Verification of contact formation transitions

Next, we perform a graph traversal of G, verifying if an arc in G, representing a compliant transition between two neighboring contact states s_i and s_j in G, is still possible with the held A, given that s_i and s_j are both feasible for the held A. The graph traversal starts from a feasible contact state s in G for the held A (see Figure 3.11). The algorithm can be written using a recursive function Feasibility-Check, shown in Algorithm 3.2.

We use a virtual compliant controller to check the feasibility of an arc between two feasible nodes. The virtual compliant controller generates a path of feasible configurations for the held A that describes a compliant relaxation motion between the given feasible configuration of the first node and the feasible configuration of the second node. The motion is generated step by step

Algorithm 3.1 The algorithm to check the feasibility of the nodes of a contact state graph G..

initialize:
$graph \Leftarrow G$
function: Node-Check ($graph$)
for each node s of the contact state graph G **do**
  number of checks = 0;
  while node s not feasible AND number of checks is smaller than maximum checks **do**
    Generate random CF_s-compliant configuration C_s
    if C_s is reachable for the manipulator **then**
      node s is labeled feasible;
    end if
    number of checks++;
  end while
end for

Algorithm 3.2 The algorithm to check the feasibility of a CF compliant motion.

initialize:
$state \Leftarrow s$
function: Transition-Check ($state$)
for each arc a_i between $state$ and a feasible neighbor $state_i$ **do**
  if a_i is not labeled feasible AND there is a feasible state transition motion of the held A from $state$ to reach $state_i$ **then**
    a_i is labeled feasible;
    Transition-Check ($state_i$);
  end if
end for

in small movements for the held A, steering it towards its goal. This time-step based approach avoids the difficulty of generating configuration space obstacles (Donald 1985). Also, we avoid the problem of *manipulator jumps*, this is when using inverse kinematics, two nearby Cartesian configurations of the moved object A could give two significantly different manipulator configurations.

The velocity of the held A is not directly specified in the joint space of the manipulator, but instead we use multiple local specifications in the Cartesian space. For this purpose, we further decompose each PC in a CF into ECs. Now we can specify the complex joint space motion of the held A using simple local specifications at ECs between the held A and the environment B. At each EC we specify a local force between the held A and the environment by virtually attaching a linear spring between the surface elements of the EC. The spring is positioned along the contact normal of the EC through the contact point of the EC, as shown in Figure 3.12. The force applied by the spring on the surface elements depends on the stiffness of the spring, its rest length and the distance between the surface elements. We distinguish different functions for ECs, such as ensuring compliant motion, avoiding obstacles or goal attraction. Depending on the function of an EC we use a different local specification by different virtual springs:

- *Maintaining a contact:* A CF-compliant motion can be realized by maintaining the ECs that are required in the CF. We call these ECs the *constraining ECs*. To maintain a constraining EC, we attach a stiff spring with rest length zero between the surface elements of the EC, to ensure that the contacting elements of the EC remain in contact.

- *Relaxation motion:* An instantaneous relaxation motion from $CF_i \rightarrow CF_j$ can be realized by keeping the constraining ECs of CF_j and breaking the ECs in CF_i that are not allowed in CF_j. We call the ECs that need to be broken to realize the relaxation motion the *relaxation ECs*. To break a relaxation EC we position a compressed spring with medium stiffness and medium rest length between the contacting surface elements of the EC, to push them apart.

- *Obstacle avoidance:* To perform local obstacle avoidance, we avoid all ECs between the held A and the environment that are not needed for the contact formation. We call these ECs the *unintended ECs*. To prevent a collision at an unintended EC, we position a compressed spring with medium stiffness and a high rest length between the surface elements of the EC, to push them apart.

- *Goal attraction:* The attraction between the held A and a desired goal configuration is based on the ECs that are required in the goal configuration. We call these ECs the *goal ECs*. At each goal EC we position

a spring with a low stiffness and zero rest length that pulls the surface elements towards the goal configuration.

- *Joint limits:* To avoid a joint from reaching the end of its range, we position a torsion spring with a high stiffness and zero rest angle at the joint, to push the joint away from its end position.

Figure 3.12 shows a configuration of the held A with three ECs. At each EC the contact normal n_i and the spring between the surface elements at the contact are shown. Each spring at a local EC applies a force to the manipulator or to the manipulated object. In the virtual compliant controller approach, all these local forces are combined into the total force applied to the manipulator and the manipulated object. Using a dynamic model of the manipulator and the manipulated object, the total force is translated into a motion of the manipulator at each timestep. Therefore all the local springs at ECs result in the desired motion of the manipulator.

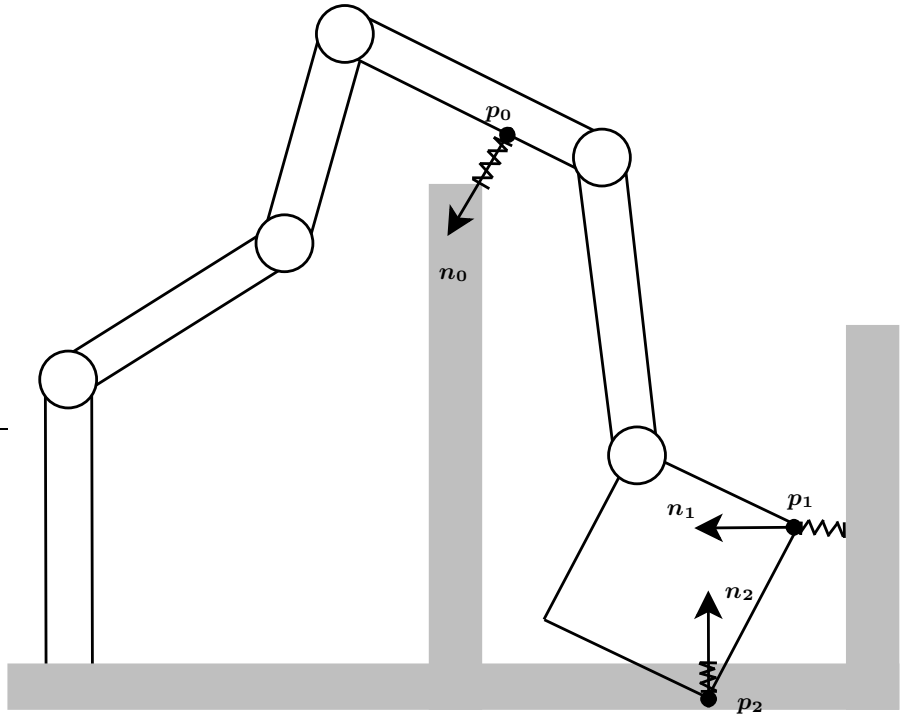


Figure 3.12: The spring along the contact normal $\boldsymbol{n_0}$ is used for local obstacle avoidance, while the springs long the contact normals $\boldsymbol{n_1}$ and $\boldsymbol{n_2}$ are used for maintaining a desired CF.

	K	x_0
Constraining EC	1000 $[N/m]$	0 $[m]$
Relaxation EC	50 $[N/m]$	0.05 $[m]$
Unintended EC	50 $[N/m]$	0.2 $[m]$
Goal EC	1 $[N/m]$	0 $[m]$
Joint EC	1000 $[N/m]$	0 $[m]$

TABLE 3.2: The spring characteristics given by a stiffness K and a rest length x_0, for each function of an EC.

Implementation

This thesis presents the implementation of the virtual compliant controller. The implementation is capable of automatically generating a compliant relaxation motion between two given configurations at neighboring CFs, for a *serial* robot manipulating a *polyhedral* object in contact with a *polyhedral* environment.

At each time step of the relaxation motion we first search all ECs that exist in the configuration of the held A at that time. The implementation only considers contacts between the manipulated object and the environment, not the contacts between the manipulator and the environment. For finding the ECs between the manipulated object and the environment, we use the collision detection algorithm by (Gilbert 1988) to find the closest features between two polyhedral objects, extended with Zhang's algorithm (Xiao and Zhang 1995) to find all PCs between two polyhedral objects. The collision detection algorithm not only finds the parts of the polyhedral objects in contact, but also the parts that are approaching a new (and possibly undesired) contact. In the next step we attach virtual springs to the surface elements of each EC. The spring characteristics depend on the type of EC, and are given in Table 3.2. Finally the motion of the held A is simulated based on an approximate dynamic model of only the manipulated object A:

$$\sum_i \boldsymbol{w}_i = \boldsymbol{M}\dot{\boldsymbol{t}} + \boldsymbol{D}\boldsymbol{t}, \tag{3.28}$$

in which $\boldsymbol{w}_i$ is the wrench generated by the force applied by the spring at EC_i expressed at the reference frame of A, and $\boldsymbol{t}$ the twist of A. The inertia matrix $\boldsymbol{M}$ is chosen equal to the real inertia of the manipulated object A and is given by:

$$\boldsymbol{M} = \begin{bmatrix} 3.6\ [kg]\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 1.9\ [kgm^2]\boldsymbol{I}_{3\times3} \end{bmatrix}, \tag{3.29}$$

and the viscous damping matrix is given by:

$$\boldsymbol{D} = \begin{bmatrix} 10\ [kg/sec]\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 5\ [kgm^2/sec]\boldsymbol{I}_{3\times3} \end{bmatrix}. \tag{3.30}$$

Given the twist of A at the previous time step, we then calculate the twist of A at the next time step using Equation (3.28). This new twist is then converted to the joint velocities of the held A using the inverse instantaneous kinematics for the serial kinematic chain. The implemented example uses the kinematic model of the Kuka 361, a six degrees of freedom industrial manipulator. The implementation however applies to any serial kinematic chain that consists of a sequence of rotational and/or translational joints. Figure 3.15 and Figure 3.15 show the OpenGL-based visualization of the virtual compliant controller, with the Kuka 361 robot manipulating a cube in contact with two perpendicular faces of a corner. The implementation presented in this thesis is limited to the automatic generation of a relaxation motion, and does not cover the whole approach to take into account the manipulator constraints. Therefore the applications and examples presented in this thesis are based on a contact state graph where the manipulator constraints were added manually.

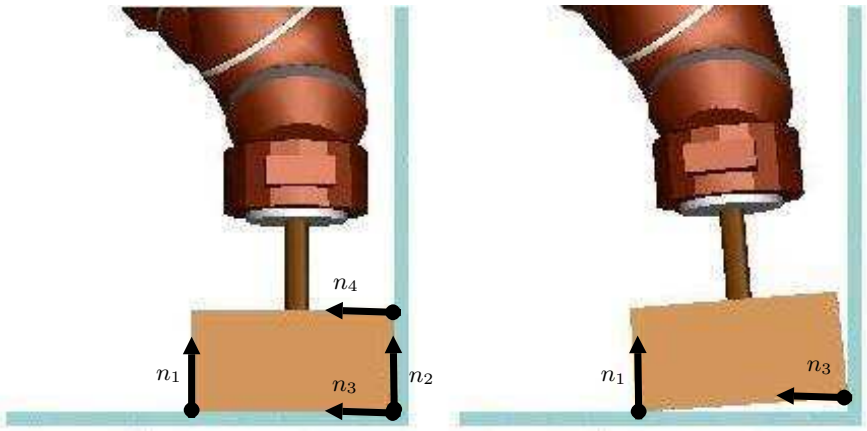


Figure 3.13: A compliant relaxation motion, with $\{n_1, \ldots, n_4\}$ the contact normals.

3.4 Path planner

Compliant motion planning can be defined as follows (Latombe 1991): given a CF_1-compliant start pose $\boldsymbol{X}_1$ and a CF_m-compliant end pose $\boldsymbol{X}_n$, find a path between them in the contact space of the manipulated object and

the environment. The path must be collision-free for the manipulator. Figure 3.14 shows a simplified representation of the motion planning problem, for a 2-dimensional configuration space of two contacting objects. The dotted line represents the searched compliant path connecting $\boldsymbol{X}_1$ and $\boldsymbol{X}_n$. Many robotics tasks require compliant motions, but planning such motions poses special challenges not present in collision-free motion planning. One challenge is to achieve compliant configurations to a desired CF, especially when the configuration manifold of the CF is hard to describe analytically due to high geometrical complexity and/or high dimensionality.

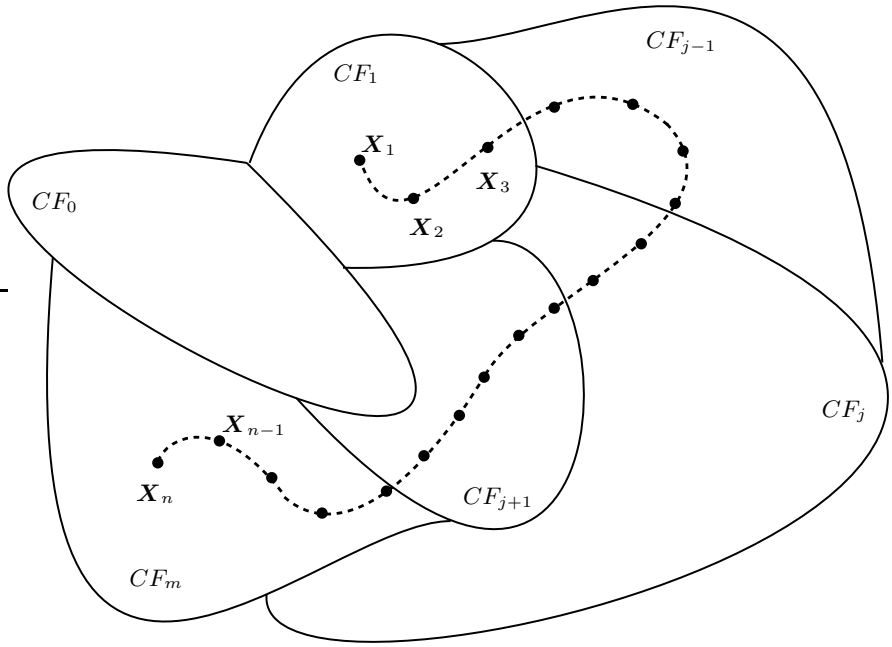


FIGURE 3.14: A simplified 2-dimensional configuration space representation of a compliant path from $\boldsymbol{X}_1$ at CF_1, to $\boldsymbol{X}_n$ at CF_m.

Ji and Xiao developed a hybrid (partly discrete, partly continuous) geometric approach to tackle this problem (Ji and Xiao 2001a). First a high-level discrete graph search in the (manually) modified contact state graph with manipulator constraints results in a sequence of contact transitions between adjacent CFs, connecting CF_1 and CF_m. Next a low-level continuous motion planner, based on a randomized planner for planning CF-compliant motion between two arbitrary polyhedral solids, is used. This motion planner extends the *probabilistic roadmap paradigm* (Kavraki and Latombe 1998) for

planning collision-free motion to the space of contact configurations. This motion planner is used to interpolate between the contact formations of the high level planner. Within each contact formation CF_j of the high level path, with $j = 1 \dots m$, it produces a sequence of CF_j-compliant poses. The first pose of CF_j connects to the last pose of CF_{j-1}, and the last pose of CF_j connects to the first pose of CF_{j+1}, resulting in the desired compliant path.

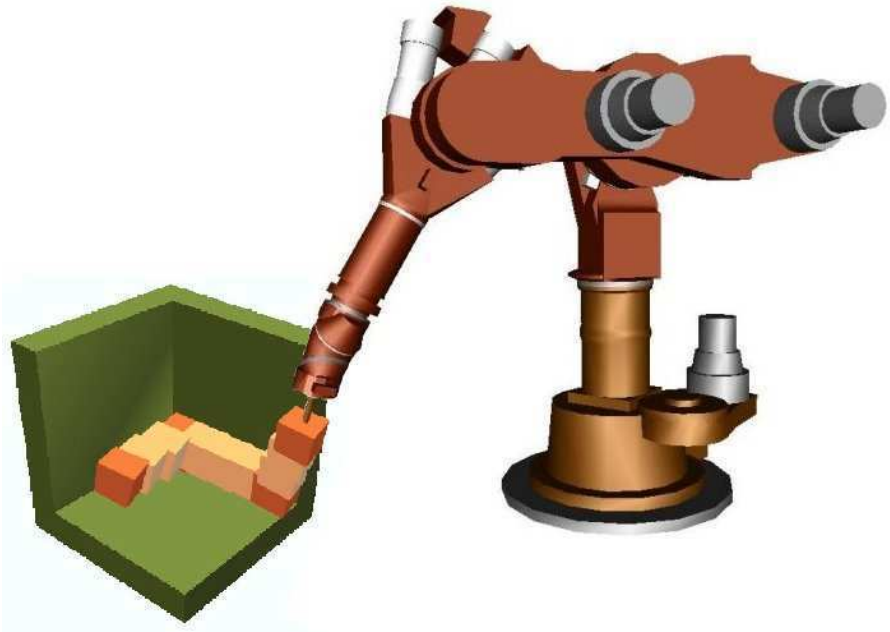


Figure 3.15: The compliant path generated by the compliant path planner is given by a sequence of contact formations and their respective poses.

The output of this compliant planner, as shown in Figure 3.15, only contains geometrical and topological information in the form of a sequence of poses $\boldsymbol{X}_1 \dots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \dots CF_m$. Each two poses $\boldsymbol{X}_i$ and $\boldsymbol{X}_{i+1}$ are at the same or at neighboring contact formations, as shown in Figure 3.14.

3.5 Conclusions

This chapter describes how contacts between rigid polyhedral objects are characterized topologically by contacts between the surface elements of the objects. Elementary contacts (ECs) are characterized by a contact point and a

contact normal, and form the lowest level of contact description. Principal contacts (PCs) are characterized by a contact plane, and can be decomposed into one or more ECs. The highest level of contact description, the contact formation (CF) is defined by the set of PCs formed. A first order approximation of the contact kinematics is given by the local twist and wrench space.

A contact state graph represents all possible CFs between two rigid polyhedral objects as nodes, and the adjacency relationship between the CFs as arcs. In this thesis we build upon the goal contact relaxation algorithm to automatically generate a contact state graph given the geometric model of two objects and a locally most constrained CF. The resulting modified contact state graph can be used to plan compliant motions for a robot manipulator. The implementation of the approach to automatically add manipulator constraints to a given contact state graph only covers the automatic generation of compliant relaxation motions between two given configurations of neighboring CFs.

A compliant motion planner builds upon the modified contact state graph with manipulator constraints to generate a compliant path between a given start and goal configuration. The output of a compliant planner only contains geometrical and topological information in the form of a sequence of poses $\boldsymbol{X}_1 \ldots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \ldots CF_m$.

Chapter 4

Human demonstration for compliant motion

For the things we have to learn before we can do them, we learn by doing them.

Aristotle, Ethica Nichomachea, ca. 350 B.C.

4.1 Introduction

While in the previous chapter a compliant path is obtained from an automatic compliant path planner, in this chapter a compliant path is obtained using *programming by human demonstration* (PbD), where a human demonstrates the desired compliant motion task, as illustrated in Figure 4.1 (top right). The demonstration can be performed in a virtual environment using a haptic device, in the real world by directly interacting with a robot through a master slave system, or, as used in this thesis, by observing human motion when the demonstrator directly manipulates the objects in the environment without

the use of a robot. After the demonstration, in an interpretation step, these sensor data are translated into a compliant path, given the geometric model of the objects. This path is defined by the same parameters as a path generated by the compliant motion planner presented in the previous chapter. The interpretation step distinguishes the PbD approach from teach methods (see Section 2.2.1. When using a teach method, the recorded sensor data is simply *replayed*, without any higher level interpretation. When, because of uncertainties in the task, during the execution of the task the low level sensor signals differ from the recorded low level sensor signals, the task execution fails. In PbD the sensor data is *interpreted* based on a task model, and a high level task description is stored. During the execution of the task, the system evaluates and corrects the motion of the robot manipulator, so that the final robot motion matches the high level task plan; this results in a robust task execution. Due to uncertainties in the task, the low level sensor signals of the executed task are different from the low level sensor signals during the demonstration, but because the robot motion is corrected during the execution based on the high level task description, the high level task still gets accomplished. For example, a high level task description could say *more an object sideways till you hit a wall, and then move along the wall*. For this task to be executed successfully based on the high level description, it is not important to know the exact position of the wall; the system only needs to detect when the object hits the wall, and then continue with the second part of the task. This same task based on a low level task description in terms of desired robot joint velocities, would fail when the position of the wall is not exactly the same during the teaching phase and the execution phase; because the low level description has no concept of a "wall" it will not be able to detect when the object hits the wall at an unexpected position.

Major challenges in the automatic translation from human compliant motion demonstration into a path plan of a compliant motion are: (i) to recognize the *contact formation* to which the human demonstration is currently subjected, (ii) to estimate the geometric parameters of that contact formation (this is the position of contact point(s), the direction of contact normal(s), etc.), and (iii) to detect when exactly the human demonstration execution changes between two contact formations. This thesis generalizes and scales the previously presented approach of Gadeyne, Lefebvre, and Bruyninckx (2005) to cope with *all possible contacts* between two polyhedral objects. To cope with this increased complexity, a more accurate prediction step is used, based on the topological information contained in a contact state graph (Xiao and Ji 2000; Xiao and Ji 2001), and the pose of the contacting objects. Efficient algorithms for the pose and consistency measurement equations allow the estimators to be used in *realtime*. The approach in this chapter is applied in an experiment where a human demonstrator manipulates a cube in contact

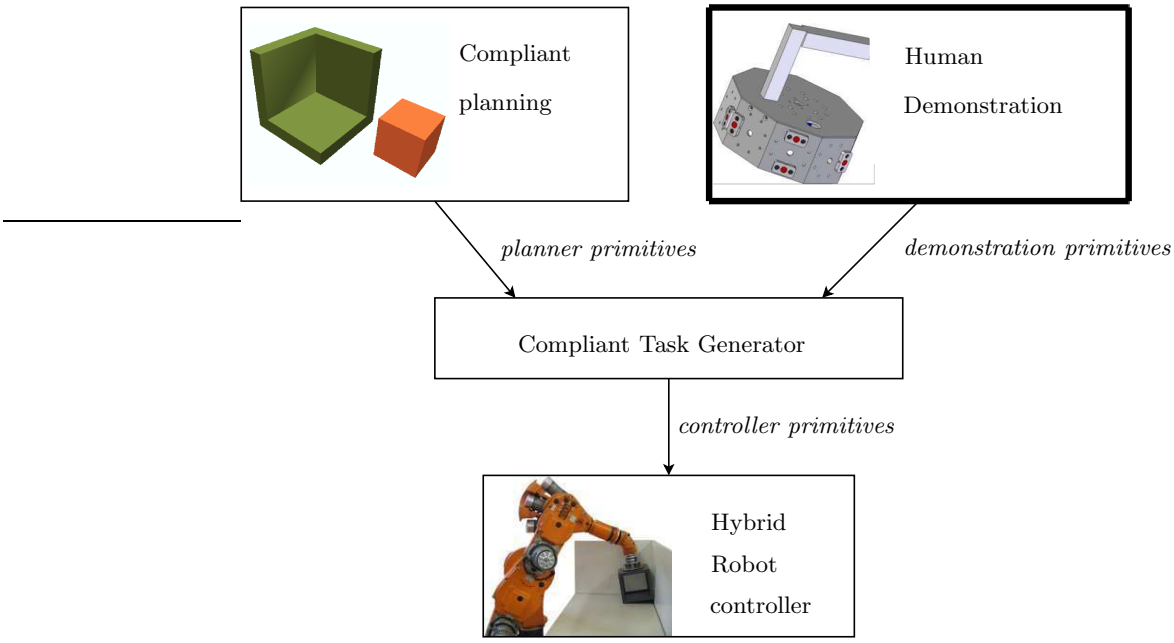


FIGURE 4.1: This thesis presents two high level approaches for task specification in active compliant motion: compliant path planning and programming by human demonstration. The compliant task generator converts the task specification into instantaneous setpoints for the hybrid robot controller. This section discusses programming by human demonstration.

with the three faces of a corner, as shown in Figure 4.2. The result of the demonstration is a compliant path, described by the recognized sequence of contact formations and the corresponding poses. The real world experimental results that show the effectiveness of the presented approach are presented in Chapter 7.

This chapter is organized as follows. The first section describes the design and sensor processing of the demonstration tool which is used to collect sensor data during human demonstration in compliant motion. The second section discusses the interpretation of these sensor data, using Bayesian estimation techniques. The algorithms that are used to implement the system and measurement models of the particle filter are presented in the next section, followed by a discussion about the limitations of the approach. Finally the conclusions are presented.

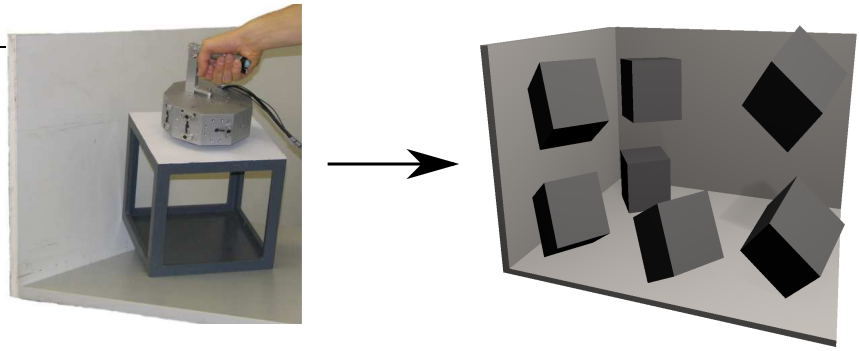


Figure 4.2: In the presented experiment, a human demonstrator manipulates a cube in contact with the three faces of a corner. The demonstration results in a geometric task description formed by the recognized sequence of contact formations and poses.

4.2 Demonstration tool

In programming by human demonstration, a task specification for a compliant task involving a manipulated object and its environment is obtained by observing a human demonstrate the desired task. Several approaches are available to perform a demonstration of a compliant motion task, such as directly interacting with the robot, demonstrating in a virtual environment, or by observing a human demonstrator directly manipulating and interacting with an object. This last option requires a *demonstration tool* (Breidenbach, Koeppe, and Hirzinger 1996; Hirai, Noguchi, and Iwata 1996) that allows a human demonstrator to manipulate an object while multiple sensors register various parameters of the demonstration. The object attached to the demonstration tool is called the *manipulated object*, while the fixed objects in the environment are called *environmental objects*.

4.2.1 Design

Requirements

The demonstration tool is designed with a focus on manipulation and identification tasks. The most important design requirements are:

- *pose and wrench measurements:* the targeted tasks to be demonstrated are compliant motion tasks. This requires 6D pose and 6D wrench

measurements.

- *small and lightweight:* to allow for easy manipulation of objects, the demonstration tool must be small, so that the manipulated objects are close to the demonstrator's hand, and lightweight, for easy movement.
- *robust and rigid:* the pose measurements are obtained from a camera system which is sensitive to deformations of the demonstration tool.
- *room for other sensors:* for further research purposes there has to be room for other sensors, such as a camera and a laser distance sensor.

Based on these specifications, two designs for the demonstration tool were proposed (Princen and Stallaert 2004; Claassen and Serdons 2006). In this thesis, the latter design was chosen for its robustness. A CAD model of this chosen design is shown in Figure 4.3. Figure 7.11 shows the real tool during an experiment. The tool is a hollow cylinder-like shape, consisting of nine faces in 40 [*deg*] increments. The handle on top provides an easy grasp for the human demonstrator to manipulate the demonstration tool and the object attached to it.

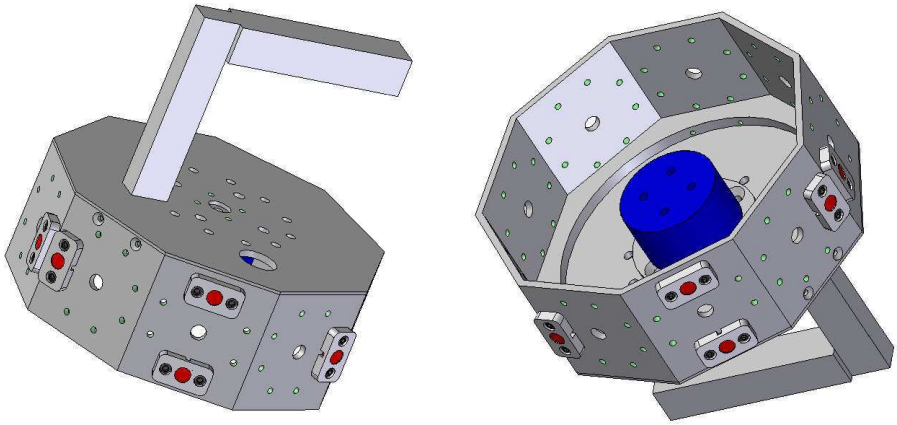


FIGURE 4.3: The design of the demonstration tool, seen from above (left) and below (right). The red LED markers are used to track the pose of the demonstration tool in space, while a wrench sensor mounted inside the demonstration tool measures the interaction forces with the environment.

Sensors

- The *Krypton K600* optical system is used to measure the pose of the demonstration tool, as shown in Figure 4.4. The K600 consists of three

calibrated linear cameras that observe a number of LED markers fixed to the demonstration tool. On each of the nine faces of the demonstration tool, up to four LED markers can be mounted on eight different positions. Each LED marker is flashed separately, and its position is observed by the three cameras simultaneously. By means of triangulation, the 3D position in space of each of the LED markers is calculated. The measured positions of the LED markers relative to the camera system are $\boldsymbol{p}^k_{L_1} \ldots \boldsymbol{p}^k_{L_v}$, with v the number of visible LED markers. The system measures LED positions at 100 $[Hz]$, with a volumetric accuracy of 90 $[\mu m]$.

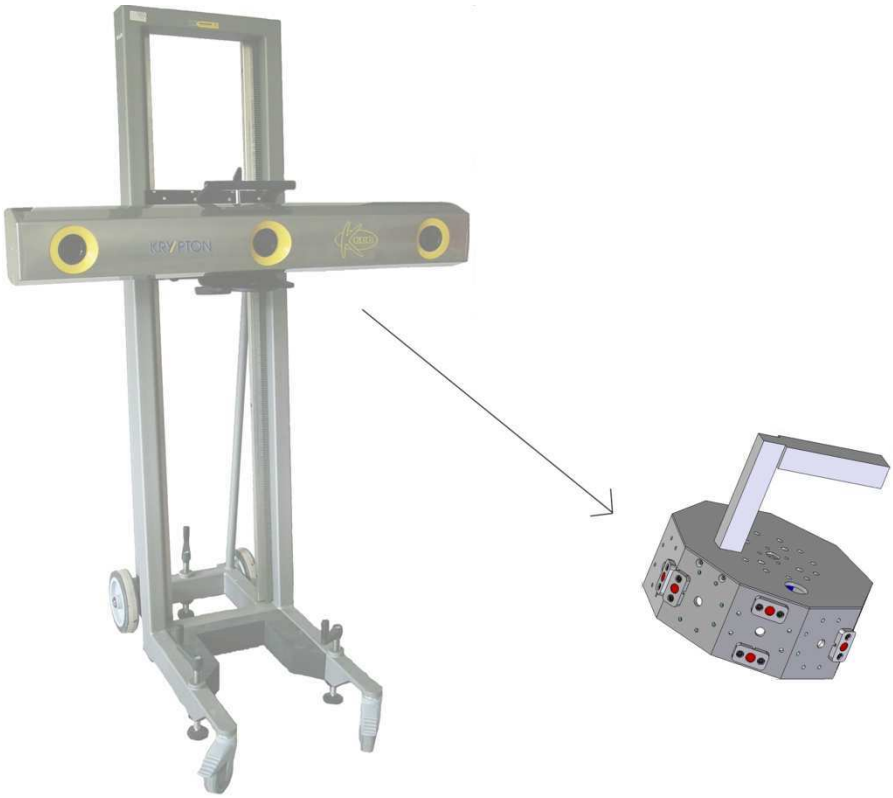


Figure 4.4: The Krypton K600 6D optical system uses three cameras and triangulation algorithms to accurately measure the spatial position of each of the LED markers on the demonstration tool.

- The *wrench sensor* is the 50M31A from JR3, which is their smallest sen-

sor currently available. The sensor is mounted between the demonstration tool and the object that is manipulated, and measures the wrench applied by the human demonstrator. The sensor has a diameter of 50 $[mm]$ and a height of 31 $[mm]$. The maximum measurable force is 100 $[N]$ for f_x and f_y, and 200 $[N]$ for f_z. The sensor is rather big with respect to the tool, but all electronics reside in the sensor itself; the link between the sensor and its interface card is digital.

- The demonstration tool also provides room for other sensors, such as a laser distance sensor and a small camera.

4.2.2 Pose and twist estimation

While the wrench $\boldsymbol{w}_m$ is directly measured by a physical sensor, the pose $\boldsymbol{X}_m$ and twist $\boldsymbol{t}_m$ of the manipulated object are indirectly measured through the position of the LED markers. When $v \geq 4$ non-collinear LED markers $L_1 \ldots L_v$ on the demonstration tool are visible to all three cameras simultaneously, the relative pose $\boldsymbol{X}_t^k$ between the demonstration tool (t) and the camera (k) can be calculated using the method described below. The positions of the visible LED markers relative to the camera are represented by $\boldsymbol{p}_{L_1}^k \ldots \boldsymbol{p}_{L_v}^k$. The positions of the visible LED markers relative to the demonstration tool are constants obtained during an initial calibration phase (Section 4.2.4), and are represented by $\boldsymbol{p}_{L_1}^t \ldots \boldsymbol{p}_{L_v}^t$. The homogeneous pose transformation matrix $\boldsymbol{X}_k^t$ transforms the positions of the LED markers from the camera to the demonstration tool (Higham 1986):

$$\boldsymbol{X}_k^t \begin{bmatrix} \boldsymbol{p}_{L_1}^t & \cdots & \boldsymbol{p}_{L_v}^t \\ 1 & \ldots & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_{L_1}^k & \cdots & \boldsymbol{p}_{L_v}^k \\ 1 & \ldots & 1 \end{bmatrix}. \tag{4.1}$$

Reducing the pose matrix to a rotation $\boldsymbol{R}_k^t$ and a translation $\boldsymbol{p}_k^t$, it can be calculated by:

$$\begin{bmatrix} \boldsymbol{R}_k^t & \boldsymbol{p}_k^t \end{bmatrix} = \begin{bmatrix} \boldsymbol{p}_{L_1}^k & \cdots & \boldsymbol{p}_{L_v}^k \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{L_1}^t & \cdots & \boldsymbol{p}_{L_v}^t \\ 1 & \ldots & 1 \end{bmatrix}^\dagger \tag{4.2}$$

in which † represents the Moore Penrose pseudo-inverse (Penrose 1955) of a matrix. The pose defined by $\boldsymbol{R}_k^t$ and $\boldsymbol{p}_k^t$ can be transformed into a minimal six-dimensional pose representation:

$$\boldsymbol{x}_m = \begin{bmatrix} x & y & z & \alpha & \beta & \gamma \end{bmatrix}^T, \tag{4.3}$$

using α, β and γ ZYX-Euler angles[1]. This "measured" pose $\boldsymbol{x}_m$ is an input to a linear estimation problem to obtain both the demonstration tool's pose

[1] The singularities of the representation can be avoided by using a different representation (such as Roll Pitch Yaw angles) depending on the position.

$\boldsymbol{X}_m$ and twist $\boldsymbol{t}_m$, based on a constant acceleration model (Bar-Shalom and Li 1993). A constant acceleration model assumes the demonstration tool is moving with a constant acceleration, but the acceleration is constantly adapted based on the pose measurements. This results in a smooth estimation for the pose and twist of the demonstration tool, while directly deriving the twist from the pose measurements would result in a noisy and inaccurate twist estimation. The Kalman filter (KF) (Kalman 1960; Sorenson 1985) is the preferred tool for this linear estimation problem with low uncertainties. The filter uses an 18-dimensional state vector which contains the pose $\boldsymbol{x}$, the velocity $\dot{\boldsymbol{x}}$ (derivative of $\boldsymbol{x}$) and the acceleration $\ddot{\boldsymbol{x}}$ (derivative of $\dot{\boldsymbol{x}}$). All three 6-dimensional parameters are estimated from the measured pose $\boldsymbol{x}_m$.

The filter's *system update* extrapolates the estimated pose $\hat{\boldsymbol{x}}$, velocity $\hat{\dot{\boldsymbol{x}}}$ and acceleration $\hat{\ddot{\boldsymbol{x}}}$ at time step k, to make a prediction of the pose $\tilde{\boldsymbol{x}}$, velocity $\tilde{\dot{\boldsymbol{x}}}$ and acceleration $\tilde{\ddot{\boldsymbol{x}}}$ at time step $k+1$, using a constant acceleration model:

$$\begin{bmatrix} \tilde{\boldsymbol{x}} \\ \tilde{\dot{\boldsymbol{x}}} \\ \tilde{\ddot{\boldsymbol{x}}} \end{bmatrix}_{k+1} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{\Delta t} & \boldsymbol{\Delta t}^2/2 \\ \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{\Delta t} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}} \\ \hat{\dot{\boldsymbol{x}}} \\ \hat{\ddot{\boldsymbol{x}}} \end{bmatrix}_k . \tag{4.4}$$

The filter's *measurement update* uses the difference between the measured pose $\boldsymbol{x}_m$ and the predicted pose $\tilde{\boldsymbol{x}}$ at time step $k+1$, to update the estimated pose $\hat{\boldsymbol{x}}$, velocity $\hat{\dot{\boldsymbol{x}}}$ and acceleration $\hat{\ddot{\boldsymbol{x}}}$ at time step $k+1$:

$$\begin{bmatrix} \hat{\boldsymbol{x}} \\ \hat{\dot{\boldsymbol{x}}} \\ \hat{\ddot{\boldsymbol{x}}} \end{bmatrix}_{k+1} = \begin{bmatrix} \tilde{\boldsymbol{x}} \\ \tilde{\dot{\boldsymbol{x}}} \\ \tilde{\ddot{\boldsymbol{x}}} \end{bmatrix}_{k+1} + \boldsymbol{K}\left(\boldsymbol{x}_m - \tilde{\boldsymbol{x}}\right), \tag{4.5}$$

in which $\boldsymbol{K}$ is called the Kalman gain (Kalman 1960), which is a function of the Gaussian uncertainty on the estimated state, and the additive Gaussian noise on the measured pose $\boldsymbol{x}_m$.

The minimal representation of the estimated pose $\hat{\boldsymbol{x}}$, which uses ZYX Euler angles α, β and γ, is transformed into the pose $\boldsymbol{X}_m$ which uses a rotation matrix (Sciavicco and Siciliano 1996):

$$\boldsymbol{X}_m = \begin{bmatrix} c\gamma\ c\alpha - c\beta\ s\alpha\ s\gamma & c\gamma\ s\alpha + c\beta\ c\alpha\ s\gamma & s\gamma\ s\beta & x \\ -s\gamma\ c\alpha - c\beta\ s\alpha\ c\gamma & -s\gamma\ s\alpha + c\beta\ c\alpha\ c\gamma & c\gamma\ s\beta & y \\ s\beta\ s\alpha & -s\beta\ c\alpha & c\beta & z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.6}$$

in which $c\alpha$ and $s\alpha$ are abbreviations of $cos(\alpha)$ and $sin(\alpha)$. The derivative $\hat{\dot{\boldsymbol{x}}}$ represents the change of ZYX Euler angles over time. For given ZYX Euler angles α, β and γ, this derivative of ZYX Euler angles is transformed into the

twist $\boldsymbol{t}_m$ using (Sciavicco and Siciliano 1996):

$$\boldsymbol{t}_m = \begin{bmatrix} \boldsymbol{I}_{3\times3} & & \boldsymbol{0}_{3\times3} & \\ & 0 & -sin(\alpha) & cos(\alpha)cos(\beta) \\ \boldsymbol{0}_{3\times3} & 0 & cos(\alpha) & sin(\alpha)cos(\beta) \\ & 1 & 0 & -sin(\beta) \end{bmatrix} \hat{\boldsymbol{x}}. \tag{4.7}$$

4.2.3 Contact wrench calculation

The wrench sensor is mounted between the demonstration tool and the manipulated object, and measures the total wrench $\boldsymbol{w}_s$. While the wrench sensor also measures the gravity and inertia caused by the mass of the manipulated object, we are only interested in the contact interaction of the manipulated object with the environment. While the inertia forces are neglected, the gravity forces are calculated and subtracted from the total measured wrench, to obtain the contact interaction wrench $\boldsymbol{w}_m$:

$$\boldsymbol{w}_m = \boldsymbol{w}_s - \boldsymbol{w}_o - \begin{bmatrix} m\boldsymbol{g} \\ m\boldsymbol{g} \times \boldsymbol{p}_c \end{bmatrix}, \tag{4.8}$$

in which m is the mass of the manipulated object, and $\boldsymbol{p}_c$ is the position of the center of gravity of the manipulated object. The 3×1 gravity constant is given by $\boldsymbol{g}$. The wrench offset $\boldsymbol{w}_o$ is a constant wrench that is added to the measured wrench, and is equal to the load on the wrench sensor when it was first powered on; when first powering on the wrench sensor, it initiates itself at a zero wrench, independent of the load at that time. The next section explains how the mass, the center of gravity, and the wrench offset are calculated in a calibration phase.

4.2.4 Calibration

To obtain the measured contact wrench, the wrench offset and the gravitational force have to be subtracted from the total measured wrench, as shown in Equation (4.8). The mass of the manipulated object, its center of gravity, as well as the offset on the wrench sensor are estimated during a calibration phase. In general, 10 parameters must be estimated:

- the wrench offset (6 parameters),
- the mass of the manipulated object (1 parameter), and
- the center of gravity of the manipulated object (3 parameters).

The gravitational force is always oriented downwards. However, the Krypton camera system measures the pose of the demonstration tool relative to the

camera, but the pose of the camera relative to the world reference is unknown. Therefore, in the camera frame it is also unknown what the orientation of the downward gravitation force is, so in order to perform gravity compensation, the orientation of the vertical axis of the camera frame must be known with respect to the world frame. Since it is difficult to manually align the Krypton camera frame with the world frame, this orientation cannot be measured directly, so two extra parameters are to be determined in the calibration procedure:

- the orientation of the vertical world axis relative to the camera (2 parameters),

increasing the number of unknown parameters to 12.

The preferred tool to estimate the 12 parameters (the state vector) for this estimation problem is the Non Minimal State Kalman Filter (NMSKF) (Lefebvre, Bruyninckx, and De Schutter 2005). The NMSKF transforms the measurement equation to a higher dimensional non-minimal state space, making the measurement equation linear in the non-minimal state variables. This renders the NMSKF robust to large uncertainties on the state estimate. For this calibration procedure, this robustness to large uncertainties is needed, since the offsets on the force measurement vary highly, and since the mass of the manipulated object is a priori not known.

Fig. 4.5 shows the definition of the relevant frames and vectors. The wrench vector $\boldsymbol{w}_g$ is the gravity vector. Expressed at the center of gravity of the manipulated object c, it is given by only a force component:

$$ {}^{c}_{c}\boldsymbol{w}_g = \begin{pmatrix} -m\boldsymbol{g} \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_g \\ \mathbf{0} \end{pmatrix} \tag{4.9} $$

This corresponds to a wrench ${}^{s}_{s}\boldsymbol{w}_g$ expressed in the sensor frame s:

$$ {}^{s}_{s}\boldsymbol{w}_g = \boldsymbol{S}^{c}_{s} \; {}^{c}_{c}\boldsymbol{w}_g. \tag{4.10} $$

If the demonstration tool is moved in free space with only small accelerations, so that the inertial influence on the wrench measurement is negligible, the following measurement equation holds:

$$ \boldsymbol{w}_s - \boldsymbol{w}_o - \boldsymbol{S}^{c}_{s} \; {}^{c}_{c}\boldsymbol{w}_g = \mathbf{0}, \tag{4.11} $$

since the total measured wrench $\boldsymbol{w}_s$, compensated for gravity and the offset $\boldsymbol{w}_o$, must then be zero. This equation is an implicit and non-linear measurement equation, function of the state and the measurements. This measurement equation is applied in the NMSKF to estimate the 12 unknown state parameters.

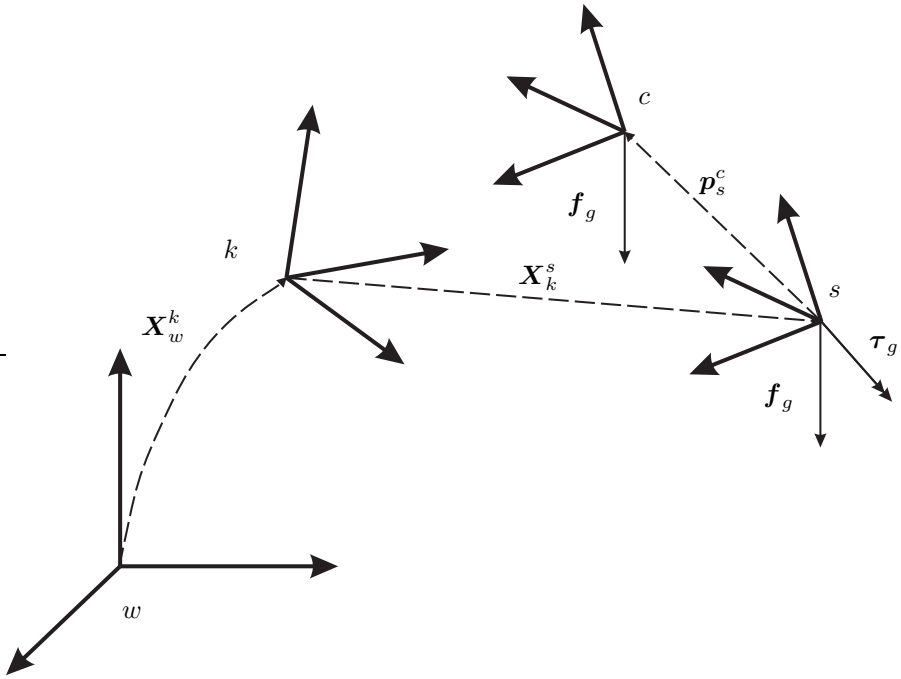


FIGURE 4.5: The frames, relevant to the force-torque calibration problem. w is the world reference frame, k is the camera frame, s is the reference frame of the wrench sensor (attached to the demonstration tool), and c is the center of gravity of the manipulated object.

During the calibration procedure, the demonstration tool is moved into different poses, and in each pose a wrench measurement is taken. Each wrench measurement is informative about a different part of the unknown mass, center of mass and wrench offset. Figure 4.6 shows how the estimation of the mass evolves during the calibration procedure. The full line shows how the mean of the estimate evolves towards the correct value, while the dashed lines show the decreasing 2 σ boundaries. The estimates on the other state parameters show a similar evolution. The filter successfully estimates the 12 state parameters during the calibration phase.

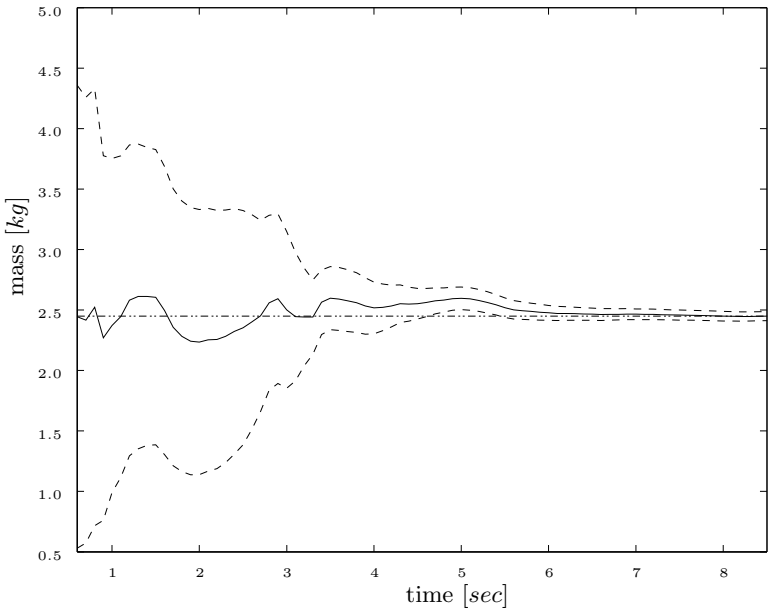


Figure 4.6: The time evolution of the calibration procedure where the mass of the manipulated object is estimated. The full line shows the mean of the estimation, the dashed lines show the 2 σ boundaries.

4.3 Simultaneous recognition of contact formations and estimation of geometric parameters

A compliant motion task can be segmented into a sequence of CFs. At each CF, different contact constraints apply. Therefore, in order to estimate uncertain geometric parameters of the objects involved in a compliant motion task, the knowledge of the current CF model is required. On the other hand, in order to improve the segmentation into CFs, the knowledge about the geometric parameters should increase. This means that the estimation problem for compliant motion tasks consists of two connected sub-problems: the recognition of the (*discrete*) CF and the estimation of (*continuous*) geometric parameters. Both sub-problems should be solved simultaneously.

4.3.1 Bayesian estimation

Hybrid probability density function

For the simultaneous recognition of (discrete) CFs and the estimation of (continuous) geometric parameters, a hybrid *Probability Density Function* (PDF) is required. A hybrid PDF contains both continuous and discrete variables. A time-invariant variable is called a *parameter*, while a time-dependent variable is called a *state*. As shown in Figure 4.7, the continuous parameters in this thesis are called the geometric parameters, denoted by $\boldsymbol{\Theta}$, and represent the pose of the manipulated object relative to the demonstration tool and the pose of the environmental object relative to a world reference. Note that while the pose of the objects is unknown, their geometry is known. The discrete state in this thesis represents the CF at time step k, denoted by $\mathcal{CF}_k$. Figure 4.8 shows an example of a hybrid PDF, for a one-dimensional continuous parameter $\boldsymbol{\Theta}$ and a one-dimensional discrete state $\mathcal{CF}$. The hybrid PDF represents the belief that, at time step k, the discrete state $\mathcal{CF}_k$ is j, with $0 \leq j < \#$ CFs, and the continuous parameters $\boldsymbol{\Theta}$ have a certain value $\boldsymbol{\theta}$, given that the measurements $\boldsymbol{Z}_{1\ldots k}$ have a certain value $\boldsymbol{z}_{1\ldots k}$:

$$P\left(\boldsymbol{\Theta} = \boldsymbol{\theta}, \mathcal{CF}_k = j \mid \boldsymbol{Z}_{1\ldots k} = \boldsymbol{z}_{1\ldots k}\right). \tag{4.12}$$

In the rest of the thesis, the notation $\boldsymbol{A} = \boldsymbol{a}$ is shortened into $\boldsymbol{a}$, wherever the distinction between a stochastic variable and an actual value is unambiguous. Each $\mathcal{CF}_k = j$ of the hybrid PDF has its own continuous PDF:

$$P\left(\boldsymbol{\theta} \mid \mathcal{CF}_k = j, \boldsymbol{z}_{1\ldots k}\right). \tag{4.13}$$

Particle filter

The causal relations between the hybrid state at each time step, and the measurements at each time step, are represented in a graphical *Bayesian Network* (BN) (Charniak 1991; Jordan 1999) in Figure 4.9. The grey nodes denote observed random variables, while the transparent nodes denote unknown (hidden) random variables. The arrows denote the causal relationship between the nodes. Figure 4.10 shows a detailed view of the BN inside the hybrid state and the measurements. The BN for this estimation problem is conform the Markov assumption that the past and future data are independent if one knows the current state.

Gadeyne, Lefebvre, and Bruyninckx (2005) show that while Kalman filter variants cannot cope with the cross-dependency between discrete and continuous variables in the BN, particle filter variants (sequential Monte Carlo methods) can by using a hybrid PDF (Doucet, Gordon, and Krishnamurthy 2001). Therefore this thesis uses a particle filter to recursively update the

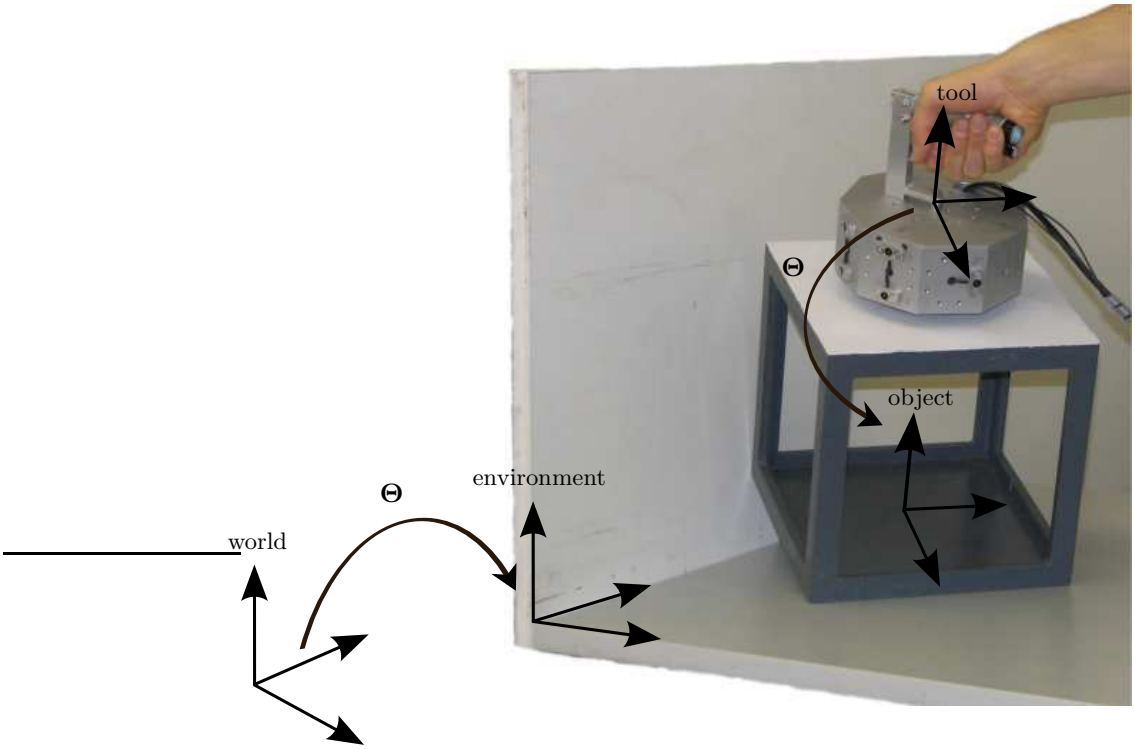


Figure 4.7: The continuous geometric parameters $\mathbf{\Theta}$ represent the pose of the manipulated object relative to the demonstration tool and the pose of the environmental object relative to a world reference.

given BN, and estimate the hybrid PDF. In a particle filter algorithm, a PDF is represented by a number of discrete samples or *particles*. Each particle corresponds to one possible value of the hybrid state; one value for the discrete state, and one value for each of the continuous parameters. With enough particles, it is possible to make a good approximation of any discrete, continuous or hybrid PDF. This allows a particle filter to deal with virtually any type of PDF, while a Kalman filter can only deal with Gaussian PDFs. However there is always a trade-off between the number of particles needed to represent a PDF, and the performance and memory requirements of a particle filter.

In each recursion step of the particle filter algorithm, every single particle is first updated by the system model and then by the measurement model. The system model changes a new *value* for each particle based on a dynamic model that predicts how the system changes over time. Since the geometric

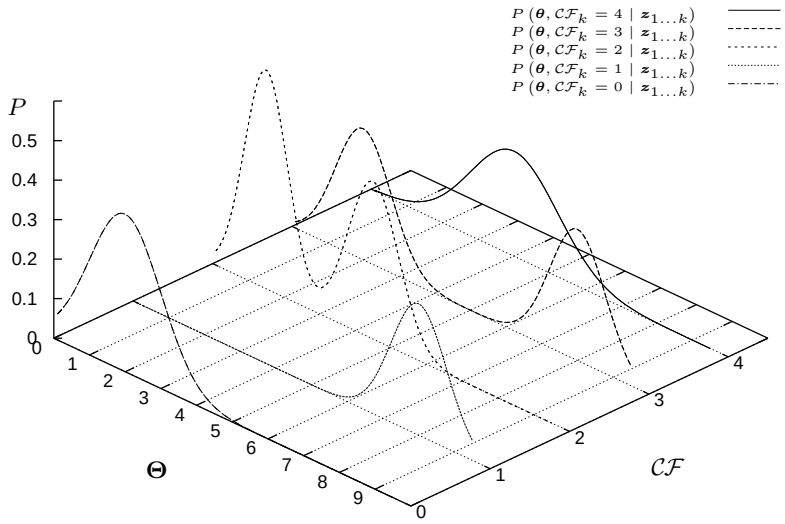


FIGURE 4.8: An example of a probability density function (PDF) of a hybrid joint density, with a one-dimensional continuous geometric parameter $\boldsymbol{\Theta}$ and a one-dimensional discrete state $\mathcal{CF}$. All the information about the system is contained in this one hybrid PDF, while a traditional Kalman filter does not have sufficient degrees of freedom in their Gaussian PDFs to represent the same information.

parameters are static, the system model only makes a prediction of the next CF, given the previous CF and the geometric parameters; hence in this step, particles can "jump" between discrete CFs. The measurement model changes the *weight* of each particle based on the sensor measurements, and in this way corrects each of the predictions made by the system model. This weight corresponds to the probability of a sensor measurement, given the state of the particle. For each measurement, the expected range is given by a PDF. The choice of these PDFs is always somewhat arbitrary, however, it is possible to make a problem specific but physically founded choice, considering the accuracy of (i) the physical sensors used, (ii) the calibration of the sensors, and (iii) the models used to describe the real world.

This section describes the measurement and system models that are used to update the hybrid PDF, while Section 4.4 describes the algorithms of the measurement and system models.

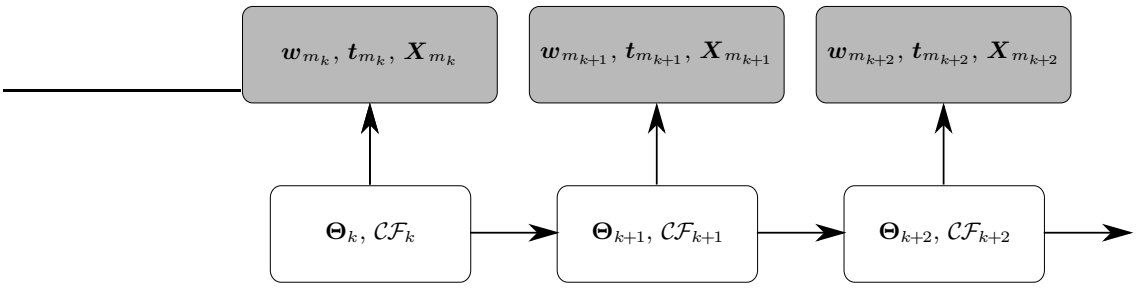


Figure 4.9: The causal relations between the components represented in a sequential Bayesian network. The grey nodes denote observed random variables, while the transparent nodes denote unknown (hidden) random variables. A particle filter is used to update this network recursively.

4.3.2 Measurement model—correction

The correction step uses a measurement model to calculate the hybrid joint density at time step k, given the prediction in Equation (4.22) for the hybrid joint density at time step k:

$$\begin{aligned} P\left(\boldsymbol{\theta}, \mathcal{CF}_k = j \mid \boldsymbol{z}_{1\ldots k}\right) \propto \\ P\left(\boldsymbol{z}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) P\left(\boldsymbol{\theta}, \mathcal{CF}_k = j \mid \boldsymbol{z}_{1\ldots k-1}\right). \end{aligned} \tag{4.14}$$

The measurement model represents the belief in a measurement $\boldsymbol{z}_k$, given the geometric parameters $\boldsymbol{\theta}$ and the $\mathcal{CF}_k = j$, and is defined by:

$$P\left(\boldsymbol{z}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right). \tag{4.15}$$

In this thesis, the measurement model is split up in two separate measurement models:

$$P\left(\boldsymbol{z}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) = P\left(\boldsymbol{X}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) P\left(\boldsymbol{w}_k, \boldsymbol{t}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right), \tag{4.16}$$

with $\boldsymbol{z}_k = \boldsymbol{X}_k, \boldsymbol{w}_k, \boldsymbol{t}_k$ all the measurements taken at timestep k. Both the first and the second measurement model are applied in each correction step. This section first presents the first measurement model, which is based on the pose measurement $\boldsymbol{X}_m$ in Equation (4.6). This first measurement model explains the relation between the pose measurements and the unknown geometrical parameters and CF. This relation is based on the distance between the geometrical features of the objects in contact. Then, this section presents the second measurement model, which is based on the wrench and twist measurements $\boldsymbol{w}_m$ in Equation (4.8) and $\boldsymbol{t}_m$ in Equation (4.7). This second

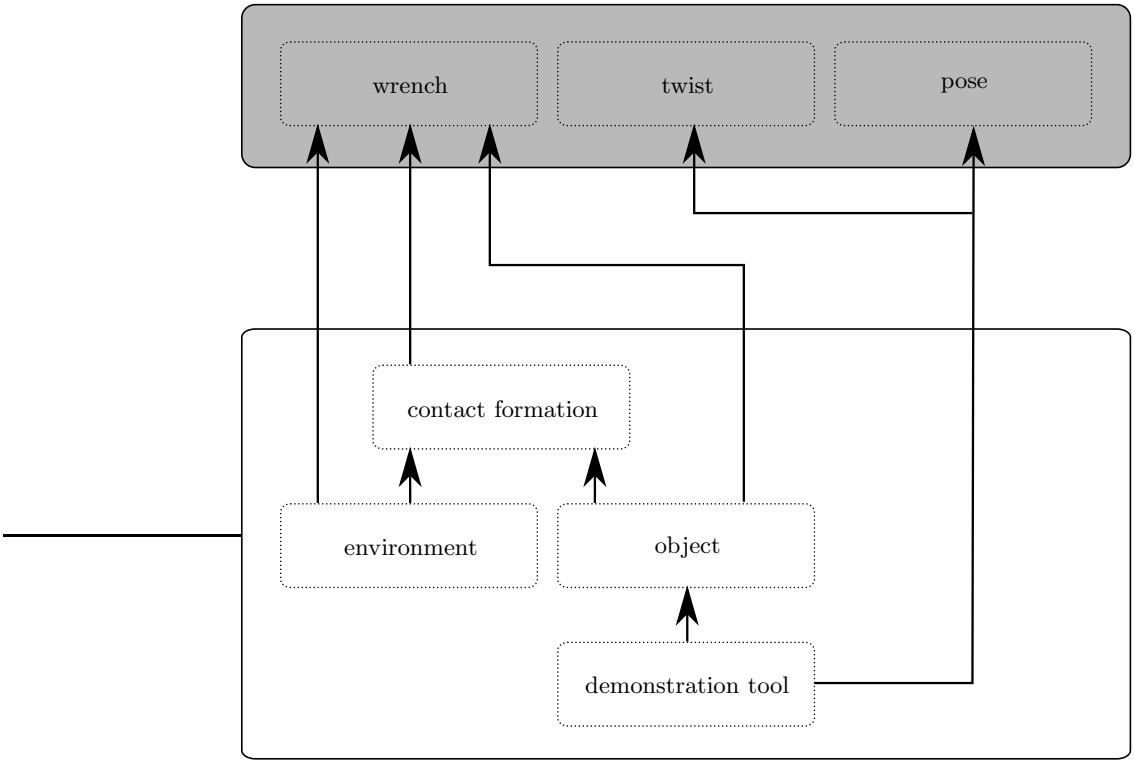


FIGURE 4.10: Detailed view of the relations between the components represented in a Bayesian network. The grey nodes denote observed random variables, while the transparent nodes denote unknown (hidden) random variables. The hybrid character is shown by the discrete contact formation and the continuous pose of the object and the environment, included in one single node.

measurement model explains the relation between the twist and wrench measurements, and the unknown geometrical parameters and CF. This relation is based on the first order kinematic contact constraints of the two objects in contact.

Contact distance measurement model based on pose measurements

The contact distance measurement model expresses that when the manipulated object is in contact with the environmental object, the distance between the objects at the contact points should be zero, thereby closing the kinematic

chain between the objects. The distance between the objects at non-contact points should be greater than zero, expressing that the objects do not penetrate nor contact.

The decomposition of a general CF into ECs (see Section 3.2.1) allows the automatic generation of the contact distance measurement equation for different CF models (Lefebvre, Bruyninckx, and De Schutter 2003). Say the number of possible ECs between the manipulated object and the environmental object at *all possible* CFs is p. This number is a constant that only depends on the geometry of the two contacting objects, not on the current CF. Each CF in the contact state graph can be decomposed into one or more ECs, therefore, the total number of possible ECs is ever greater than the number of CFs, and typically in the range of 10^2 to 10^3. At each $EC_1 \dots EC_p$ the distance between the objects is given by a distance $d_1 \dots d_p$. A new measurement variable $\boldsymbol{d}_m$ is created by combining all these distances into one distance vector:

$$\boldsymbol{d}_m = \begin{bmatrix} d_1 & \dots & d_p \end{bmatrix}^T . \tag{4.17}$$

This new distance measurement variable is a nonlinear function of the pose measurement $\boldsymbol{X}_m$ and the geometric parameters $\boldsymbol{\Theta}$. The contact distance measurement equation expresses the belief in the distance vector, given the current state $\mathcal{CF}_k = j$ and the geometric parameters $\boldsymbol{\theta}$:

$$P\left(\boldsymbol{X}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) = P\left(\boldsymbol{d}_m \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right). \tag{4.18}$$

The difference between the "measured" distance (calculated from the measured pose) and the expected distance (zero at a contact, positive at other ECs) is the measure for the probability of the distance measurement.

Residue measurement model based on wrench-twist measurements

The residue measurement model expresses the consistency between the contact constraints, and the wrench and twist measurements $\boldsymbol{w}_m$ and $\boldsymbol{t}_m$ (Bruyninckx, De Schutter, and Dutré 1993b; Ohwovoriole and Roth 1981). The consistency is expressed by a residue vector $\boldsymbol{r}_m$, which is the part of the measured twist and wrench that is not explained by the first order kinematics of an ideal frictionless contact; it should vanish when the measurements and the model are consistent. For a given pose and CF, the first order kinematics are represented by a wrench space $\mathcal{W}$ and a twist space $\mathcal{T}$. The wrench space contains all possible wrenches that can be applied between the contacting objects at the current pose, and is represented by a matrix $\boldsymbol{W}$ which spans the wrench space. The twist space contains all possible instantaneous twists that maintain the contact, and is represented by a matrix $\boldsymbol{T}$ which spans the twist space. The consistency between $\boldsymbol{w}_m$, $\boldsymbol{t}_m$ and $\mathcal{W}$, $\mathcal{T}$, is expressed by

the 12-dimensional residue vector $\boldsymbol{r}_m$, containing a six-dimensional wrench residue and a six-dimensional twist residue:

$$\boldsymbol{r}_m = \begin{bmatrix} \boldsymbol{I} - \boldsymbol{W}\boldsymbol{W}^{\dagger_{\boldsymbol{K}_w}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} - \boldsymbol{T}\boldsymbol{T}^{\dagger_{\boldsymbol{K}_t}} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_m \\ \boldsymbol{t}_m \end{bmatrix}. \tag{4.19}$$

The operator $^{\dagger_{\boldsymbol{K}_w}}$ represents the weighted pseudo-inverse (Doty, Melchiorri, and Bonivento 1993; Nakamura 1991) of a matrix using a positive definite weighting matrix $\boldsymbol{K}_w$. The residue $\boldsymbol{r}_m$ contains the difference between the measured wrench (twist) and its projection onto the wrench (twist) space. It is a nonlinear function of the pose $\boldsymbol{X}_m$, the CF, the geometrical parameters $\boldsymbol{\Theta}$, and the measured wrench $\boldsymbol{w}_m$ and twist $\boldsymbol{t}_m$. The residue measurement equation expresses the belief in the residue vector, given the current state $\mathcal{CF}_k = j$ and the geometric parameters $\boldsymbol{\theta}$:

$$P\left(\boldsymbol{w}_k, \boldsymbol{t}_k \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) = P\left(\boldsymbol{r}_m \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right). \tag{4.20}$$

The difference between the "measured" residue vector (calculated from the measured wrench, twist and pose) and the expected residue vector (zero) is the measure for the probability of the residue vector.

Consistency versus reciprocity To link the wrench and twist measurements to the geometric parameters in a given CF, two different models were previously presented in literature: the *consistency* model (used in this thesis) and the *reciprocity* model (Bruyninckx, De Schutter, and Dutré 1993b; Ohwovoriole and Roth 1981).

The *reciprocity* model expresses that a measured twist (wrench) produces no work against the wrench (twist) vectors of the wrench (twist) space, and is expressed by the 6-dimensional power vector:

$$\boldsymbol{e}_m = \begin{bmatrix} \boldsymbol{T}^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_m \\ \boldsymbol{t}_m \end{bmatrix}. \tag{4.21}$$

For a s-dimensional twist space and a $(6-s)$-dimensional wrench space, the 6-dimensional power vector contains s components resulting from $\boldsymbol{T}^T\boldsymbol{w}_m$ and $(6-s)$ components from $\boldsymbol{W}^T\boldsymbol{t}_m$. The reciprocity model falls short when comparing two contact situations with a different number of contact constraints. Say one contact situation has a 1-dimensional wrench space and a 5-dimensional twist space, while the other contact situation has a 2-dimensional wrench space and a 4-dimensional twist space. The 6-dimensional power vector of the first case will then contain 1 component related to the wrench space and 5 components related to the twist space, while the 6-dimensional power vector of the second case will then contain 2 components related to the wrench

space and 4 components related to the twist space. To compare the probability on each of the two cases, the probabilities on the components of the power vector are compared. However, one of the compared components will result from the twist space of the first case, and the wrench space of the second case. Imagine that both the twist and wrench measurements are zero, then the reciprocity model will prefer the first case (with the higher dimensional twist space) when the uncertainty on the twist is highest, and the second case (with the higher dimensional wrench space) when the uncertainty on the wrench is highest.

The *consistency* model, which expresses that the measured twist (wrench) is a linear combination of the twist (wrench) base vectors of the twist (wrench) space, does not suffer from the same shortcoming. Its 12-dimensional residue vector in Equation (4.19) always contains 6 components related to the measured twist, and 6 components related to the measured wrench, thus avoiding favoring a higher dimensional twist or wrench space. Therefore, while the reciprocity model is only appropriate to distinguish between two geometric models within the same CF, the consistency model can also be used to distinguish between different CFs.

4.3.3 System model—prediction

The prediction step uses the system model to make a prediction for the hybrid joint density at time step k, given the hybrid joint density at time step $k-1$:

$$\begin{aligned} &P\left(\boldsymbol{\theta}, \mathcal{CF}_k = j \mid \boldsymbol{z}_{1\ldots k-1}\right) = \\ &\sum_i P\left(\mathcal{CF}_k = j \mid \boldsymbol{\theta}, \mathcal{CF}_{k-1} = i\right) P\left(\boldsymbol{\theta}, \mathcal{CF}_{k-1} = i \mid \boldsymbol{z}_{1\ldots k-1}\right). \end{aligned} \tag{4.22}$$

This simplified prediction step is valid because the estimated geometric parameters $\boldsymbol{\Theta}$ are time-invariant, and only the state $\mathcal{CF}$ changes during time (Gadeyne, Lefebvre, and Bruyninckx 2005). The system model is a state transition prediction function that expresses the belief in a CF transition from a $\mathcal{CF}_{k-1} = i$ at time step $k-1$, to a $\mathcal{CF}_k = j$ at time step k, given the geometric parameters $\boldsymbol{\theta}$, and is defined by:

$$P\left(\mathcal{CF}_k = j \mid \boldsymbol{\theta}, \mathcal{CF}_{k-1} = i\right). \tag{4.23}$$

To predict the next CF out of hundreds of theoretically possible next CFs, the topological information from the objects' contact state graph is used. Between each two CFs that are connected in the contact state graph, a direct CF transition is possible, while between two unconnected CFs a transition is only possible through one or more other CFs. Therefore the system model assumes that in one update, a CF can only "jump" to one of its neighboring CFs. This assumption based on the contact state graph drastically reduces the

number of possible next CFs; all the non-neighboring transition probabilities in Equation (4.23) are zero, and hence Equation (4.22) contains few non-zero terms. The probability of a CF transition to a neighboring CF is proportional to the probability of contact distances at the ECs of the neighboring CF, as explained in the previous section.

The accurate prediction of CF transitions is important for the filter's performance, since the number of particles needed for a given uncertainty on the geometric parameters is directly linked to the quality of the prediction step.

Jump in contact state graph The contact state graph represents the adjacency relation between CFs by the connection of two nodes by an arc. Two CFs are adjacent when a direct transition between them is possible, without going through any other CF. The system model directly applies this adjacency relation by only allowing a CF transition between adjacent CFs in Equation (4.32). This knowledge about the adjacency relation drastically reduces the search field for possible next CFs. Typically a CF has about 2–10 neighboring CFs, while the total number of CFs easily exceeds 200–300. However, it is possible that more than one CF transition to a neighboring CF occurs in a very short time interval, making it appear as a "jump" in the contact state graph to a non-neighboring CF. In the example in Figure 4.11, this could be an apparent jump between CF_1 and CF_3, without going through CF_2. When starting at CF_1, the system model (which models the CF transitions) will not be able to directly recognize the new CF_3, since it is not adjacent to CF_1. However, a path in the contact state graph exists from CF_1 to CF_2 to CF_3, where the number of contact constraints increases gradually. This path allows the estimation to converge from CF_1 to CF_3, by first going through CF_2. The estimation converges because given the pose wrench and twist measurements at the new CF_3, the intermediate CF_2 is more probable than the initial CF_1, because it has more contact constraints than CF_1 to explain the new contact forces measured in CF_3. This guides the estimation from CF_1 to CF_2, and subsequently from CF_2 to CF_3.

In general, the presented estimation approach can deal with a jump in the contact state graph, as long as there exists a path from the initial CF to the new CF with a gradually increasing or decreasing number of contact constraints. For a human manipulating an object, this type of event is quite "natural" and therefore not uncommon in programming by human demonstration. A human uses visual feedback to for example place an object on a table, jumping from no contact to a face-face contact, without first making a vertex-face contact, an edge-face contact and finally the face-face contact. However, these events which are natural for a human, always have a gradually increasing or decreasing number of contact constraints, and are therefore no problem for the presented estimation approach. When this is not the case and

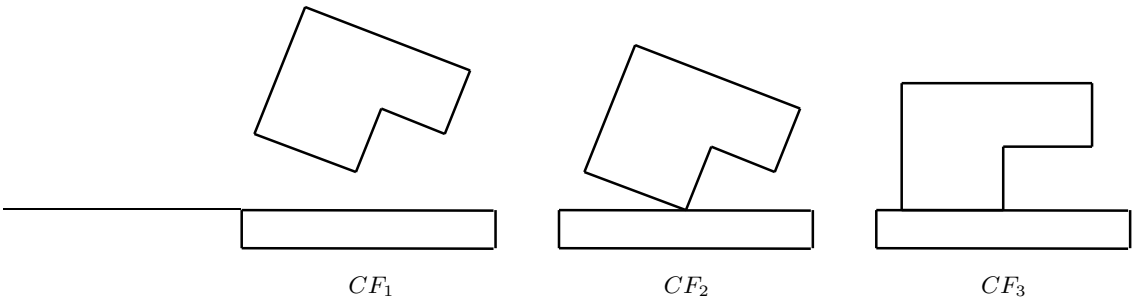


Figure 4.11: Moving directly from CF_1 to CF_3 corresponds to a jump in the contact state graph. However, the estimated CF will converge to CF_3 by first passing through CF_2.

the jump in the contact state graph is too large, the localization in the contact state graph is lost, resulting in the "kidnapped robot" problem known from mobile robotics research (Fox, Burgard, Dellaert, and Thrun 1999). Adding some random CFs to the particle filter can help to cope with this problem.

4.4 Algorithms

This section describes the implementation of the system model in Equation (4.23) and the measurement models in Equation (4.18) and Equation (4.20), presented in the previous section. The efficiency of the implementation is achieved by exploiting application-specific knowledge and using application-specific "shortcuts", such as:

- assuming probabilities to be independent,
- not calculating or only approximating probabilities that are not relevant, such as the probabilities of contact distances at ECs not adjacent to the current CF,
- developing numerically efficient algorithms that take advantage of the orthonormal nature of the matrices obtained from a singular value decomposition, and
- choosing easy to evaluate PDFs based on normal distributions and uniform distributions.

The particle filter algorithms use a 13-dimensional hybrid joint density PDF, consisting of a 12-dimensional continuous parameter and one discrete state.

The continuous parameter $\boldsymbol{\Theta}$ contains geometric parameters that represent the unknown pose of the environmental object relative to a world reference, and the unknown pose of the manipulated object relative to the demonstration tool, as shown in Figure 4.7. The geometry of both the environmental object and the manipulated object is known. The discrete state $\mathcal{CF}$ contains the CF between the manipulated object and the environment, and can be any of the many hundreds of CFs in a complete contact state graph of the contacting objects.

4.4.1 Contact distance measurement equation

The pose measurement model in Equation (4.18) expresses the belief in the distance vector $\boldsymbol{d}_m$, which contains the distances between the involved objects at all possible ECs. Calculating all these distances for each pose between the objects would be numerically expensive. Therefore, only the distances at ECs that are relevant in the assumed CF[2] are calculated. The relevant distances are the distances at the ECs of the assumed CF, as well as the ECs directly connected to the assumed CF by an edge. When in the example in Figure 4.12 the assumed CF (say CF_a) only includes EC_2, the distances $d_1 \dots d_3$ are calculated, and not the distances $d_4 \dots d_6$. When the assumed CF (say CF_b) includes EC_2 as well as EC_4, the distances $d_1 \dots d_5$ are calculated, and not the distance d_6. In the former case, distance d_4 is not calculated, and thus the probability on CF_a does not decrease when d_4 gets smaller. However, when d_4 gets smaller, the probability on CF_b (which is a neighbor of CF_a) increases, and CF_b still becomes more probable than CF_a, although d_4 is not calculated in the case of CF_a.

The probabilities of the distances $d_1 \dots d_p$ are not independent, but are all linked by the probability of the measured pose $\boldsymbol{X}_m$, since all distances are a function of the measured pose. The nonlinear relation between the probabilities of the pose and the distances is expensive to calculate, and therefore the distances d_i are assumed to be independent. The probability of the distance vector, as expressed in Equation (4.18), can then be calculated by:

$$P\left(\boldsymbol{d}_m \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) = \prod_{i=1}^{p} P\left(d_i \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right), \tag{4.24}$$

where $P\left(d_i \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right)$ expresses the belief in a distance measurement at the i^{th} EC.

The distance calculation at the ECs is helped by the use of spherical bounding boxes around the elements of the ECs. The center and radius of the spherical bounding box for a vertex, an edge and a face are given in

[2]The assumed CF is the CF that is given in Equation (4.18), the $\mathcal{CF}_k = j$

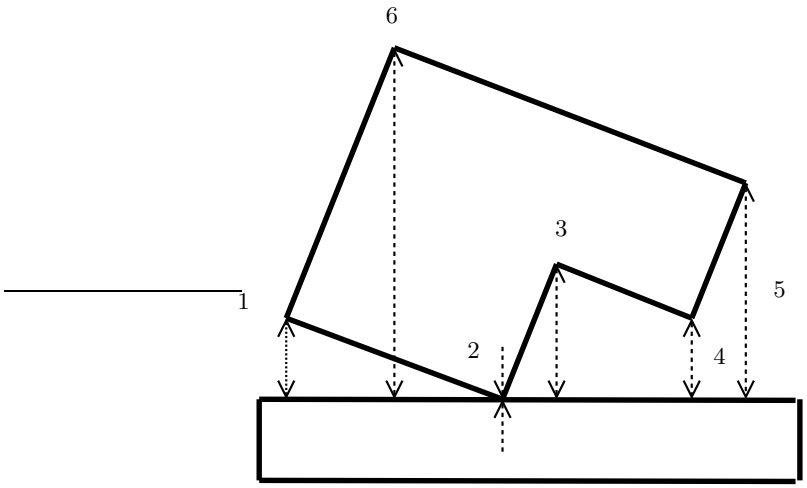


Figure 4.12: For the calculation of the distance vector $\boldsymbol{d}_m$ only the distances at the ECs of assumed CF and the ECs that are directly connected to the assumed CF by an edge, are calculated.

Table 4.1. Only when two bounding boxes intersect, the exact distance at an EC is calculated. When the bounding boxes do not intersect (meaning that the elements of the EC are far apart) the exact distance has no real influence on the filter's behavior, and is approximated by the distance between the bounding boxes.

While previously presented approaches such as (Lefebvre, Bruyninckx, and De Schutter 2003) and (Gadeyne, Lefebvre, and Bruyninckx 2005) suffer from the simplification that all features (edges and faces) are considered infinite, the use of spherical bounding boxes avoids this drawback. This is illustrated in Figure 4.13, where a top and a side view of a vertex and face are shown. The side view illustrates how the distance d_2 between the vertex and the face is too small because the face is considered infinite. The distance d_1 to the bounding box however is a good approximation of the real distance d_r between the vertex and the face.

The PDF of a distance d_i has the physical meaning of measurement noise on the distance measurement. However, the distance is not measured by a physical sensor, but calculated from the pose measurement and therefore the "true" PDF is not known. The PDF for the distance at an EC that is part of the current CF, we choose a Gaussian distribution, defined by a mean and a 2 σ boundary, as shown by the dashed line in Figure 4.14. The mean is

	center	radius
vertex	$\boldsymbol{p}$	0
edge	$\frac{\boldsymbol{p}_1 + \boldsymbol{p}_2}{2}$	$\frac{\|\|\boldsymbol{p}_1 - \boldsymbol{p}_2\|\|}{2}$
face	$\frac{\sum_{i=0}^{n} \boldsymbol{p}_i}{n}$	$max(\|\|\text{center} - \boldsymbol{p}_i\|\|)$, with $i = 1 \dots n$

TABLE 4.1: The spherical bounding box around a feature is defined by a center point and a radius.

chosen zero because the expected value of a distance at an EC that is part of the current CF is zero. The 2 σ boundary is chosen based on both the measurement noise on the pose measurement, and the uncertainty due to modelling or calibration errors. The PDF for the distance at an EC that is not part of the current PC, is inspired by a Gaussian distribution, and is shown by the continuous line in Figure 4.14. This PDF has its maximum likelihood at the intersection with the dashed Gaussian. The mean is chosen larger than zero because the expected value of a distance at an EC that is not part of the current CF is larger than zero. The covariance of this PDF is also defined by the noise on the pose measurements, and the uncertainty due to modelling or calibration errors. Figure 4.14 shows the chosen PDFs for the experiment presented in Chapter 7. Note that because only rigid objects are modelled, the uncertainty on the pose measurements would increase when the objects are more flexible.

4.4.2 Residue measurement equation

The residue measurement model in Equation (4.20) expresses the belief in the residue vector $\boldsymbol{r}_m$, which is a measure for the consistency between the first order kinematics of the contact model, and the measured twist and wrench. This section presents an efficient approach to calculate the residue vector. The approach exploits the orthonormal nature of the vectors obtained from a *singular value decomposition* (SVD), to replace two weighted pseudo-inverses by two more efficient transposes, for a specific choice of the weighting matrices.

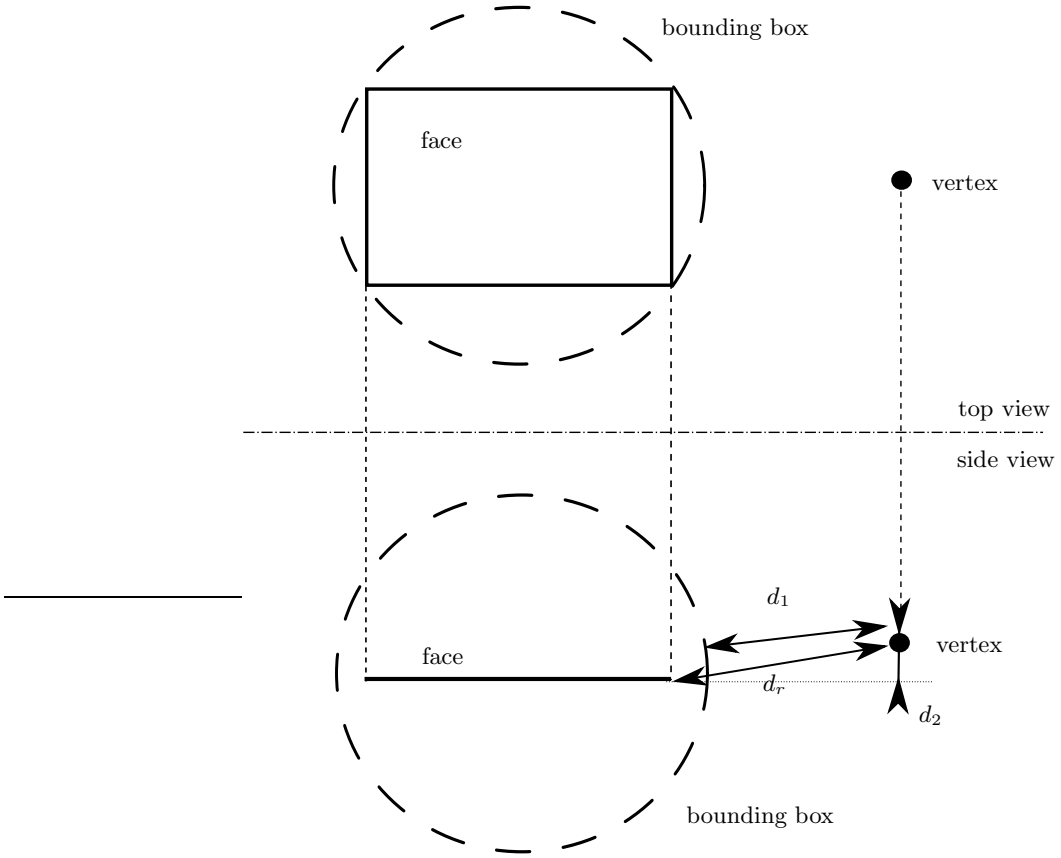


Figure 4.13: Spherical bounding boxes around each contact feature improve the contact distance calculation, were previously presented approaches suffer from the simplification that all features are infinite.

Implementation with a single SVD

The first order kinematics of an ideal frictionless contact are represented by the local wrench and twist space at the CF. As explained in Section 3.2.2, the twist and wrench space are calculated by decomposing the CF into ECs, and are represented by a basis $\boldsymbol{T}$ for the twist space and a basis $\boldsymbol{W}$ for the wrench space. The twist space $\boldsymbol{T}$ and the wrench space $\boldsymbol{W}$ are calculated simultaneously at the cost of a single SVD. On top of that, the calculation of the residue $\boldsymbol{r}_m$ in Equation (4.19) requires two more weighted pseudo-inverses $\boldsymbol{W}^{\dagger_{\boldsymbol{K}_w}}$ and $\boldsymbol{T}^{\dagger_{\boldsymbol{K}_t}}$ that each require another SVD. This brings the total cost

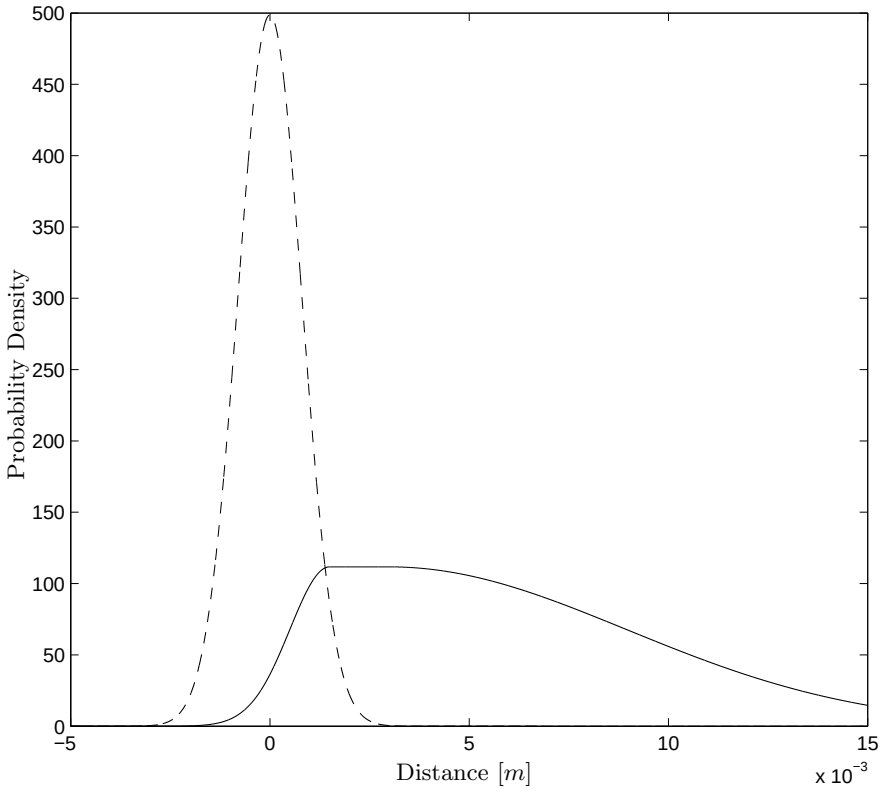


FIGURE 4.14: The probability density function on the distance at an EC between two objects in contact (dashed line) and at an EC between two objects not in contact (continuous line).

for the residue measurement model to three SVDs. These calculations are numerically expensive. Therefore this section presents a method to reduce the total numerical cost to a single SVD, by choosing the weighting matrices $\boldsymbol{K}_w$ and $\boldsymbol{K}_t$ in a specific way. The choice of the weighting matrices is motivated using the physical interpretation of the matrices.

Choosing a diagonal weighting matrix $\boldsymbol{K}_w$, the product of a wrench space with this weighting matrix corresponds to a change of units[3] of the wrench

[3]Example of changing the units of a wrench vector using a diagonal weighting matrix:

$$\begin{bmatrix} \boldsymbol{w}_1[kN] \\ \boldsymbol{w}_2[mN] \\ \boldsymbol{w}_3[N] \\ \boldsymbol{w}_4[Ndm] \\ \boldsymbol{w}_5[Nkm] \\ \boldsymbol{w}_6[Nm] \end{bmatrix} = \begin{bmatrix} 0.001 & & & & & \\ & 1000 & & & & \\ & & 1 & & & \\ & & & 10 & & \\ & & & & 0.001 & \\ & & & & & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_1[N] \\ \boldsymbol{w}_2[N] \\ \boldsymbol{w}_3[N] \\ \boldsymbol{w}_4[Nm] \\ \boldsymbol{w}_5[Nm] \\ \boldsymbol{w}_6[Nm] \end{bmatrix}$$

space:

$$\boldsymbol{W}'_{CF} = \boldsymbol{K}_w \boldsymbol{W}_{CF}, \tag{4.25}$$

in which $\boldsymbol{W}'_{CF}$ spans the same wrench space as $\boldsymbol{W}_{CF}$, using different units. The reciprocity condition, as previously discussed in Equation (3.14), imposes that when the units of the wrench space change with $\boldsymbol{K}_w$ between $\boldsymbol{W}$ and $\boldsymbol{W}'$, the units of the twist space change with $\boldsymbol{K}_w^{-1}$ between $\boldsymbol{T}$ and $\boldsymbol{T}'$:

$$\boldsymbol{W}'^T \boldsymbol{T}' = (\boldsymbol{K}_w \boldsymbol{W})^T (\boldsymbol{K}_w^{-1} \boldsymbol{T}) = \boldsymbol{0}. \tag{4.26}$$

Therefore, when choosing the weighting matrices:

$$\boldsymbol{K}_t = \boldsymbol{K}_w^{-1}, \tag{4.27}$$

the SVD of $\boldsymbol{W}'_{CF}$ instead of Equation (3.10), results in a wrench space $\boldsymbol{W}'$ weighted with $\boldsymbol{K}_w$ and a twist space $\boldsymbol{T}'$ weighted with $\boldsymbol{K}_w^{-1}$:

$$\boldsymbol{W}'_{CF} = \boldsymbol{K}_w \boldsymbol{W}_{CF} = \begin{bmatrix} \boldsymbol{W}' & \boldsymbol{T}' \end{bmatrix} \boldsymbol{S}' \boldsymbol{V}'^T, \tag{4.28}$$

The calculation of the residue vector is independent of the bases used to represent the wrench and twist space. However, choosing the numerically orthonormal matrices $\boldsymbol{W}'$ and $\boldsymbol{T}'$ to represent the wrench and twist space, the pseudo-inverse of the wrench and twist space is reduced to a transpose:

$$\begin{aligned} \boldsymbol{W}'^\dagger &= \boldsymbol{W}'^T, \\ \boldsymbol{T}'^\dagger &= \boldsymbol{T}'^T. \end{aligned} \tag{4.29}$$

This results in the calculation of the residue in Equation (4.19) with a numerical cost of only one SVD in Equation (4.28):

$$\boldsymbol{r}_m = \boldsymbol{K} \begin{bmatrix} \boldsymbol{I} - \boldsymbol{W}' \boldsymbol{W}'^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} - \boldsymbol{T}' \boldsymbol{T}'^T \end{bmatrix} \boldsymbol{K}^{-1} \begin{bmatrix} \boldsymbol{w}_m \\ \boldsymbol{t}_m \end{bmatrix}, \tag{4.30}$$

in which

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_w^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{K}_w \end{bmatrix}. \tag{4.31}$$

The importance of reducing the number of SVDs required becomes clear when using profiling tools to measure the computational time spent on each of the sub-algorithms. Reducing the number of SVDs from 3 to 1 decreased the computational cost of the overall filter with 55 [%], as shown in Table 4.2. Using the efficient algorithm still 60 [%] of the overall computation time is spent on the single SVD.

The PDF of a residue $\boldsymbol{r}_m$ has the physical meaning of measurement noise on the residue vector, and hence the measurement noise on the part of the measured twist and wrench that is not explained by the first order contact

	% spent on SVD	#particles per sec.
before	82	40,500
after	60	90,000

TABLE 4.2: The efficiency increase by eliminating two SVD evaluations.

kinematics. A good approximation for this PDF is the uncertainty on the twist and wrench measurements. This uncertainty is a combination of the sensor noise and the modelling or calibration errors. The uncertainty on the wrench measurement is dominated by the fact that friction is neglected in the measurement model, and is therefore chosen based on an estimated friction coefficient and the expected maximum wrench measurement. The uncertainty on the twist measurement depends on the stiffness of the objects in contact; only rigid objects are modelled, and hence the uncertainty on the twist increases when the objects are more flexible. Both the PDF on the twist measurement and the wrench measurement are represented by a 6-dimensional Gaussian distribution.

4.4.3 System equation

The system update calculates the prediction density at time step k, given the estimated variables at time step $k-1$, as described by Equation (4.22). This prediction step is based on the probability of a transition from a $\mathcal{CF}_{k-1} = i$ to a $\mathcal{CF}_k = j$. The adjacency relationship between CFs, which is defined by arcs connecting CFs in the contact state graph (see Section 3.3), indicates if a direct transition between two CFs is possible without passing through any other CF (see Figure 3.8). If two CFs i and j are not adjacent, a direct transition between these CFs is not possible, and hence the probability of a transition is defined by:

$$P\left(\mathcal{CF}_k = j \mid \boldsymbol{\theta}, \mathcal{CF}_{k-1} = i\right) = 0. \tag{4.32}$$

For two adjacent CFs i and j, the probability of a transition is *chosen* to depend on the distance vector $\boldsymbol{d}_m$ between the two objects, as defined in Equation (4.17). The smaller the distance at the ECs of the two objects, the more likely a transition will occur:

$$P\left(\mathcal{CF}_k = j \mid \boldsymbol{\theta}, \mathcal{CF}_{k-1} = i\right) = P\left(\boldsymbol{d}_m \mid \boldsymbol{\theta}, \mathcal{CF}_k = j\right) \tag{4.33}$$

The choice to make a CF transition depend on the contact distance is arbitrary, but is a good approximation of the reality. A more advanced system model could for example also take the direction of the manipulator's velocity into account. The probability on the contact distance is calculated identically

to the contact distance probability in Equation (4.18). The uncertainty on the contact distances for the system model are chosen identical to the uncertainties on the contact distances for the pose measurement model presented in Section 4.4.1. By choosing the uncertainties of the system model and the pose measurement model identical, the system model increases the likelihood for a particle to "jump" to a neighboring CF, when that particle is more likely to be "accepted" in the neighboring CF by the pose measurement model; hence a particle is less likely to be eliminated by "jumping" to the wrong neighboring CF. Note that because the probability of a CF transition only depends on the involved CFs and the pose measurement, the approach does not require a separate training phase to assign probabilities to CF transitions.

4.4.4 Software

Framework

The algorithms presented in this thesis are implemented within the framework offered by the open source C^{++} Bayesian Filtering Library (BFL) (Gadeyne 2001). BFL offers a unifying framework for all recursive Bayesian filters, such as Kalman filters, extended Kalman filters and particle filters. It provides efficient implementations of various filter algorithms. To implement the presented particle filter in BFL, only the application specific parts need to be provided; the system model, the measurement model, and the hybrid PDF. BFL provides a C^{++} class structure to define the desired interface of these parts. All functionality that is not application specific is provide by BFL; the functionality to update the PDF, keep a bookkeeping of particles, re-sample the particles, keep, etc. More details about this software framework can be found in (Gadeyne 2005).

Scope of implementation

The implementation of the presented particle filter approach applies to any two rigid *polyhedral* objects in contact; hence the implementation is not limited to the cube-in-corner example presented in Chapter 7, but can be applied to any two rigid polyhedral objects in contact, such as for example putting a book in a bookshelf in between two other books. The implementation requires a description of the geometry of the objects, given by a list of the features of the object (vertices, edges and faces) and the three-dimensional coordinates of the vertices. In addition to the geometric description, the implementation also requires a complete contact state graph of the objects. This graph is automatically generated using the software provided by Jing Xiao of the University of North Carolina at Charlotte and her research group. Their algorithms generate a complete contact state graph between two polyhedral objects con-

taining hundreds of CFs, within seconds. For the presented cube-in-corner example a single goal-contact-relaxation subgraph is sufficient to describe all possible contacts between the cube and the corner; most applications however require multiple goal-contact-relaxation subgraphs to be merged into one single contact state graph.

The measurement models do not model *friction* forces, *inertia* forces or the *deformation* of the objects in contact. While the particles can deal with a limited difference between the real world and the modelled world, large differences result in a failure of the estimation process. Experimental results provide a rough estimation of the allowed difference between the real and the modelled world. When the friction and inertia forces remain less than 20 [%] of the total measured contact force, and when the position of the contact points is changed less than 2 [%] by the deformation of the objects, the filters are still able to function properly.

The sensors integrated on the demonstration tool have a limited accuracy and maximum load. Therefore *size* of the polyhedral objects used is bounded; contacts between very small objects ($< \pm 0.1$ $[m]$) cannot be distinguished using the sensor measurements, and very large objects ($> \pm 1.0$ $[m]$) would overload the sensors. This is not a limitation of the implementation, but a limitation of sensors used in the experimental setup.

Comparison with previous approach

The effectiveness of the presented approach is shown when comparing the obtained results with the results previously obtained by Gadeyne et al. in (Gadeyne, Lefebvre, and Bruyninckx 2005). The main improvements with respect to the approach in (Gadeyne, Lefebvre, and Bruyninckx 2005) are found in the number of particles needed to estimate the hybrid state, and the performance of the presented algorithms.

- **Number of particles** The approach in (Gadeyne, Lefebvre, and Bruyninckx 2005) does not use the topological information of a contact state graph, and therefore requires to consider all 245 possible CFs at each timestep. The approach in this thesis only requires to evaluate the *neighboring* CFs, which reduces the number of CFs to consider to an average of 5 CFs at each timestep. To allow the same uncertainty on the geometrical parameters, the approach in (Gadeyne, Lefebvre, and Bruyninckx 2005) would require the same number of particles per CF, resulting in 49 times more particles. For the presented experiment this would result in $980,000$ required particles instead of the $20,000$ particles that are required in the experiments presented in Chapter 7.
- **Performance** The approach in (Gadeyne, Lefebvre, and Bruyninckx 2005) is able to process $3,000$ particles per second on a 1.1 $[GHz]$ laptop.

Assuming this older hardware is a factor 4 slower than the 2 $[GHz]$ laptop used for the experiments in this thesis, the presented approach is still 7.5 times faster in processing particles ($90,000$ particles per second compared to $4 \times 3,000$ particles).

Combining both the decreased number of required particles and the increased performance, the presented approach is 367 times faster than the approach in (Gadeyne, Lefebvre, and Bruyninckx 2005). For the experiments presented in Chapter 7 this would result in processing sensor measurements at $\frac{1}{82}$ $[Hz]$, while the implementation presented in this thesis is capable of processing sensor measurements in realtime at 4.5 $[Hz]$. In addition to this increased processing speed, the presented approach considers the geometric features of the object (faces and edges) to be finite, and replaced the reciprocity measurement model by the improved consistency measurement model.

4.5 Limitations

This section discusses the limitations of programming by human demonstration in general and by the presented approach in particular.

4.5.1 Human demonstration

Available sensors Humans can quickly and efficiently demonstrate complex manipulation tasks using their fine manipulation skills, physical insight and experience. During the demonstration, humans use many of their "sensors" to successfully complete a given task. The strategy applied by a human to complete the task depends on the sensors available; imagine the difference in strategy when putting a peg in a hole with your eyes open or with your eyes closed. In programming by human demonstration, sensors on a demonstration tool measure different task parameters. In order to successfully repeat the task with a robot, using the same strategy as the human demonstrator used during the demonstration, the demonstration tool and the robot should be equipped with similar sensors as used by the human demonstrator. In practice however, for a given demonstration tool, the human demonstrator should only use his own sensors when the demonstration tool is equipped with similar sensors. For a demonstration with the presented demonstration tool, which is equipped with pose and wrench sensors, the demonstrator should rely on the pose and wrench sensing of his arm, and not use his advanced vision and tactile sensors. To achieve an optimal result, the human demonstrator should also keep the type of strategy in mind that the estimators are trying to extract, which is a sequence of CFs. To overcome this limitation and make the approach more intuitive for a human demonstrator, a camera could

be mounted on the demonstration tool. Using a new measurement model, the detected features in the camera image could then help to estimate the geometrical parameters and recognize the current CF.

Minimal demonstration Inherently, the task description extracted from a human demonstration is limited by the information extracted from and contained in the demonstration. While a general task specification describes an appropriate action for each possible situation that can occur during the execution, a task description obtained by a single demonstration only contains an action for the events that occurred during that one demonstration. A possible solution lies in the combination of multiple demonstrations into one task description, but even then it is a challenge to include an appropriate action for each possible situation. Not only the number of included discrete events in the demonstration is minimal; also the continuous sensor data contain minimal information because a human is very efficient at achieving a task with minimal power-exchange with the environment. This poses extra challenges to extract a task description from the information available.

4.5.2 Performance versus uncertainty

Using the algorithms presented in this chapter, the particle filter is able to process more than $20,000$ particles at a rate of 4 $[Hz]$. This is sufficient for the online estimation of geometrical parameters and CF segmentation, with an initial uncertainty in the range of 1 $[cm]$ on the translations and 0.5 $[rad]$ on the rotations. Experimental results show that for a larger initial uncertainty on the parameters, more particles are needed to obtain a correct CF recognition and estimate of the geometrical parameters. As a consequence, when the uncertainty on the parameters is larger, more computational power is needed to achieve online estimation. Therefore the performance of the particle filter not only depends on the efficiency of the algorithms, but also on the initial uncertainty on the parameters. However, even when the initial uncertainty on the parameters is large, which requires many extra particles, once the estimation of the geometrical parameters converges, the number of particles can be drastically reduced to increase the overall performance of the filter. The smallest number of particles that is required for a given uncertainty on the geometrical parameters, can only be obtained experimentally.

4.5.3 Information in measurements

The measurements gathered from a human demonstration are not always informative about all geometrical parameters. The example in Figure 4.15 shows a cube that is moved downwards on a *straight line*. This motion gives no information about the position of the vertical side of the corner. When

combining this situation with a large uncertainty on the geometrical parameters, this results in the measurement models favoring the wrong CF: the v_1-f_1 CF with the vertical wall is favored over the CF with no contacts, and hence the position of the corner is estimated as the dotted line instead of the "true" full line. This failure of the filter is caused by the pose measurements that are perfectly *compatible* with the v_1-f_1 CF, while the wrench and twist measurements are *not incompatible* with the v_1-f_1 CF. The problem lies in the residue measurement model that only checks for components of the wrench and twist measurements that are *incompatible* with the CF. When moving downwards, the measured wrench is zero, and therefore the incompatible component of the wrench is also zero. As soon as the motion of the object has a component in the direction of the $v_1 - f_1$ CF, the residue measurement will again favor the no-contact CF. Although this presents an important problem when executing a compliant motion task with a robot manipulator, in programming by human demonstration this is only a minor problem, because a human is not capable of moving an object on a perfectly straight line. A possible solution lies in an adapted measurement model with a more "human-like" contact reasoning, which is to *expect* a certain force when two objects are in contact.

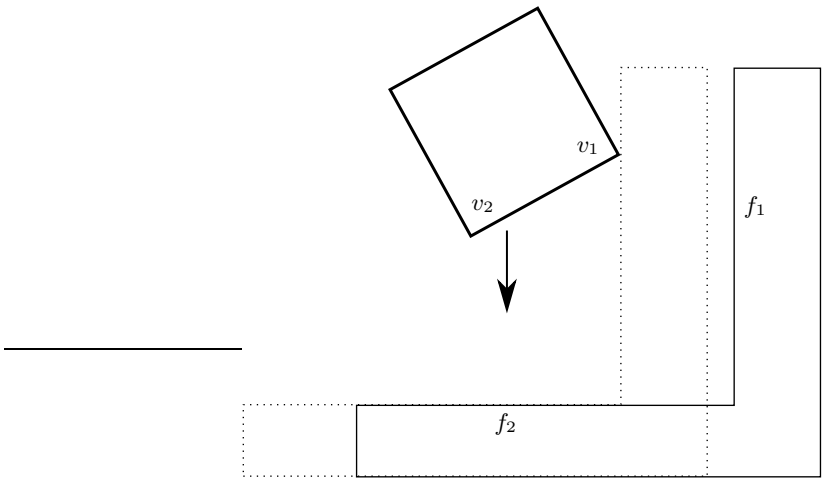


Figure 4.15: The pose, wrench and twist measurements are not always informative about all geometrical parameters. When for example moving on a *straight line* this could result in the measurement models favoring the wrong v_1-f_1 CF over the correct no-contact CF.

4.6 Conclusion

This chapter presents a contribution to the task specification process for sensor-controlled robot systems that physically interact with the environment. While the previous chapter discussed task specification based on an off-line compliant path planner, this chapter uses programming by human demonstration to obtain a task specification. Both methods result in the same type of task specification: a sequence of poses $\boldsymbol{X}_1 \ldots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \ldots CF_m$. The human demonstration approach starts with a demonstration step, where a human demonstrates a compliant task by directly manipulating an object in contact with its environment, using a demonstration tool. During the demonstration, the Krypton 6D optical system measures the pose and twist of the manipulated object, while a wrench sensor measures the interaction forces between the contacting objects. Subsequently, in the interpretation step, these measurements are interpreted using sequential Bayesian estimation techniques.

Similar to the more familiar localization, tracking and recognition problems in mobile robotics, the approach *simultaneously* recognizes CF transitions and estimates geometrical parameters. The approach is based on the Bayesian sequential Monte Carlo method, or particle filter, and can cope with hybrid (partly discrete, partly continuous) joint posterior variables containing both the CF and geometrical parameters of the environment. While previously presented results in this field only allowed a limited number of CFs and CF transitions, this approach scales the search space to *all possible CFs* between the contacting objects, using the topological information contained in a contact state graph. This extension in combination with new efficient algorithms, allow the *realtime* simultaneous recognition of CFs and estimation of geometric parameters. The approach applies to convex as well as to concave polyhedral objects with a known geometry, but at an unknown pose, and is able to efficiently recognize the CF at each step of a human demonstration out of many hundreds of possible CFs. The approach does not use any prior knowledge about the probability of CF transitions. Training the CF transitions in the contact state graph using sensor data gathered during multiple demonstrations is an interesting topic for future research.

In this chapter the particle filter is discussed in the context of programming by human demonstration. However, the particle filter can also be used to process sensor data collected during the execution of a compliant motion task. When processing these data online, during the execution, the estimators can provide feedback to the robot controller about the current CF and estimate geometrical parameters related to the robot manipulator and its environment. The performance of the presented algorithms is sufficient to process all sensor data in realtime. Using the estimators online to provide feedback for the robot controller is discussed in Chapter 6.

Chapter 5

Compliant task generator

An object in motion will be heading in the wrong direction, an object at rest will be in the wrong place.

Gerrold's First and Second Law of Infernal Dynamics

5.1 Introduction

This chapter presents the *Compliant Task Generator*, an approach for the automatic conversion of a geometric path into a force based task specification for a compliant robot controller, which is illustrated in Figure 5.1 (center). As presented in the two previous chapters, a geometric path can be obtained from an off-line compliant path planner, or from a human demonstration. Both approaches result in a geometric compliant path description as a sequence of six-dimensional poses and their corresponding contact formations. However, such a path only contains geometric and topological information, and hence it is not directly suited as an input to a hybrid force/velocity controller. This chapter describes the automatic conversion of a geometric path represented

by the planner or demonstration task primitives, into a force based task specification for the hybrid controller represented by the controller task primitives. The conversion is achieved by *manually adding* information about the desired magnitude of the wrench and twist to the planner primitives. The direction of the wrench and twist, as well as the local wrench and twist spaces are automatically derived from the given CF and pose. This allows us to specify a complex compliant path using an off-line planner or a human demonstration, and immediately execute it on a real robot manipulator under active force control. The approach is verified by real world experiments, presented in Chapter 7.

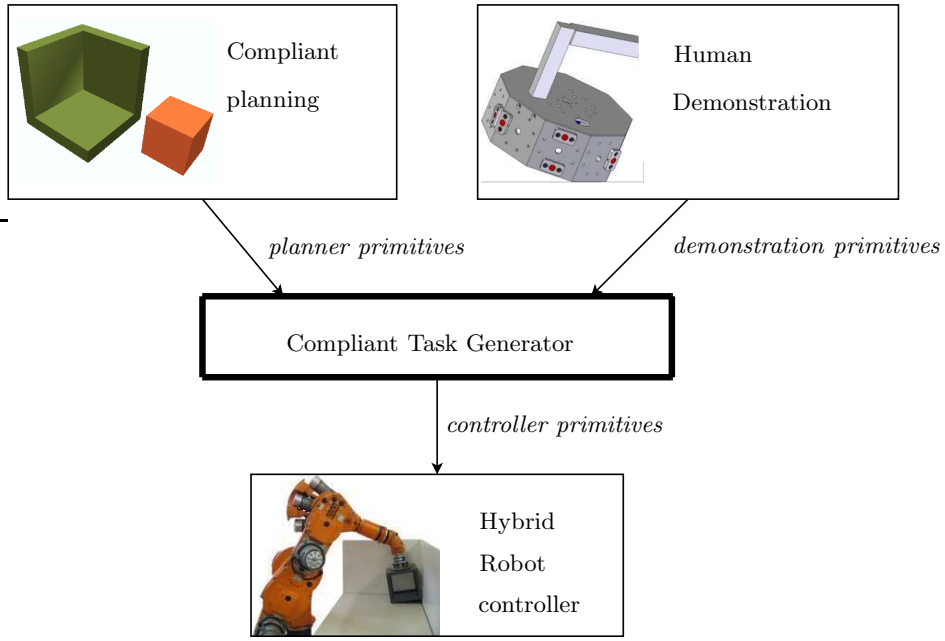


Figure 5.1: This thesis presents two high level approaches for task specification in active compliant motion: compliant path planning and programming by human demonstration. The compliant task generator converts the task specification into instantaneous setpoints for the hybrid robot controller. This section discusses the compliant task generator.

The chapter is organized as follows. The first section describes the task primitives from the compliant motion planner, the human demonstration, and the hybrid robot controller. The next section presents the automatic generation of the controller task primitives, based on the planner or demonstration

task primitives. Finally, this chapter discusses the invariance of the presented method to changes in units, reference frame and scale.

5.2 Task description primitives

This section describes the "interfaces" to the compliant path planner, programming by human demonstration and the hybrid force/velocity controller. The interfaces are defined by the task primitives that describe the input and output to the compliant task generator, as shown in Figure 5.1. The output task primitives or controller task primitives are the task specification for the controller; they specify the same path as the input task primitives which are the planner or demonstration task primitives, but in a form the controller understands.

5.2.1 Planner task primitives

The compliant path planner presented in Chapter 3 takes a start and a goal configuration as an input, and automatically generates a compliant path connecting the two given configurations. This planner is based on the given *geometric model* of the contacting objects and therefore the planner task primitives only describe the geometry of the calculated compliant path, in terms of a sequence of poses $\boldsymbol{X}_1 \dots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \dots CF_m$. Each two poses $\boldsymbol{X}_i$ and $\boldsymbol{X}_{i+1}$ are at the same or at neighboring contact formations, as shown in Figure 3.14. The planner provides no information about the desired velocity of the objects and the desired force interaction between the objects.

5.2.2 Demonstration task primitives

The programming by human demonstration approach presented in Chapter 4 exploits the human manipulation skills to obtain a description of a compliant motion task. Identical to the planner task primitives, the compliant path generated by human demonstration is described by a sequence of poses $\boldsymbol{X}_1 \dots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \dots CF_m$. In addition to the geometric description of the compliant path, the demonstration task primitives also contain wrench and twist measurements, describing the contact interaction and the kinematics of the compliant motion.

5.2.3 Controller task primitives

The chosen controller strategy in this thesis is the *Hybrid Control Paradigm* (HCP) (Raibert and Craig 1981; Mason 1981; Fisher and Mujtaba 1992; Duffy

1990), one of the three major force control paradigms together with *Impedance Control* (Hogan 1985; Hogan 1987) and the *Parallel Force Control* (Chiaverini and Sciavicco 1993). The HCP assumes a *geometric* interaction model. In HCP terminology, an object in contact with its environment has s degrees of freedom (DOF) which are twist controlled, and $(6-s)$ DOF which are wrench controlled. The s twist controlled DOF are described by a s-dimensional twist space $\mathcal{T}$ and the $(6-s)$ wrench controlled DOF are described by a $(6-s)$-dimensional wrench space $\mathcal{W}$.

The controller task primitives are a desired wrench $\boldsymbol{w}_d$ to specify the contact wrench between the manipulated object and its environment, a desired twist $\boldsymbol{t}_d$ to specify the velocity of the manipulated object in its environment, and a desired pose $\boldsymbol{X}_d$. The controller task primitives also contain the local wrench and twist spaces $\boldsymbol{W}$ and $\boldsymbol{T}$.

5.3 Compliant task generator

This section describes the core of the approach, the automatic conversion of a geometric path represented by the planner or demonstration task primitives ($\boldsymbol{X}_1 \ldots \boldsymbol{X}_n$ and $CF_1 \ldots CF_m$), into a force based task specification for the hybrid controller represented by the controller task primitives ($\boldsymbol{w}_d$, $\boldsymbol{t}_d$, $\boldsymbol{X}_d$, $\boldsymbol{W}$ and $\boldsymbol{T}$). The conversion is achieved by manually adding information about the desired magnitude of the wrench and twist to the planner primitives. The direction of the wrench and twist, as well as the local wrench and twist spaces are automatically derived from the given CF and pose.

5.3.1 Extending planner primitives

In contrast to the demonstration primitives, the planner primitives contain no information about the desired wrench and twist. This section describes how wrench and twist information is added to the planner primitives to generate the controller primitives. To add this information to the planner primitives we *manually* specify the desired magnitudes of the wrench and twist. However, it is not possible to directly specify the magnitude of a twist or wrench, because both contain components of two distinct subspaces. Therefore we also need to specify two norms to give a meaning to each of the specified wrench and twist magnitudes. The desired directions of the wrench and twist are derived *automatically* from the given CFs and poses.

Specification of magnitudes

The desired contact force level and execution speed are specified by the user in the form of a *magnitude* and a *norm* for the desired twist and for the desired

wrench. To obtain a specification that is invariant with respect to changes in reference frame, physical units or scale (Manes 1992; Blajer 1997; Doty, Melchiorri, and Bonivento 1993; De Luca and Manes 1994), we choose two norms with a physical meaning. The norm for the desired twist is defined by a generalized inertia matrix $\boldsymbol{M}$, while the norm for the desired wrench is defined by a generalized compliance matrix $\boldsymbol{C}$. When the inertia and compliance matrices define a norm for the desired twist and wrench, the specified magnitudes have the physical meaning of a kinetic and a potential energy:

$$\|\boldsymbol{t}_d\|_{\boldsymbol{M}} = \frac{\boldsymbol{t}_d^t \boldsymbol{M} \boldsymbol{t}_d}{2} = E_{kin}, \tag{5.1}$$

$$\|\boldsymbol{w}_d\|_{\boldsymbol{C}} = \frac{\boldsymbol{w}_d^t \boldsymbol{C} \boldsymbol{w}_d}{2} = E_{pot}. \tag{5.2}$$

The magnitudes and norms can be directly specified by an inertia, a compliance and two energy levels, or indirectly by a magnitude for all components of the desired twist and wrench.

Direct specification The inertia and compliance can be chosen to reflect the true dynamical properties of the system defined by the manipulator, the manipulated object and the environment. In this case the specified magnitudes for the desired twist and wrench reflect the real kinetic and potential energy stored in the system during the execution of the compliant path.

The inertia and compliance can also be chosen as an arbitrary norm for the desired wrench and twist, invariant with respect to changes of reference frame, physical units or scale. In this case, the inertia and compliance represent a virtual system. During the execution of the compliant path, the specified magnitudes have the meaning of energy levels in the virtual system.

Indirect specification In some cases it is more intuitive to define the magnitudes $\bar{v}$ and $\bar{\omega}$ of the translational and rotational components of the desired twist, and the magnitudes $\bar{f}$ and $\bar{\tau}$ of the force and torque components of the desired wrench. The desired magnitude $\bar{v}$ specifies the desired norm of the translational component of a twist with a zero rotational component, while the desired magnitude $\bar{\omega}$ specifies the desired norm of the rotational component of a twist with a zero translational component. The force and torque magnitudes have an analogue meaning, but for the wrench. The magnitudes of the torque and translational velocity are specified at a given point on the manipulated object, because only when expressed at a certain point they have a meaning. From the specification of these magnitudes at a given point, we derive two invariant norms as an inertia and compliance matrix, and two magnitudes as the kinetic and potential energy.

The inertia matrix is defined by:

$$\boldsymbol{M} = m \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & l_t^2 \boldsymbol{I}_{3\times3} \end{bmatrix}, \tag{5.3}$$

where $m = 1\ [kg]$ and where l_t is the characteristic length for the twist:

$$l_t = \frac{\bar{v}}{\bar{\omega}}\ [m], \tag{5.4}$$

and the magnitude of the twist is defined by the kinetic energy:

$$E_{kin} = m\bar{v}^2. \tag{5.5}$$

In the same way, the compliance matrix is defined by:

$$\boldsymbol{C} = c \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 1/l_w^2 \boldsymbol{I}_{3\times3} \end{bmatrix}, \tag{5.6}$$

where $c = 1\ [m/N]$ and where l_w is the characteristic length for the wrench:

$$l_w = \frac{\bar{\tau}}{\bar{f}}\ [m], \tag{5.7}$$

and the magnitude of the wrench is defined by the potential energy:

$$E_{pot} = c\bar{f}^2. \tag{5.8}$$

Specification of directions

The desired twist $\boldsymbol{t}_d$, pose $\boldsymbol{X}_d$ and wrench $\boldsymbol{w}_d$ are automatically calculated using the set of poses $\boldsymbol{X}_1 \ldots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \ldots CF_m$, together with the manually specified magnitudes and norms for the desired twist and wrench.

Twist The desired twist $\boldsymbol{t}_d$ at time $t \in [t_i, t_{i+1}[$, to move from $\boldsymbol{X}_i$ to $\boldsymbol{X}_{i+1}$ with a magnitude E_{kin}, is calculated in two steps. First we define a constant twist $\boldsymbol{t}_i$ to move from $\boldsymbol{X}_i$ to $\boldsymbol{X}_{i+1}$:

$$\boldsymbol{t}_i = \begin{bmatrix} \boldsymbol{v}_i^T & \boldsymbol{\omega}_i^T \end{bmatrix}^T, \tag{5.9}$$

and

$$\begin{bmatrix} [\boldsymbol{\omega}_i\times] & \boldsymbol{v}_i \\ \boldsymbol{0} & 0 \end{bmatrix} = \log\left(\boldsymbol{X}_i^{-1}\boldsymbol{X}_{i+1}\right) / \left(t_{i+1} - t_i\right). \tag{5.10}$$

The logarithm of a homogeneous transformation matrix (Murray, Li, and Sastry 1994, Section 3.2) is used to interpolate between two discrete setpoints

of the planner. This results in an interpolation along the screw axis[1]. The $[\ \times]$ operator is defined in Section 3.2.2. In the second step we scale this constant twist $\boldsymbol{t}_i$ to the desired twist $\boldsymbol{t}_d$, so that its magnitude equals E_{kin}:

$$\boldsymbol{t}_d = s_t \boldsymbol{t}_i. \tag{5.11}$$

The scaling factor s_t has no units and is defined by the magnitude E_{kin}:

$$E_{kin} = \frac{(s_t \boldsymbol{t}_i)^T \boldsymbol{M} (s_t \boldsymbol{t}_i)}{2}. \tag{5.12}$$

This results in:

$$s_t = \sqrt{\frac{2E_{kin}}{\boldsymbol{t}_i^T \boldsymbol{M} \boldsymbol{t}_i}}. \tag{5.13}$$

The direction of the desired twist changes discontinuously between two planner setpoints, while its magnitude remains constant. Replacing the linear interpolation between the pose setpoints by a smooth interpolation avoids such discontinuous changes of the direction.

Pose The desired pose $\boldsymbol{X}_d$ at time $t \in [t_i, t_{i+1}[$, between $\boldsymbol{X}_i$ and $\boldsymbol{X}_{i+1}$ is defined by the integration of the desired twist $\boldsymbol{t}_d$, using the exponential function (Murray, Li, and Sastry 1994, Section 3.2):

$$\boldsymbol{X}_d = \boldsymbol{X}_i \exp\left(\begin{bmatrix} [\boldsymbol{\omega}_d \times] & \boldsymbol{v}_d \\ \mathbf{0} & 0 \end{bmatrix} (t - t_i) \right), \tag{5.14}$$

with

$$\boldsymbol{t}_d = \begin{bmatrix} \boldsymbol{v}_d^T & \boldsymbol{\omega}_d^T \end{bmatrix}^T. \tag{5.15}$$

Wrench The desired wrench $\boldsymbol{w}_d$ at time $t \in [t_i, t_{i+1}[$, is calculated in two steps. First we decompose all PCs of the contact formation CF_{j+1} at the pose $\boldsymbol{X}_{i+1}$, into ECs; an EC is positioned at each of the boundary points of the polygonal contacting area, as described in Section 3.2.1 and illustrated in Figure 3.6. The number of ECs at this CF is called r. We position a wrench vector $\boldsymbol{w}_k$ at each EC_k, with $k = 1 \ldots r$. In a local frame with the origin at the contact point and the x-axis along the contact normal, oriented from the environment towards the manipulated object, each wrench is represented by a wrench with a zero torque component:

$$\boldsymbol{w}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T. \tag{5.16}$$

[1] This results in discrete changes of the direction of the desired twist at each timestep t_i. An alternative approach could use a spline function to interpolate a smooth trajectory between the given poses.

These wrenches at the ECs only define the direction of the desired wrench and not the magnitude; therefore their magnitude is of no importance. The sum of all wrenches at the ECs is called $\boldsymbol{w}_i$, and defines the direction of the desired wrench:

$$\boldsymbol{w}_i = \sum_{k=1}^{r} \boldsymbol{w}_k. \tag{5.17}$$

This approach to define the direction of the desired wrench based on the ECs of the PC is an arbitrary *choice*. An alternative approach to define the direction of the desired wrench could be to place a wrench with zero torque at the centroid of the polygon bounding the contacting area.

In the second step we scale this wrench $\boldsymbol{w}_i$ to the desired wrench $\boldsymbol{w}_d$, so that its magnitude equals E'_{pot}:

$$\boldsymbol{w}_d = s_w \boldsymbol{w}_i. \tag{5.18}$$

The scaling factor s_w has no units and is defined by the magnitude E'_{pot}:

$$E'_{pot} = \frac{(s_w \boldsymbol{w}_i^T)\boldsymbol{C}(s_w \boldsymbol{w}_i)}{2}. \tag{5.19}$$

This results in:

$$s_w = \sqrt{\frac{2E'_{pot}}{\boldsymbol{w}_i^T \boldsymbol{C} \boldsymbol{w}_i}}. \tag{5.20}$$

The magnitude E'_{pot} is chosen to depend on the *contact strength* of the contact formation CF_{j+1} at $\boldsymbol{X}_{i+1}$ and the specified magnitude E_{pot}:

$$E'_{pot} = \dim(\mathcal{W}) E_{pot}. \tag{5.21}$$

The contact strength is equal to the number of wrench controlled degrees of freedom, denoted by $\dim(\mathcal{W})$. This results in a magnitude for the desired wrench, where at a point contact (vertex-face, face-vertex or edge-edge) it is smaller than at a line contact (edge-face or face-edge), and at a line contact it is smaller than at a plane contact (face-face). This choice results in a contact wrench that increases with the number of contact constraints, and therefore the part of the total contact wrench that each of the contact constraints resists, remains constant.

5.3.2 Extending demonstration primitives

Where the planner primitives required the manual specification of the magnitudes and norms of the desired twist and wrench, the demonstration primitives already contain twist and wrench measurements from the human demonstration. However, these real world measurements are not completely compatible

with the contact model of the hybrid force/velocity controller. This incompatibility is due to sensor inaccuracies, geometric uncertainties and real world effects such as friction and inertia that are not modelled by the hybrid controller paradigm. To obtain the desired twist $\boldsymbol{t}_d$, the desired wrench $\boldsymbol{w}_d$ and the desired pose $\boldsymbol{X}_d$, the measured wrench is projected onto the twist and wrench space, and interpolated for each controller cycle.

Twist The desired twist $\boldsymbol{t}_d$ at time $t \in [t_i, t_{i+1}[$, to move from $\boldsymbol{X}_i$ to $\boldsymbol{X}_{i+1}$, based on the measured twists $\boldsymbol{t}_{m_{t_i}}$ and $\boldsymbol{t}_{m_{t_{i+1}}}$, is defined by:

$$\boldsymbol{t}_d = \boldsymbol{T}\boldsymbol{T}^{\dagger_{\boldsymbol{K}_M}} \left(\frac{\boldsymbol{t}_{m_{t_{i+1}}} - \boldsymbol{t}_{m_{t_i}}}{t_{i+1} - t_i}(t - t_i) + \boldsymbol{t}_{m_{t_i}} \right). \tag{5.22}$$

The inertia matrix $\boldsymbol{K_M}$ is used to obtain an invariant projection of the twist onto the twist space, as explained in Section 3.2.2.

Pose The desired pose $\boldsymbol{X}_d$ at time $t \in [t_i, t_{i+1}[$, between $\boldsymbol{X}_i$ and $\boldsymbol{X}_{i+1}$ is defined by the integration of the desired twist $\boldsymbol{t}_d$, using the exponential function (Murray, Li, and Sastry 1994, Section 3.2)

$$\boldsymbol{X}_d = \boldsymbol{X}_i \exp\left(\begin{bmatrix} [\boldsymbol{\omega}_d\times] & \boldsymbol{v}_d \\ \mathbf{0} & 0 \end{bmatrix} (t - t_i) \right), \tag{5.23}$$

with

$$\boldsymbol{t}_d = \begin{bmatrix} \boldsymbol{v}_d^T & \boldsymbol{\omega}_d^T \end{bmatrix}^T. \tag{5.24}$$

Wrench The desired wrench $\boldsymbol{w}_d$ at time $t \in [t_i, t_{i+1}[$, to move from $\boldsymbol{X}_i$ to $\boldsymbol{X}_{i+1}$, based on the measured wrenches $\boldsymbol{w}_{m_{t_i}}$ and $\boldsymbol{w}_{m_{t_{i+1}}}$, is defined by:

$$\boldsymbol{w}_d = \boldsymbol{W}\boldsymbol{W}^{\dagger_{\boldsymbol{K}_C}} \left(\frac{\boldsymbol{w}_{m_{t_{i+1}}} - \boldsymbol{w}_{m_{t_i}}}{t_{i+1} - t_i}(t - t_i) + \boldsymbol{w}_{m_{t_i}} \right). \tag{5.25}$$

The compliance matrix $\boldsymbol{K_C}$ is used to obtain an invariant projection of the wrench onto the wrench space, as explained in Section 3.2.2.

5.3.3 Defining wrench and twist controlled subspaces

To obtain bases $\boldsymbol{W}$ and $\boldsymbol{T}$ for the wrench space and the reciprocal twist space at time $t \in [t_i, t_{i+1}]$, when moving from $\boldsymbol{X}_i$ at CF_j, to $\boldsymbol{X}_{i+1}$ at CF_{j+1}, we choose to use the contact information of CF_{j+1}, and not the current contact formation CF_j. This choice allows us to break unilateral contact constraints under velocity control, and add unilateral contact constraints under force control: when breaking a contact constraint, the twist space of CF_{j+1} is

higher dimensional than the twist space CF_j, allowing a velocity controlled motion to break the contact; when creating a new contact constraint, the wrench space of CF_{j+1} is higher dimensional than the wrench space of CF_j, allowing a force controlled motion to add the contact. To obtain a base for the wrench and twist spaces at CF_{j+1}, the method described in Section 3.2.2 is applied.

5.4 Discussion

5.4.1 Invariance of the method

Specification in terms of physical properties

The presented method is based on physical properties such as magnitudes defined by a kinetic or potential energy, and norms defined by an inertia and a compliance. Also, the reciprocity condition between twist and wrench space, is based on produced work in the interaction between twists of $\mathcal{T}$ and wrenches of $\mathcal{W}$, and hence is a physical property. Since physical properties do not vary under changes of reference frame, physical units or scale, the method is invariant with respect to these changes.

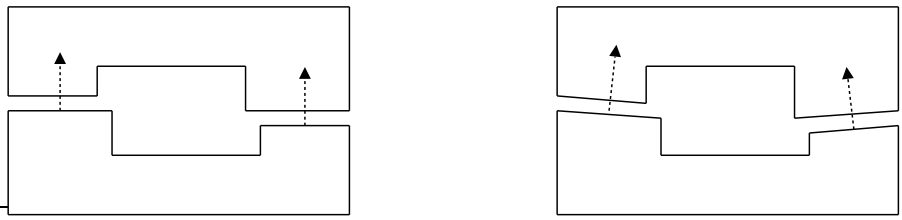
Invariant twist and wrench projection

The representation of the twist and wrench subspaces is not invariant with respect to changes of reference frame, physical units or scale. Depending on these changes, the numerical SVD algorithm will produce different bases that span the same subspace, as shown in Equation (3.10). However, all operations applied to twists and wrenches in this chapter are invariant with respect to the specific representation of the subspaces. For the projection of twists and wrenches onto the twist and wrench space, we use a *weighted* pseudo inverse which minimizes the kinetic or potential energy in the projection error, as shown in Equation (3.22). These kinetic and potential energy are physical properties and hence they are invariant.

Selection of twist and wrench spaces

Starting from a set of vectors $\boldsymbol{W}_{CF}$ that span the wrench space, the SVD in Equation (3.10) results in a matrix $\boldsymbol{U}$ in Equation (3.11) which contains a base for both the wrench and the twist space. A threshold ϵ for the singular values determines whether a column of $\boldsymbol{U}$ belongs to the wrench space or to the twist space. In the planar example in Figure 5.2, the wrench space of a) is two-dimensional while the wrench space of b) is three-dimensional, because the contacting faces are parallel in a) but not in b). The threshold

ϵ determines when the contacting faces are considered parallel or not, and hence when the transition between a two- and a three-dimensional wrench space occurs. This threshold depends on the units and reference frame of $\boldsymbol{W}_{CF}$.



a b

FIGURE 5.2: The threshold ϵ for the singular values defines if a motion degree of freedom belongs to the twist or the wrench space. In the planar example, a) has a two-dimensional wrench space, while b) has a three-dimensional wrench space.

5.4.2 Applicability

The presented method applies to polyhedral objects in contact. This limitation is imposed by the method to find all ECs in the current contact state, which is aimed at polyhedral objects. The ECs are used to calculate the twist and wrench spaces, and to define the direction of the desired wrench. To extend the approach to general curved objects, an algorithm is required to find all ECs between two curved objects in contact. While the contact between polyhedral objects can be a point contact, a line contact and face contact, the contact between curved objects is always associated with one or more contact *points* (Bruyninckx 1995). This suggests that the extension to general curved objects could be straightforward. Tang and Xiao (2006) already presented work on the automatic generation of a contact state graph for curved objects.

For the motion between two intermediate poses $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k+1}$ from the compliant path planner or the human demonstration, the presented method uses the wrench and twist space at the pose $\boldsymbol{X}_{k+1}$ to approximate the wrench and twist space during the whole trajectory. This approximation is valid when the distance between each two subsequent poses is small ($< \pm 0.01\ [m]$ or $< \pm 0.1\ [rad]$), *or* when the wrench and twist subspaces do not change significantly during the compliant motion. The first condition depends on the output of the compliant path planner or the human demonstration, and is easily fulfilled by choosing a small step size in the task specification process.

Decreasing the step size in the task specification has no negative consequences for the task generator. The second condition depends on the type of CF and the motion within the CF. During a compliant motion in a face-face contact, the wrench and twist spaces are invariant, but during a compliant motion in a $2\times$ vertex-face contact the wrench and twist spaces vary. As a result, the presented method is only generally applicable when the step size in the task specification process is chosen small.

5.4.3 Choice of Parameters

The compliant motion generated by our approach depends on a number of arbitrarily chosen parameters.

Planner Parameters

For the off-line planning of the compliant path, we define a step size, using a translational and rotational component $\Delta trans$ and Δrot. A smaller step size will result in a smaller translational and rotational distance between subsequent poses $\boldsymbol{X}_i$ and $\boldsymbol{X}_{i+1}$ of the planner output. The effect of this step size is discussed in Section 7.2.5.

Compliant Task Generator Parameters

For the compliant task generation, we define a desired magnitude for the twist and wrench, and two invariant norms. As explained before, the inertia matrix $\boldsymbol{M}$ and the compliance matrix $\boldsymbol{C}$ can correspond to the physical properties of the experimental setup, can be arbitrary, or can be calculated indirectly from specified magnitudes for the force, torque, translational velocity and rotational velocity components of the wrench and twist.

The compliant task generator is also based on arbitrary choices that are not "tunable" by the user. In particular, the magnitude of the desired wrench depends on the specified magnitude for the wrench, and is also chosen proportional to the dimension of the wrench space. The direction of the desired wrench is based on the ECs of the contact formation.

5.5 Conclusions

This chapter presents the *Compliant Task Generator*: an approach based on (possibly virtual) physical properties to link the planning and controller efforts in active compliant motion. A compliant path planner or a human demonstration provides a geometrical path in the form of a set of six-dimensional poses $\boldsymbol{X}_1 \ldots \boldsymbol{X}_n$ and their corresponding contact formations $CF_1 \ldots CF_m$. The hybrid compliant controller expects a desired twist $\boldsymbol{t}_d$, pose $\boldsymbol{X}_d$ and wrench $\boldsymbol{w}_d$

at each time step, together with their twist and wrench spaces $\mathcal{T}$ and $\mathcal{W}$. The conversion of the discrete planner and demonstration primitives into a continuous path represented by the controller primitives, is processed separately for the twist and wrench space. The conversion within the twist space uses a desired magnitude and a norm for the twist, while the conversion within the wrench space uses a desired magnitude and a norm for the wrench. These magnitudes and norms can be directly specified by the user, or indirectly through the desired magnitudes of the twist and wrench components.

The *Compliant Task Generator* is the first general and automated approach that links planning and human demonstration to controller efforts in active compliant motion. It applies to any compliant motion between rigid polyhedral objects with a known geometry, and is more general and simple than previously presented ad-hoc (Bruyninckx and De Schutter 1997) or rule-based methods. Moreover, the method is invariant with respect to changes of reference frame, scale and physical units. A task-specific input of two magnitudes and norms (or four magnitudes for the twist and wrench components) is sufficient to specify the desired dynamic interaction between the manipulated object and its environment, and allows the fully automated conversion of the planner or demonstration primitives into the controller primitives. The result is the immediate execution of an off-line planned or demonstrated compliant path by a manipulator, under active force control. In a real world experiment presented in Chapter 7, the approach proved both efficient and effective for all provided compliant paths, including complex contact formations and contact formation transitions.

Chapter 6

Task execution

I long to accomplish a great and noble task, but it is my chief duty to accomplish humble tasks as though they were great and noble.

Helen Keller

6.1 Introduction

In compliant motion the force interaction with constraining environmental objects helps a manipulator—or an object held by a manipulator—to overcome uncertainties associated with the task. Mason was one of the first to publish work on compliance and force control (Mason 1981). He introduced the *Task Frame Formalism* (TFF), an intuitive and manipulator independent tool for the specification of force controlled robot tasks in the *Hybrid Control Paradigm* (HCP) (Raibert and Craig 1981). However, the TFF is limited to *orthogonal* twist and wrench components, and therefore cannot model more complex CFs involving multiple simultaneous contacts. In this thesis we apply the HCP with *reciprocal* twist and wrench spaces (Lipkin and Duffy 1988;

Duffy 1990). For each configuration of the objects in contact, we define local twist and wrench spaces to model the contact kinematics; this makes it possible to model the kinematics of *any* possible CF or CF compliant motion. Real world experiments are discussed in Chapter 7.

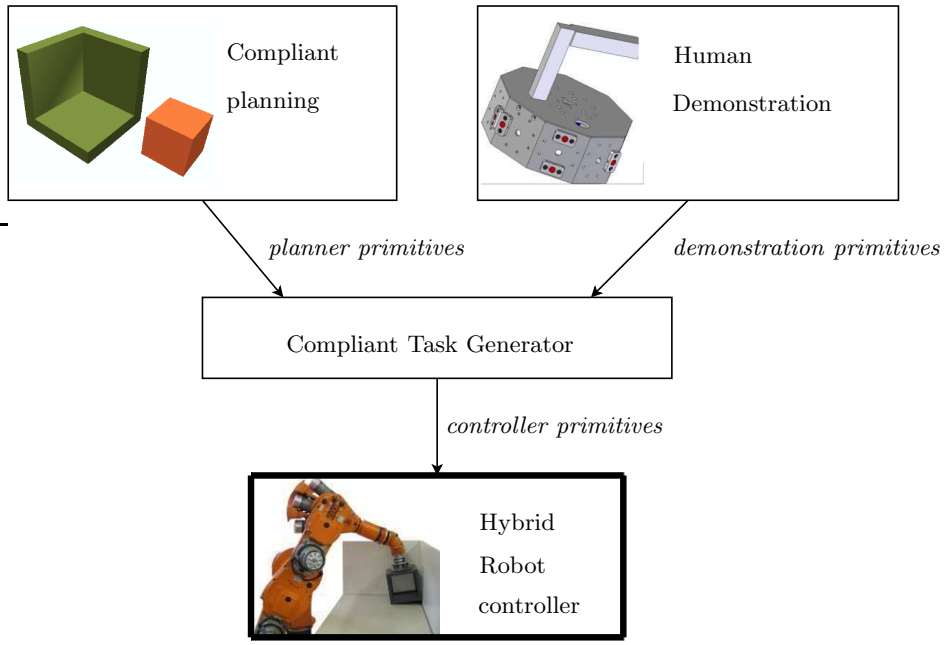


Figure 6.1: This thesis presents two high level approaches for task specification in active compliant motion: compliant path planning and programming by human demonstration. The compliant task generator converts the task specification into instantaneous setpoints for the hybrid robot controller. This section discusses the hybrid robot controller.

The previous chapter described the conversion of a geometric path obtained from the compliant motion planner or a human demonstration, into a force based task specification for the hybrid controller. This chapter explains how this force based task specification for the hybrid controller is used for the execution of the compliant motion tasks under active force control, as shown in Figure 6.1 (bottom). The first section discusses the hybrid force/velocity controller and its control algorithms to convert the given task specification into an instantaneous velocity setpoint for a velocity controlled robot manipulator. The extension of the hybrid controller with online CF estimation and a CF feedback loop is discussed in the next section. The CF feedback

loop provides a discrete feedback to the hybrid controller, allowing the hybrid controller to select an optimal control strategy based on the actual and the desired CF.

6.2 Implementation of the hybrid controller

This section discusses the hybrid controller that converts the desired twist $\boldsymbol{t}_d$, the desired pose $\boldsymbol{X}_d$ and the desired wrench $\boldsymbol{w}_d$ to a control twist $\boldsymbol{t}_c$ for the manipulator. The approach applies to a velocity controlled manipulator, identical to the approach in (De Schutter and Bruyninckx 2000), which is industrial practice. A proportional feedback loop in the twist space controls the desired twist $\boldsymbol{t}_d$ and pose $\boldsymbol{X}_d$, while a second proportional feedback loop in the wrench space controls the desired wrench $\boldsymbol{w}_d$.

6.2.1 Pose and twist controller in twist space

The desired twist $\boldsymbol{t}_d$ is directly applied in the twist space, while the desired pose $\boldsymbol{X}_d$ is used together with the measured pose $\boldsymbol{X}_m$ as a pose feedback in the twist space. Together they define the resulting velocity for the manipulated object in the s-dimensional twist space $\mathcal{T}$. The contributions of the desired velocity and the pose feedback are represented using s-dimensional coordinate vectors $\boldsymbol{u}_t$ and $\boldsymbol{u}_X$:

$$\boldsymbol{u}_t = \boldsymbol{T}^{\dagger_{\boldsymbol{M}_c}} \, \boldsymbol{t}_d, \tag{6.1}$$

$$\boldsymbol{u}_X = K_X^{FB} [\frac{1}{sec}] \, \boldsymbol{T}^{\dagger_{\boldsymbol{M}_c}} \, \boldsymbol{t}_\Delta^{FB}. \tag{6.2}$$

The pose difference $\boldsymbol{t}_\Delta^{FB}$ between the measured pose $\boldsymbol{X}_m$ and the desired pose $\boldsymbol{X}_d$ is calculated similarly to Equation (5.10), using:

$$\boldsymbol{t}_\Delta^{FB} = \left[\begin{array}{cc} \Delta\boldsymbol{p}^{FB} & \Delta\boldsymbol{\theta}^{FB} \end{array} \right]^T, \tag{6.3}$$

and

$$\left[\begin{array}{cc} \left[\Delta\boldsymbol{p}^{FB}\times\right] & \Delta\boldsymbol{\theta}^{FB} \\ \boldsymbol{0} & 0 \end{array} \right] = \log\left(\boldsymbol{X}_n^{-1}\boldsymbol{X}_d\right). \tag{6.4}$$

The scalar K_X^{FB} with units $\frac{1}{sec}$ represents the proportional pose feedback constant in $\mathcal{T}$. Matrix $\boldsymbol{T}^{\dagger_{\boldsymbol{M}_c}}$ is the weighted pseudo-inverse of $\boldsymbol{T}$, and the weighting matrix $\boldsymbol{M}_c$ is an inertia matrix. As explained in Section 3.2.2, the weighted pseudo-inverse minimizes the projection error when projecting the 6-dimensional twist and pose difference into the s-dimensional twist space. The norm for the twist has the physical meaning of kinetic energy in an object with mass distribution $\boldsymbol{M}_c$ (Bruyninckx and De Schutter 1996). The weighted pseudo inverse is calculated numerically using the SVD algorithm.

6.2.2 Wrench controller in wrench space

The desired wrench $\boldsymbol{w}_d$ is used together with the measured wrench $\boldsymbol{w}_m$ as a wrench feedback in the wrench space. In the $(6-s)$-dimensional wrench space $\mathcal{W}$ we use a $(6-s)$-dimensional coordinate vector $\boldsymbol{u}_w$ to represent the desired rate of wrench change, resulting from the wrench feedback:

$$\boldsymbol{u}_w = K_w^{FB}[\frac{1}{sec}]\ \boldsymbol{W}^{\dagger \boldsymbol{C}_c}(\boldsymbol{w}_d - \boldsymbol{w}_m). \tag{6.5}$$

The scalar K_w^{FB} with units $\frac{1}{sec}$ represents the proportional force feedback constant in $\mathcal{W}$. Matrix $\boldsymbol{W}^{\dagger \boldsymbol{C}_c}$ is the weighted pseudo-inverse of $\boldsymbol{W}$, and the weighting matrix $\boldsymbol{C}_c$ is a compliance. As explained in Section 3.2.2, the weighted pseudo-inverse minimizes the projection error when projecting the 6-dimensional wrench error into the $(6-s)$-dimensional wrench space. The norm for the wrench has the physical meaning of potential energy in an object with compliance $\boldsymbol{C}_c$.

6.2.3 Resulting manipulator twist

The s-dimensional coordinate vectors $\boldsymbol{u}_t$ and $\boldsymbol{u}_X$, together define the resulting control twist $\boldsymbol{t}_c^t$ in the twist space, and can be directly applied by a velocity controlled manipulator:

$$\boldsymbol{t}_c^t = \boldsymbol{T}\left(\boldsymbol{u}_t + \boldsymbol{u}_X\right). \tag{6.6}$$

The $(6-s)$-dimensional coordinate vector $\boldsymbol{u}_w$ defines the desired rate of wrench change at the manipulated object, and can only be applied by a velocity controlled manipulator through a compliance. The compliance $\boldsymbol{C}_c$ in the system defines the relation between a control twist $\boldsymbol{t}_c^w$ in the wrench space and the rate of wrench change:

$$\boldsymbol{t}_c^w = \boldsymbol{C}_c \boldsymbol{W} \boldsymbol{u}_w. \tag{6.7}$$

Combining the manipulator twist from the control loops in both the twist and wrench space results into the control twist $\boldsymbol{t}_c$ for the manipulator:

$$\boldsymbol{t}_c = \boldsymbol{t}_c^w + \boldsymbol{t}_c^t \tag{6.8}$$

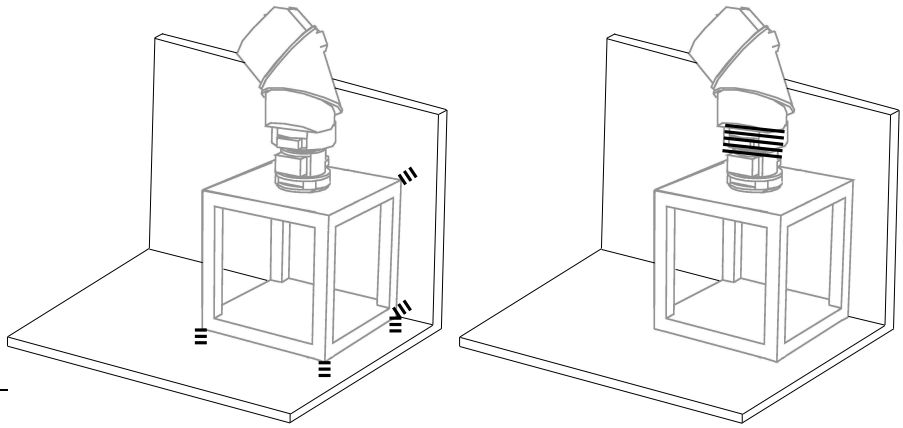
$$= \begin{bmatrix} \boldsymbol{C}_c\boldsymbol{W} & \boldsymbol{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_w \\ \boldsymbol{u}_t + \boldsymbol{u}_X \end{bmatrix}. \tag{6.9}$$

The twist $\boldsymbol{t}_c$ is converted into joint velocities $\dot{\boldsymbol{q}}$ using the manipulator Jacobian[1]. These joint velocities are controlled in an analog hardware

[1] The problem of inverting the manipulator Jacobian when it is singular or near singular falls outside the scope of this work.

proportional-integral-derivative (PID) feedback loop. We can assume that the real joint velocities correspond to the joint velocities sent to the manipulator, neglecting the dynamics of the joint velocity loops.

The hybrid controller needs a feedback constant K_w^{FB} for the wrench feedback and a feedback constant K_X^{FB} for the pose feedback. The desired twist is provided to the velocity controlled manipulator as a feedforward value. The feedback constant $K_w^{FB} > 0$ is limited by the stability of the system, and is chosen as high as possible, without making the system unstable. The feedback constant $K_X^{FB} > 0$ is used to eliminate small integration errors on the desired twist in the twist space. This feedback constant can be very low. When the weighting matrix $\boldsymbol{C}_c$ represents the real dominant compliance between the manipulator and the environment, it can be shown (De Schutter and Bruyninckx 2000) that the closed loops are stable and the errors go to zero, and that the twist and wrench controlled subspaces contain s and $(6 - s)$ completely decoupled controllers respectively. As shown in Figure 6.2, the dominant compliance can be the contact compliance, the compliance between the manipulator and the manipulated object, or a combination of both.



a *b*

FIGURE 6.2: The dominant compliance between the robot manipulator and the environment can be (a) the contact compliance, and (b) the compliance between the manipulator and the manipulated object.

6.3 Controller adaptation and selection

The contact constraints between a manipulated object and its environment change during the execution of a compliant motion task. At a CF transition, a *discrete* change in the contact constraints occurs, while during a CF compliant motion the contact constraints change *continuously*. Simultaneously with the contact constraints, the hybrid force/velocity controller of the robot manipulator is adapted. While a CF compliant motion requires a continuous change of the control algorithm, a CF transition requires a discrete change (preferably with a smoothed transition) in control algorithm. Figure 1.3 shows the control architecture for active compliant motion tasks, were each controller can be adapted online, and different controllers can be selected on demand.

6.3.1 Controller adaptation

The hybrid force/velocity controller models the local first order contact kinematics by a reciprocal twist and wrench space, as discussed in Section 3.2.2. The twist and wrench spaces are calculated based on the CF topology and the configuration within the CF. The CF *topology* defines the *type* and *dimensions* of the twist and wrench space; for example a vertex-face CF always has a 1-dimensional wrench space and a 5-dimensional twist space, while a face-face PC always has a 3-dimensional wrench space and a 3-dimensional twist space. The *configuration* within the CF defines the numerical content of the local twist and wrench bases. For a compliant motion within a given CF, the local twist and wrench spaces (and hence their bases) change simultaneously with the configuration of the contacting objects, and therefore the behavior and the control law of the hybrid controller should adapt to these changes. Because the configuration of the contacting objects is directly defined by the pose measurements of the robot manipulator, the continuous adaptation of the controllers is directly based on the pose measurements. In practice, at each timestep the local twist and wrench space is calculated based on the current pose measurement, and provided to the controller component as part of the local control strategy.

6.3.2 Controller selection

A transition between two neighboring CFs results in a discrete change of the number and/or the directions of wrench and twist controlled degrees of freedom. When adding a contact constraint the dimension of the wrench space is increased, while removing a contact constraint increases the dimension of the twist space. Therefore, a CF transition always requires a discrete switch between two hybrid controllers with a different control algorithm.

Contact formation feedback loop

In contrast to the continuous adaptation of the hybrid controller, which is directly based on the pose measurements, there is no sensor that directly provides information about the CF and CF transitions. Therefore, most active compliant motion systems are based on ad hoc or rule-based criteria to identify CF transitions and select an appropriate controller. Often a compliant motion task specification includes the set of rules to "recognize" a CF transition, for example *move compliantly along the y-axis, until the measured force along the y-axis exceeds* 15 $[N]$. This type of ad hoc specification is able to recognize simple CF transitions, but is not able to recognize more complex CF transitions, to detect unexpected CFs, or to detect the absence of a CF transition. There might also be multiple CFs that satisfy the same set of rules.

This work applies the particle filter presented in Chapter 4 to detect CF transitions and recognize the corresponding CFs in a stochastic manner. The filter combines the information from different heterogeneous sensors (wrench, twist and pose measurements) to estimate the current CF at each timestep. Multiple sensors are more informative about the CF than one single sensor. The particle filter is used in realtime, during the execution of the compliant motion, and provided to the controller as a discrete feedback, as shown in Figure 1.3 by the *selection* feedback. Based on the discrete CF feedback, an appropriate controller is selected for each CF in the executed compliant motion. The particle filter does not only provide information about *when* a CF transition occurs, but also *to which* CF a transition occurs. This allows the system to detect if the actual CF that occurs during the execution is indeed the desired CF as given in the task specification. When the actual CF differs from the desired CF, the system could move to an error state, or even plan a compliant motion from the actual CF towards the desired CF to recover from the unexpected situation.

Controller selection

A compliant motion task specification consists of a set of desired (d) contact formations $CF_1^d \dots CF_m^d$. During the execution of this compliant motion, the particle filter estimates the actual (m) contact formations $CF_1^m \dots CF_m^m$. Note that this example assumes an error free execution where the actual CF sequence corresponds to the desired CF sequence. This does not imply that the actual CF transitions occur at the same *time* as the desired CF transitions, only that the *sequence* of CF transitions corresponds. Given the information about the desired and the actual CF sequence, the system selects a control strategy. The selection strategy is based on the number of contact constraints in the actual and the desired CF. When the desired CF at a time t corresponds

to CF_k^d, five situations occur:

- The actual CF equals CF_k^m. In this situation the desired and the actual CF are equal, and the control strategy is based on either of the CFs.
- The actual CF equals CF_{k+1}^m and is more constrained than CF_k^d. In this situation a new contact constraint was added earlier than expected, and a safe control strategy based on the more constrained CF_{k+1}^m is selected.
- The actual CF equals CF_{k+1}^m and is less constrained than CF_k^d. In this situation a contact constraint was broken earlier than expected, and the selected control strategy is based on CF_{k+1}^m.
- The actual CF equals CF_{k-1}^m and is more constrained than CF_k^d. In this situation a contact constraint needs to be removed, and the selected control strategy is based on CF_k^d to remove this contact constraint.
- The actual CF equals CF_{k-1}^m and is less constrained than CF_k^d. In this situation a contact constraint needs to be added, and the selected control strategy is based on CF_k^d to add this contact constraint.

When the actual CF is not equal to CF_{k-1}^m, CF_k^m or to CF_{k+1}^m, an error is detected and the system moves to an error state.

Challenges

The same particle filter is used in online estimation of CFs during the execution of a compliant motion task and in offline estimation of CFs for programming by human demonstration. However, the challenges in both cases are different. Obviously, for online estimation the performance requirements of the filter is higher than for offline estimation. This challenge is solved by the efficient algorithms presented in Section 4.4. Another challenge originates from the different use of the estimated CFs: while in a human demonstration the CF estimation has no influence on the demonstration, the online CF estimation is used in a feedback loop to the controller. The feedback loop selects an appropriate controller for each CF and therefore the online CF estimation *influences* the task execution. The effect of this influence is easily explained with an example. Suppose the object in Figure 4.15 is approaching the horizontal face f_2. The robot manipulator will continue to move downwards, until the online estimation recognizes the new vertex-face contact v_2-f_2. The estimation however takes the twist measurement into account when evaluating different CFs, and a downward twist indicates that no contact has been established in the direction of the twist. This contradiction is taken into account by lowering the belief in the twist measurements. As a result, the online CF estimation is based on less informative measurements. Real world experimental results that verify this approach are discussed in Chapter 7.

6.4 Conclusions

This chapter presents the hybrid force/velocity controller, which separates the force and velocity feedback loops. The feedback loops operate in reciprocal twist and wrench spaces. The control strategy is adapted continuously when the contacting objects move within a CF, by calculating the local twist and wrench spaces. Discrete changes in control strategy are initiated by CF transitions. This work applies a particle filter to combine sensor information from different heterogeneous sensors for the online estimation of CFs. This general approach closes the feedback loop to the controller selection.

Chapter 7

Experiments

In theory, there is no difference between theory and practice. But, in practice, there is.

Jan L.A. van de Snepscheut

7.1 Introduction

This chapter contains the real world experimental results that verify the presented methods in this thesis, organized in two sections. The first section presents experimental results on force controlled execution of an automatically *planned* compliant motion task. First a task specification is obtained from the compliant motion planner, then the compliant task generator automatically converts this task specification into a controller level task specification, and finally the task is executed by the hybrid controller under active force control. The next section presents experimental results on force controlled execution of a *demonstrated* compliant motion task. First a task is demonstrated, and a task specification is obtained from sensor data gathered during this demonstration. Also in this experiment the compliant task generator converts the

task specification into a controller level task specification, and the hybrid controller executes the task under active force control. Additionally, a discrete event feedback loop provides realtime feedback to the controller about the actual contacts that occur during the execution.

7.2 Execution of task specification from compliant planner

This section reports on the real world experiment where the whole cycle from compliant planning to force controlled execution is automated. First a task specification is obtained from the compliant motion planner, then the compliant task generator automatically converts this task specification into a controller level task specification, and finally it is executed by the hybrid controller under active force control.

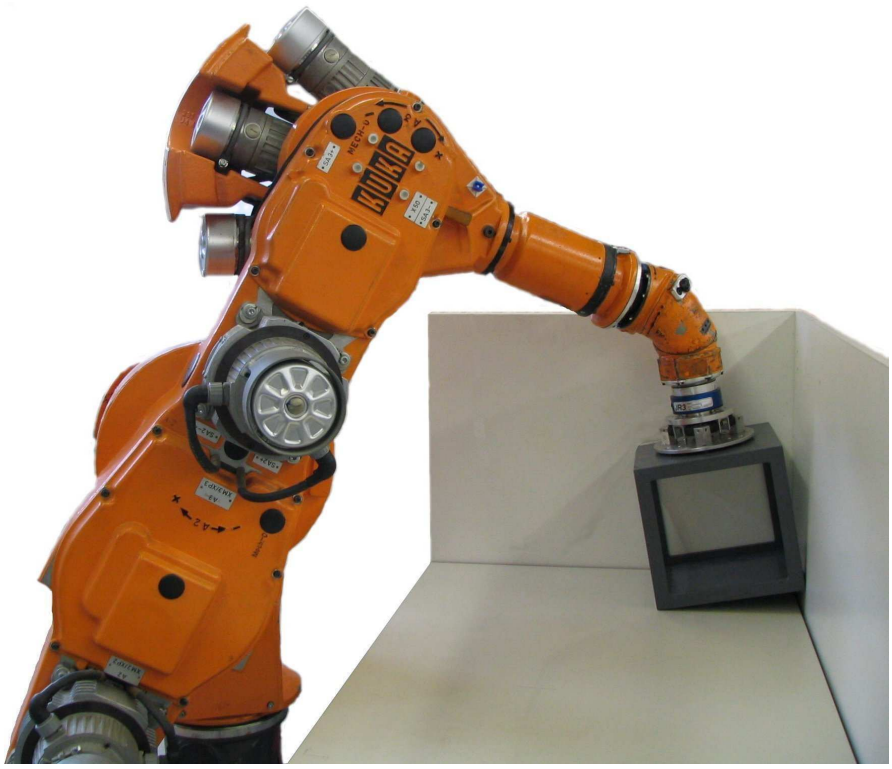


Figure 7.1: In the experimental setup the Kuka 361 six degree of freedom industrial robot manipulates a cube in contact with a corner.

7.2.1 Experimental setup

The real world experiments are executed on the *Kuka 361*, a six degrees of freedom velocity controlled industrial manipulator, shown in Figure 7.1. The original controller of the manipulator had very limited capabilities and is bypassed to a desktop computer ($P4$ 2.8 $[Ghz]$) equipped with data acquisition cards. This control computer directly controls the manipulator and reads its sensors. The software platform on the control computer is based on the hard real-time *Open RObot COntrol Software* (Orocos) framework (Bruyninckx 2001; Bruyninckx, Soetens, and Koninckx 2003) and the *Real-Time Application Interface* (RTAI) extension to the Linux kernel. The low level controller and hardware communication is implemented as a hard realtime, non-interruptible task running at 100 $[Hz]$ with a maximum latency of 16 $[\mu s]$. A six component *JR3* wrench sensor mounted on the robot wrist measures the contact wrench occurring between the manipulated object and the environment.

Figure 7.1 shows the manipulated object, a cube with an edge length of 25.0 $[cm]$, attached to the manipulator with a flexible mounting part. The inertia matrix $\boldsymbol{M}_c$ of the cube is expressed in a reference frame at the center of the cube and is approximately given by:

$$\boldsymbol{M}_c = \begin{bmatrix} 3.6\ [kg]\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 1.9\ [kgm^2]\boldsymbol{I}_{3\times3} \end{bmatrix}, \tag{7.1}$$

while the compliance matrix $\boldsymbol{C}_c$ of the flexible mounting part expressed in a reference frame at the center of the mounting part, and is approximately given by:

$$\boldsymbol{C}_c = \begin{bmatrix} 50\times10^3\ \left[\frac{N}{m}\right]\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 10^3\ \left[\frac{Nm}{rad}\right]\boldsymbol{I}_{3\times3} \end{bmatrix}. \tag{7.2}$$

The cube is moved in contact with the environment, which consists of three perpendicular faces of a corner.

7.2.2 Compliant planning

In the first step, a complete contact state graph is generated automatically, given the geometric models of the cube and the corner. Then, the user specifies the input for the compliant planner: the start and goal CF, and intermediate CFs that should be included in the compliant path. The CFs provided to the planner are not chosen to assemble the cube into the corner, but to include many different CFs to verify the effectiveness of our approach. The compliant planner uses the automatically generated contact state graph containing 245

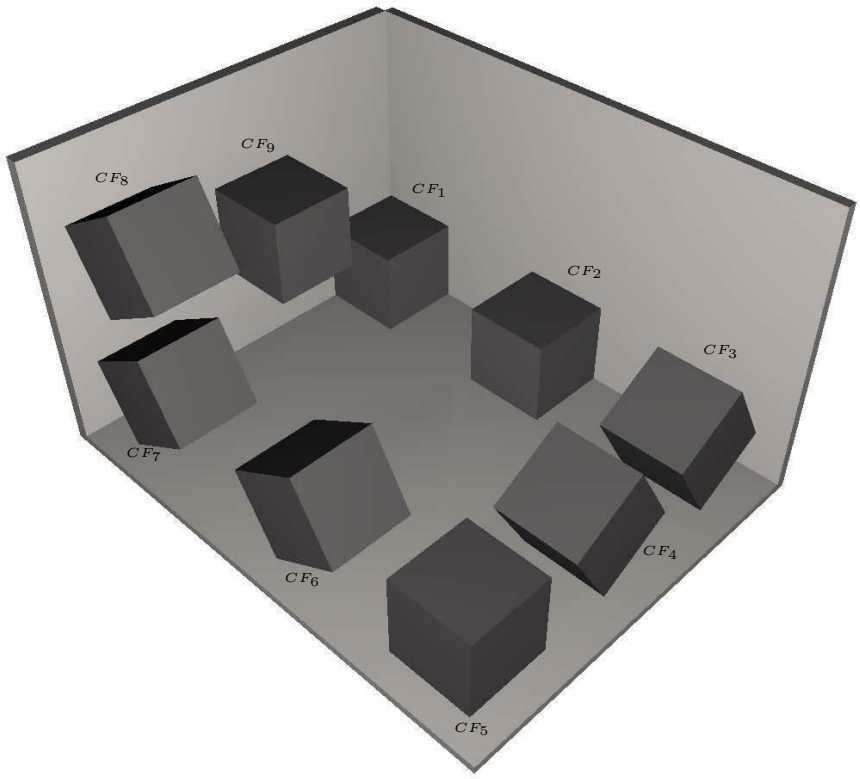


Figure 7.2: The experiment executes a sequence of contact formations including different contact relaxations and the creation of new contacts under force control. The sequence starts at a locally most constrained pose CF_1, and continues clockwise through the contact formations listed in Table 7.1. The figure only shows the contact formation sequence, not all the configurations.

CF	Principal Contacts	dim($\mathcal{W}$)	dim($\mathcal{T}$)
CF_1	$3 \times face - face$	6	0
CF_2	$2 \times face - face$	5	1
CF_3	$2 \times edge - face$	4	2
CF_4	$1 \times edge - face$	2	4
CF_5	$1 \times face - face$	3	3
CF_6	$1 \times edge - face$	2	4
CF_7	$2 \times edge - face$	4	2
CF_8	$1 \times edge - face$	2	4
CF_9	$1 \times face - face$	3	3

TABLE 7.1: The sequence of contact formations between the cube and its environment. For each contact formation the dimension of the twist and wrench space is given.

CFs and the user specified CFs, to automatically generate a compliant path that connects all CFs. The translational and rotational step size of the planner are chosen as:

$$\Delta trans = 3.0\ [cm], \tag{7.3}$$

$$\Delta rot = 5.0\frac{\pi}{180}\ [rad]. \tag{7.4}$$

The resulting sequence of CFs generated by the planner, together with the dimensions of the twist and wrench spaces are shown in Figure 7.2 and listed in Table 7.1.

While the user only specifies CF_1, CF_4, CF_7 and CF_9, the compliant planner generates all necessary intermediate CFs to connect the user specified CFs. The planned compliant path includes a sequence of CFs with relaxations of contact constraints, motions within a CFs, and creations of new contact constraints. As shown in Figure 7.2, the sequence starts at a locally most constrained CF_1 when the cube is in contact with all three faces of the corner. The sequence then continues clockwise as indicated by the numbering of the CFs.

7.2.3 Compliant task generator

The *Compliant Task Generator* automatically converts this compliant path, together with the desired magnitudes for the twist and wrench components, into a task specification for the hybrid controller. For this experiment, the desired magnitudes for the twist and wrench components, at a reference point

at the center of the cube, are chosen as:

$$\bar{f} = 25\ [N] \tag{7.5}$$
$$\bar{\tau} = 3.5\ [Nm] \tag{7.6}$$
$$\bar{v} = 0.018\ [m/s] \tag{7.7}$$
$$\bar{\omega} = 0.025\ [rad/s] \tag{7.8}$$

From this specification, an inertia matrix that defines the norm of a twist through Equation (3.25), is derived:

$$\boldsymbol{M} = 1\ [kg] \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 0.72^2[m^2]\ \boldsymbol{I}_{3\times3} \end{bmatrix}. \tag{7.9}$$

Also the compliance matrix that defines the norm of a wrench through Equation (3.27), is derived from the specification:

$$\boldsymbol{C} = 1\ [m/N] \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & 1/7.07^2\ [1/m^2]\ \boldsymbol{I}_{3\times3} \end{bmatrix}. \tag{7.10}$$

Both the inertia and the compliance matrix are expressed in a reference frame at the center of the cube. The twist and wrench magnitudes for these norms, derived from the specified magnitudes, are:

$$E_{kin} = 0.324\ 10^{-3}[J], \tag{7.11}$$
$$E'_{pot} = 625\ [J]. \tag{7.12}$$

The compliant task generator generates the task specification for the hybrid controller online, this is during the execution. At each time step, when the hybrid controller requires a new setpoint, it is automatically generated, as explained in Section 5.3.

7.2.4 Force controlled execution

The result of the planning, task generation and hybrid controller is a compliant motion of the cube in contact with the corner, through the planned sequence of CFs. In this experiment, the hybrid controller uses pose and wrench feedback constants chosen as:

$$K_X^{FB} = 0.05\ \left[\frac{1}{s}\right], \tag{7.13}$$
$$K_w^{FB} = 3.0\ \left[\frac{1}{s}\right]. \tag{7.14}$$

The weighting matrices $\boldsymbol{M}_c$ and $\boldsymbol{C}_c$ are chosen the same as in Equation (7.1) and (7.2). During the experiment, wrench and twist measurements are

recorded. Figure 7.3 shows the force and torque components of the measured contact wrench (after gravity compensation) between the manipulated object and the environment. This measured contact wrench has components in both twist and wrench space due to friction and inertia forces. Figure 7.5 shows the same measured contact wrench measurements, projected onto the wrench space. The projected contact wrench is the input to the wrench controller, which only applies to the part of the wrench in the wrench space (see Equation (6.5) where $\boldsymbol{W}^{\dagger_{\mathcal{C}_c}}$ is the projection onto the wrench space). The hybrid controller uses the projected measured wrench in the wrench feedback loop; therefore the projected measured wrench (and not the total measured wrench) should be compared to the desired wrench.

Figure 7.6 shows the desired force and torque components. By construction the desired wrench lies withing the wrench space of the CF. The desired wrench is chosen to *achieve a motion* that adds or removes a contact constraint, or maintains a CF. It is not possible nor necessary that the applied wrench equals the desired wrench at all times. The difference between the applied and the desired wrench is explained by the following effects:

- When adding a new contact constraint between CF_1 and CF_2 (for example moving from an edge-face CF_1 to a face-face CF_2) the desired wrench is changed to the desired wrench in CF_2. However, during the whole motion from CF_1 to CF_2 the applied wrench cannot yet be equal to the desired wrench because the new contact constraint is needed to apply the desired wrench.

- When removing a contact constraint between CF_1 and CF_2 (for example moving from a face-face CF_1 to an edge-face CF_2) the desired wrench is changed to the desired wrench in CF_2. However, as long as the compliance between the manipulated object and the manipulator is compressed, an extra wrench component is measured.

Figure 7.4 shows the translational and rotational components of the measured manipulator twist. The measured twist is the total motion of the robot manipulator, with components in both the twist and the wrench space. The components in the wrench space are generated by the wrench controller to achieve the desired contact wrench. Figure 7.7 shows the measured twist, projected onto the twist space. The hybrid controller uses the projected measured twist in the twist feedback loop; therefore the projected measured twist (and not the total measured twist) should be compared to the desired twist. The desired twist is shown in Figure 7.8, and by construction lies within the twist space of the CF. Vertical dotted lines divide all the twist and wrench plots into 9 sections, one per contact formation that occurs in the experiment. Each vertical line shows where a change of CF occurs. All measurements are

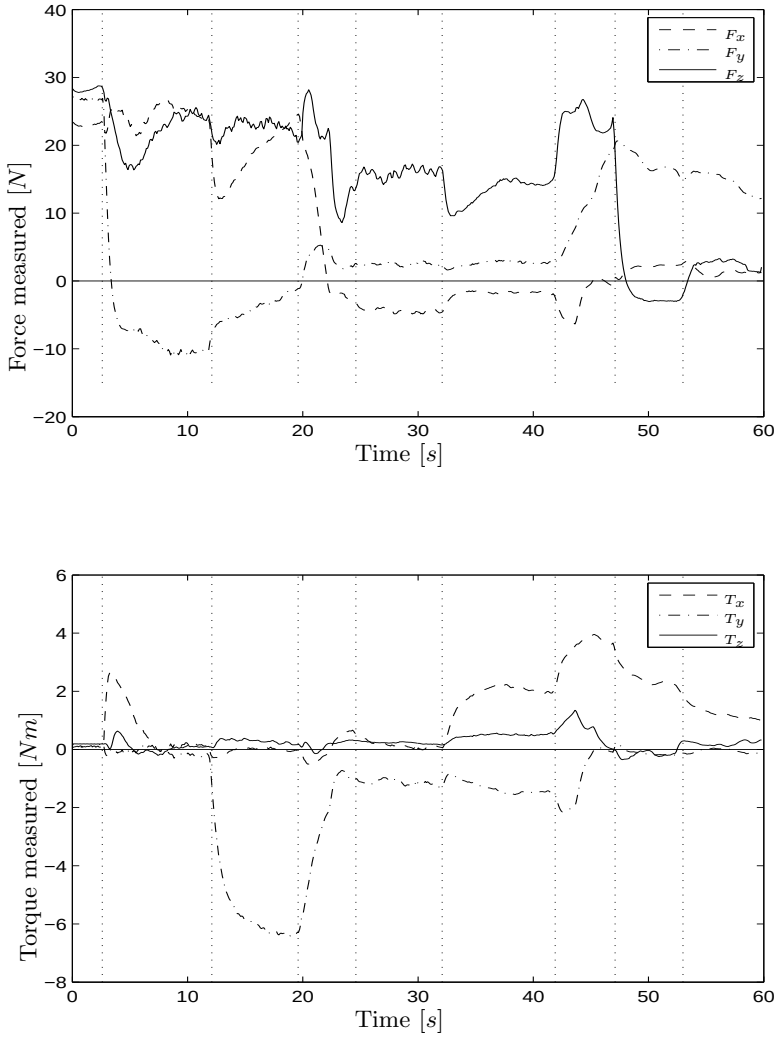


Figure 7.3: The force and torque components of the total measured wrench, reduced to a reference point at the center of the cube, expressed in the corner frame. This wrench should not be compared to the desired wrench.

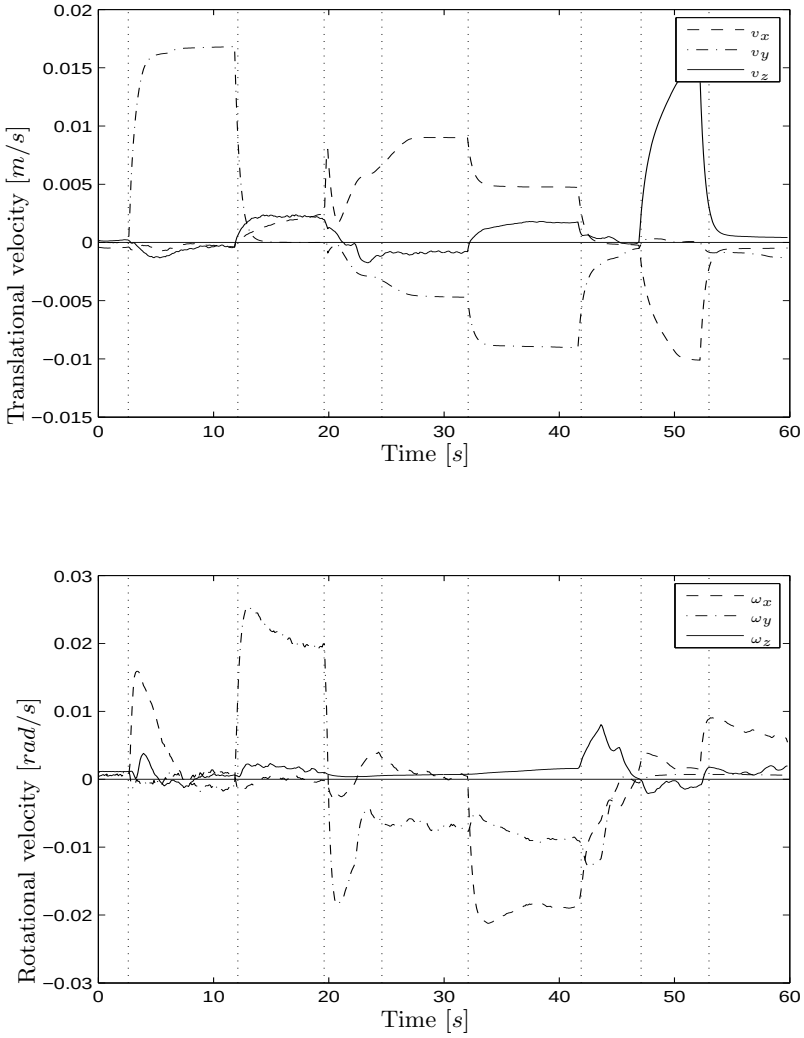
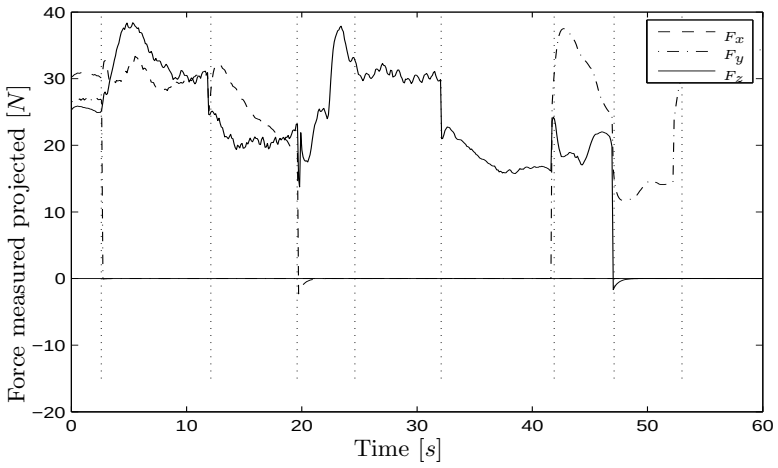


FIGURE 7.4: The translational and rotational velocity components of the measured twist of a reference point at the center of the cube, expressed in the corner frame. This twist should not be compared to the desired twist.



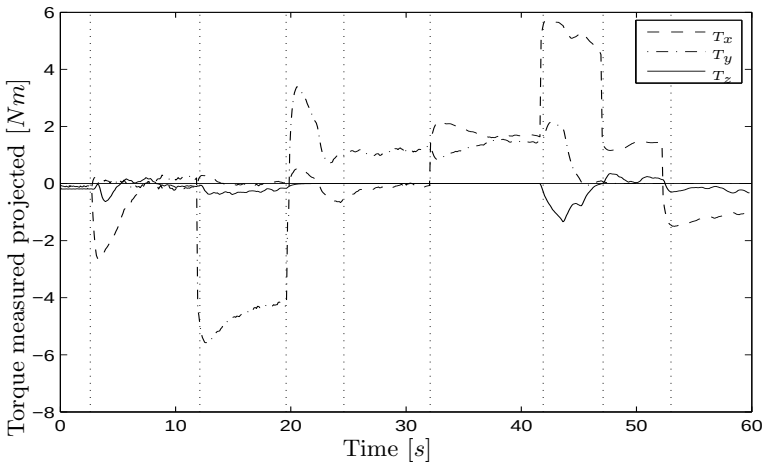


Figure 7.5: The force and torque components of the measured wrench, reduced to a reference point at the center of the cube, expressed in the corner frame, and projected onto the wrench space. This wrench can be compared to the projected desired wrench on the right hand page.

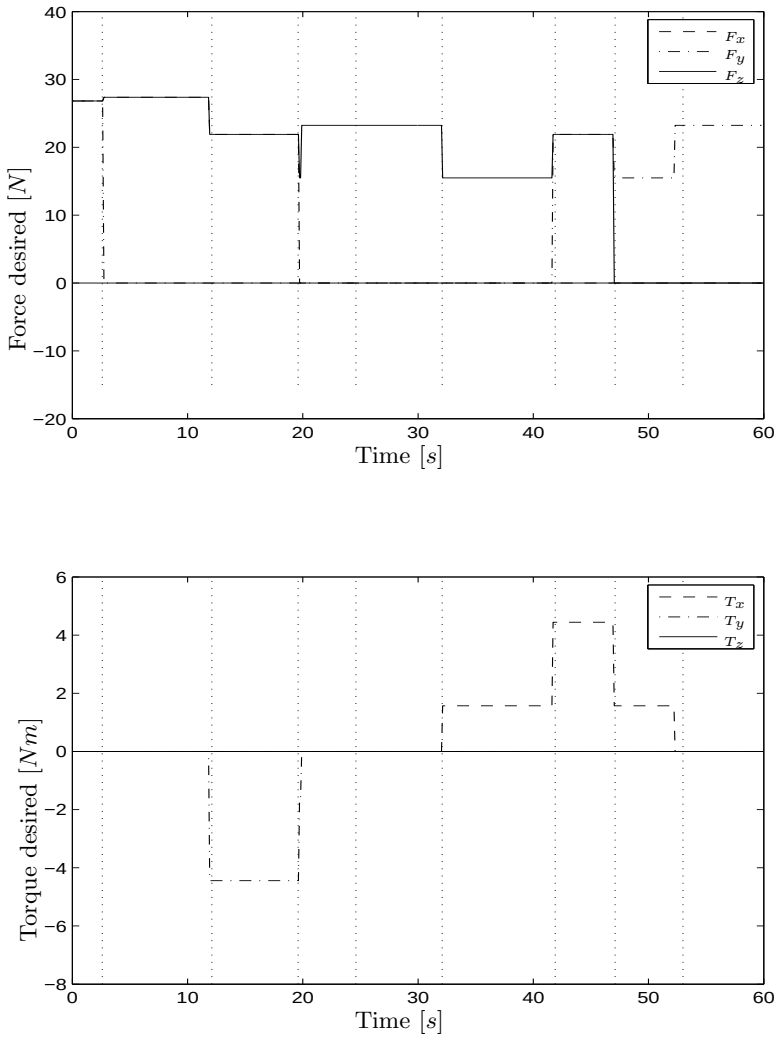


FIGURE 7.6: The force and torque components of the desired wrench, reduced to a reference point at the center of the cube, expressed in the corner frame. This wrench can be compared to the projected measured wrench on the left hand page.

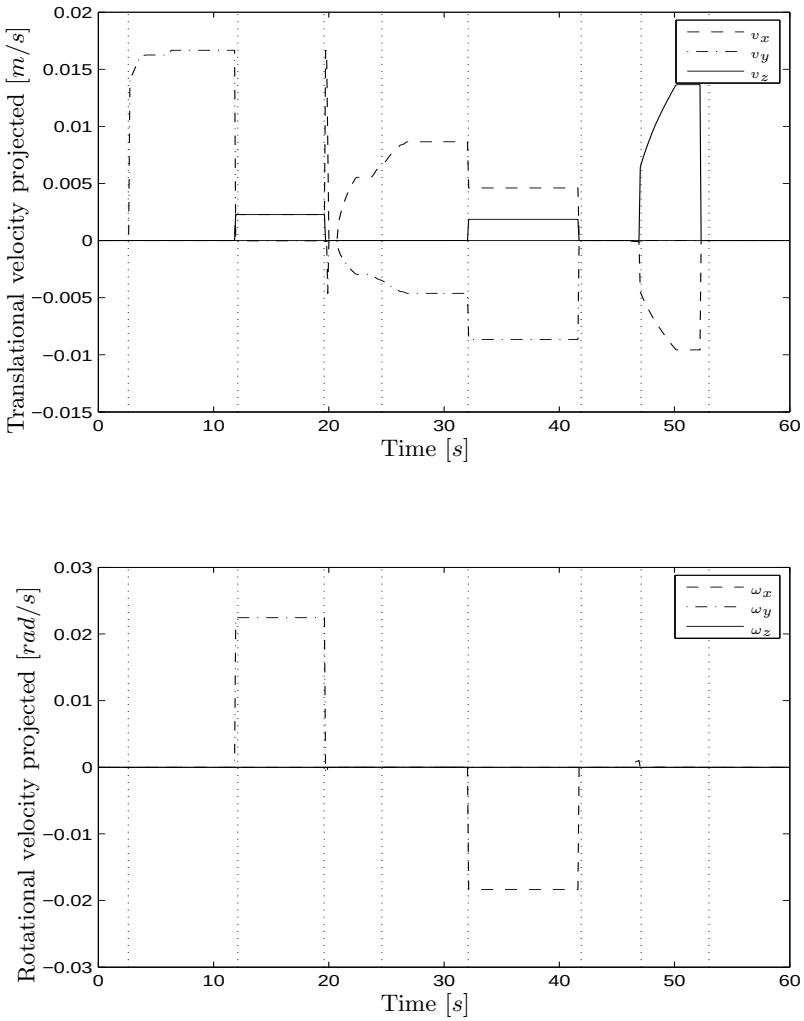


Figure 7.7: The translational and rotational velocity components of the measured twist of a reference point at the center of the cube, expressed in the corner frame and projected onto the twist space. This twist can be compared to the projected desired twist on the right hand page.

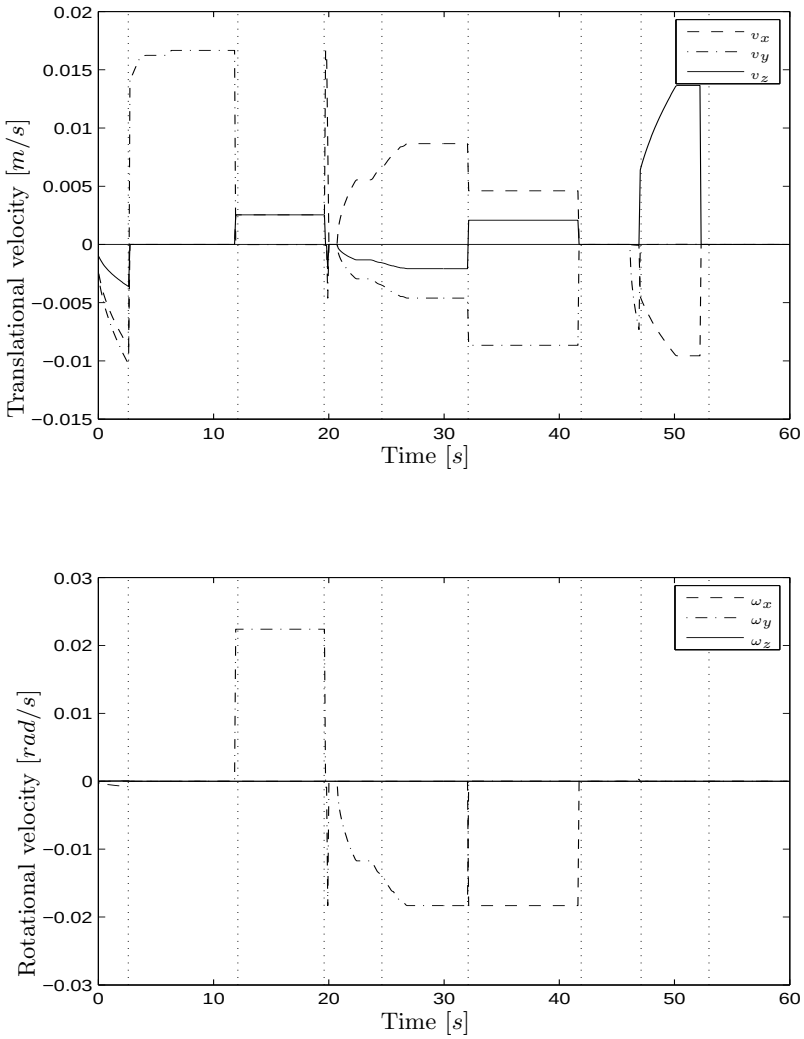


FIGURE 7.8: The translational and rotational velocity components of the desired twist of a reference point at the center of the cube, expressed in the corner frame. This twist can be compared to the projected measured twist on the left hand page.

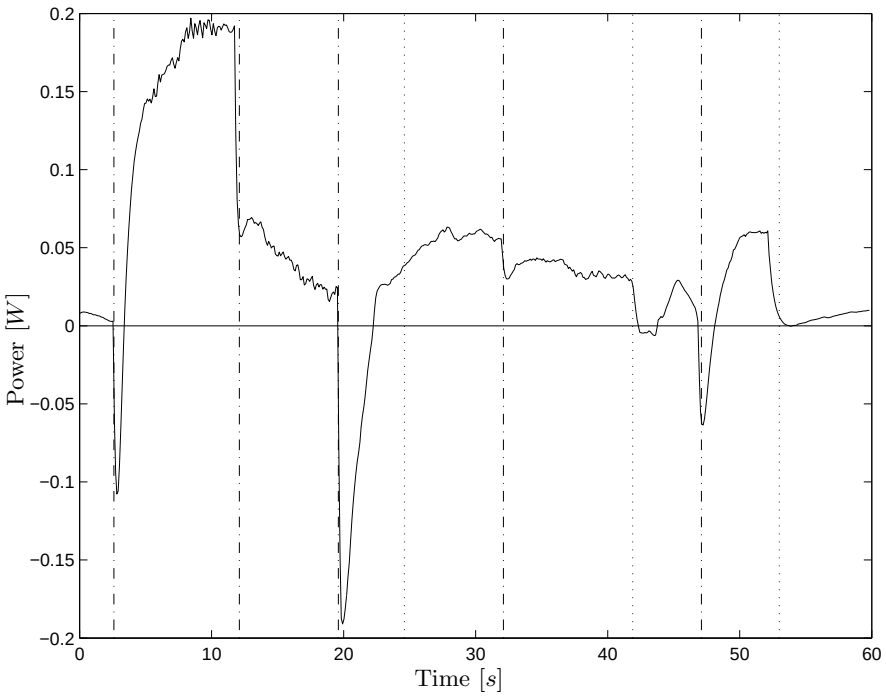


Figure 7.9: The power delivered by the robot manipulator to the manipulated object. Downward peaks show the release of energy from the flexibility during the relaxation of contact constraints.

expressed in a reference frame attached to the corner, as shown in Figure 7.10, and reduced to a reference point at the center of the manipulated cube.

The energy exchange between the manipulator and the rest of the system is shown in figure 7.9, as the power delivered by the robot manipulator. During the experiment, the manipulator delivers a positive net energy input of 2.48 $[J]$ to the system. This is the energy that is lost due to friction forces at the contact. Although the friction losses are large compared to the desired potential and kinetic energy (see Equations (7.11) and (7.12)), the energy exchanges between the manipulator and the flexibility which stores potential energy, are visible in figure 7.9. Especially during a relaxation of the contact constraints, a sudden release of energy from the flexibility occurs, resulting in a downward peak in the power graph. Vertical dash-dotted lines mark the times when the contact relaxations occur.

7.2.5 Robustness to geometric uncertainty

This experiment uses a closed loop wrench controller to compensate for uncertainties on the geometry, but no discrete feedback about the actual CF is provided to the controller for the selection of the appropriate control law[1]. Therefore the controller selection is not based on the actual CF that occurs during the execution, but it is based on the planned sequence of desired CFs. This limits the allowed uncertainty on the geometry.

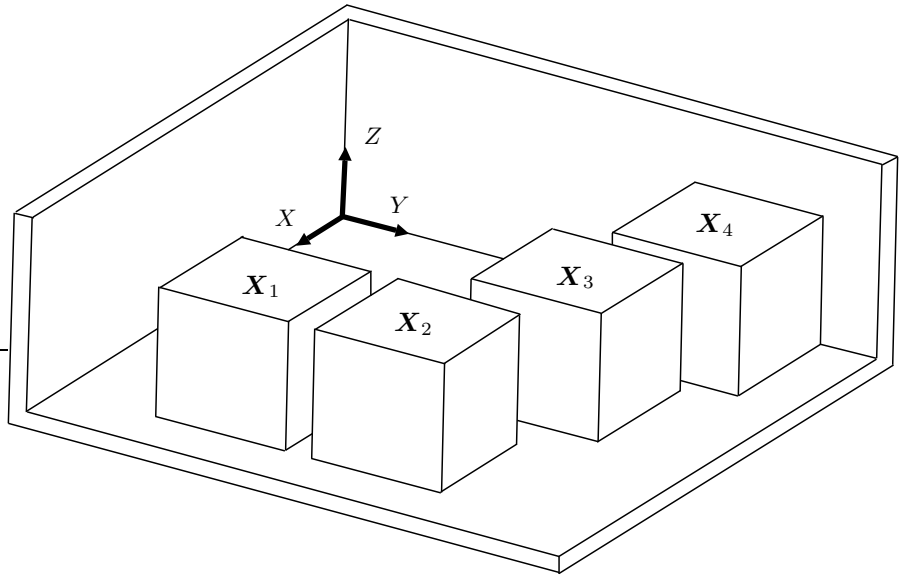


FIGURE 7.10: Creating a new contact under active force control: The motion from $\boldsymbol{X}_1$ to $\boldsymbol{X}_2$ to $\boldsymbol{X}_3$ is velocity controlled in the horizontal plane. During the motion from $\boldsymbol{X}_3$ to $\boldsymbol{X}_4$ a new contact is added under active force control instead of velocity control.

The motion between a pose $\boldsymbol{X}_i$ at a less constrained CF to a pose $\boldsymbol{X}_{i+1}$ at a more constrained CF, is executed in the local twist and wrench spaces of $\boldsymbol{X}_{i+1}$, not in the local twist and wrench spaces of $\boldsymbol{X}_i$. This means that the extra contact is added under active force control, because the wrench space of $\boldsymbol{X}_{i+1}$ is higher dimensional than the wrench space of $\boldsymbol{X}_i$. In the example in Figure 7.10 the poses $\boldsymbol{X}_1$ to $\boldsymbol{X}_3$ contain one face-face contact, while pose $\boldsymbol{X}_4$ contains two face-face contacts. The motion from $\boldsymbol{X}_1$ to $\boldsymbol{X}_2$ to $\boldsymbol{X}_3$ assumes one face-face contact, and the rotations in the plane and translation

[1] the experimental results in next section do use a discrete feedback loop.

perpendicular to the face of the next contact, are executed under velocity control. The motion from $\boldsymbol{X}_3$ to $\boldsymbol{X}_4$, where a new face-face contact is added, assumes two face-face contacts. Therefore, during the motion from $\boldsymbol{X}_3$ to $\boldsymbol{X}_4$, the translations and rotations in the plane are executed under active force control. The implication of this method is that the allowed geometric uncertainty is limited by the translational and rotational distance between $\boldsymbol{X}_3$ and $\boldsymbol{X}_4$. When the geometric uncertainty is higher, while adding the new face-face contact, it is possible that this new contact already occurs during the motion from $\boldsymbol{X}_2$ to $\boldsymbol{X}_3$, which is velocity controlled in the direction of the new contact. For the presented method to be robust, the geometric uncertainty must be smaller than the translational and rotational distance between the last pose $\boldsymbol{X}_i$ at a less constrained CF and the first pose $\boldsymbol{X}_{i+1}$ at a more constrained CF. This distance is directly defined by the step size of the compliant planner or demonstration.

The experiment in the last section of this chapter uses a closed loop wrench controller and a discrete event closed loop for the controller selection. This overcomes the limitation on the allowed geometric uncertainty as described above.

7.3 Execution of task specification from programming by human demonstration

This section reports on the real world experiments we used to validate the presented approach for programming by human demonstration and force controlled execution. From the wrench, twist and pose measurements gathered during the demonstration of a task, a geometric task description in the form of geometric parameters and contact formations, is derived using particle filter approach. The geometric task description is then automatically converted into controller setpoints by the compliant task generator. Finally the demonstrated task is executed on a real robot manipulator under active force control by the hybrid force/velocity controller. The controller selects the appropriate control law based on the online estimation of the actual CF that occurs during the execution.

7.3.1 Experimental setup

Human demonstration

In the presented experiments, a human demonstrator manipulates a cube through a complex sequence of CFs in an environment consisting of three perpendicular faces. Figure 7.11 shows the experimental setup.

FIGURE 7.11: In the experiments to validate the presented approach a human demonstrator uses a demonstration tool to manipulate a cube in contact with three perpendicular faces.

As shown in Figure 4.7, there are 12 uncertain geometric parameters in the experimental setup: the pose of the environmental object relative to the camera frame, and the pose of the manipulated object relative to the demonstration tool. The initial uncertainty on the 12-dimensional continuous geometric parameters, is represented by uniform distributions over an interval of given finite width. The uniform distribution on the pose of the environmental object relative to a world reference, and the manipulated object relative to the demonstration tool is given in Table 7.2. The discrete state can be any of the 245 possible CFs between the cube and the planes. It is known that initially there are no contacts between the manipulated objects and the environment.

Figure 7.12: In the experimental setup the Kuka 160 six degree of freedom industrial robot is used.

Parameter	width of uniform distribution
x environment	15 $[mm]$
y environment	15 $[mm]$
z environment	130 $[mm]$
R environment	0.5 $[rad]$
P environment	0.5 $[rad]$
Y environment	0.5 $[rad]$
x object	5 $[mm]$
y object	5 $[mm]$
z object	5 $[mm]$
R object	0.5 $[rad]$
P object	0.5 $[rad]$
Y object	0.5 $[rad]$

TABLE 7.2: The initial uncertainty on the 12-dimensional geometric parameter is given by a uniform distribution bounded by a given width.

Force controlled execution

The real world experiments are executed on the *Kuka 160*, a six degrees of freedom velocity controlled industrial manipulator, shown in Figure 7.12. Identical to the controller of the Kuka 361 presented in the first section of this chapter, the controller of the much larger Kuka 160 is also bypassed to a desktop computer ($P4$ 2.8 $[Ghz]$) equipped with data acquisition cards. A six component *JR3* wrench sensor measures the contact wrench occurring between the manipulated object and the environment.

The manipulated object, a cube with an edge length of 25.0 $[cm]$, is attached to the manipulator with a flexible mounting part. The weighting matrices $\boldsymbol{M}_c$ and $\boldsymbol{C}_c$ are chosen the same as in Equation (7.1) and (7.2). The cube is moved in contact with the environment, which consists of three perpendicular faces of a corner.

7.3.2 Human demonstration

During the demonstration, the cube is manipulated through the sequence of CFs shown in Figure 7.13. The sequence of CFs is not chosen to assemble the cube into the corner, but to include many different CFs to verify the effectiveness of our approach.

During the human demonstration of the targeted compliant motion task, sensors mounted on the demonstration tool measure its pose, twist and contact

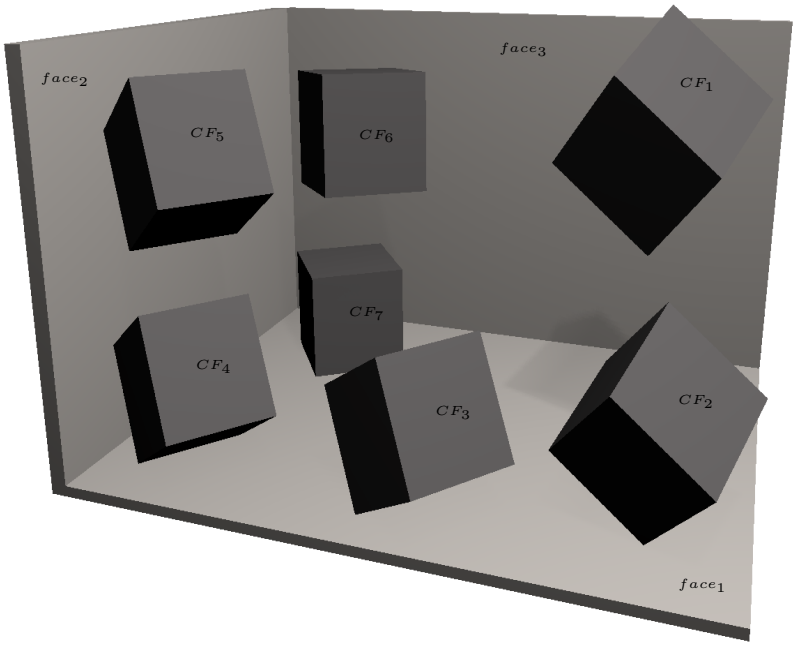


Figure 7.13: The contact formation evolution of a human demonstration where a cube is manipulated in contact with three perpendicular faces.

wrench. The Gaussian PDF:

$$\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{7.15}$$

on the wrench residue has a 2 σ boundary of 10.0 $[N]$ for the forces and 1.0 $[Nm]$ for the torques, while the Gaussian PDF on the twist residue has a 2 σ boundary of 0.001 $[m/s]$ for the translational velocity and 0.01 $[rad/s]$ for the rotational velocities. The Gaussian PDF on the distance at an EC of the current CF has a 2 σ boundary of 0.005 $[m]$. The PDF on the distance at an EC that is not part of the current CF is the only one with a non-zero expected

value of $0.1[m]$. The wrench and twist weighting matrices are chosen:

$$\boldsymbol{K}_w = \boldsymbol{K}_t^{-1} = \begin{bmatrix} 1 & & & & & \\ & 1 & & & \mathbf{0} & \\ & & 1 & & & \\ & & & 10 & & \\ & \mathbf{0} & & & 10 & \\ & & & & & 10 \end{bmatrix}. \tag{7.16}$$

Based on multiple experiments it was shown that $20,000$ particles[2] are sufficient to obtain a robust recognition of the CFs. Although for a single experiment it is sometimes possible to lower the number of particles to as few as 5000 and still obtain the same CF segmentation, at least $20,000$ particles are needed to ensure that the CF recognition for all similar experiments with a comparable uncertainty is successful. The joint posterior PDF is dynamically re-sampled using importance sampling, once the effective number of particles drops below $12,000$.

Approach to first contact

In the first part of the experiment, the cube has no contact with the environment, and approaches one of the planes of the environment. Figure 7.14 shows the time evolution of the uncertainty on the z position of the environment. The uncertainty on this position, represented by a histogram, is 1 component of the 12-dimensional continuous parameter; it is obtained by integrating[3] over the 11 other components, for a given CF. The first five sub-figures show the position parameter for the no-contact CF, while the sixth sub-figure shows the position parameter for the first vertex-face CF. Initially the position is represented by a uniform distribution, indicating that there is little knowledge about its value. When the cube approaches the plane, the probability decreases on the left side of the distribution. This shows that the cube "penetrated" one of the possible positions of the plane without detecting a contact, thus proving that possible position invalid. This evolution continues until the cube makes a vertex-face CF with the plane. The CF transition is detected due to the inconsistency between the measured wrench and the assumed no-contact CF. The knowledge about the vertex-face CF allows accurate estimation of the position of the plane, also decreasing the probability on the right side of the uniform distribution. When the probability density

[2] the implementation of the presented algorithms is capable of processing $90,000$ particles per second, on a 2 $[GHz]$ AMD 64 laptop, which is sufficient for realtime discrimination between 245 CFs and estimation of uncertain geometrical parameters.

[3] The hybrid PDF is represented by discrete samples; the integral over the parameters of the PDF is approximated by the summation over the discrete samples.

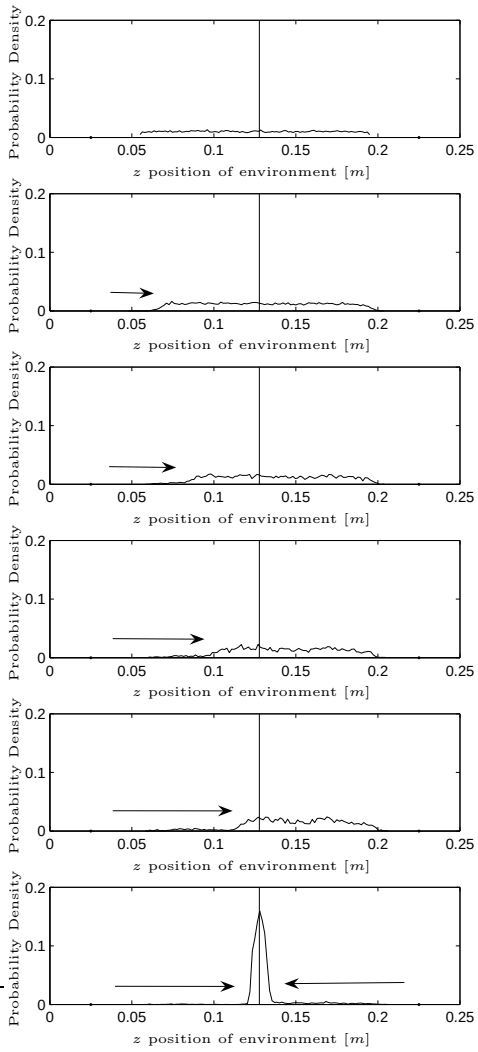


Figure 7.14: The time evolution of the probability density on the position of a plane, when approaching the plane with a cube. The vertical line shows the true position of the plane. The probability density decreases gradually when the cube approaches. The last figure shows how the width of the probability density suddenly decreases when the cube makes contact with the plane.

CF	Principal Contacts	Wrench	Twist
CF_1	no contact	0	6
CF_2	face1-vertex1	1	5
CF_3	face1-edge1	2	4
CF_4	face1-edge1 face2-vertex2	3	3
CF_5	face2-vertex2	1	5
CF_6	no contact	0	6
CF_7	face1-face3	3	3
CF_8	face2-edge3	3	3
CF_9	face1-edge4	3	3

TABLE 7.3: The sequence of contact formations between the cube and its environment during a human demonstration.

decreases on one side of the distribution, the probability density increases for the rest of the distribution, keeping the total probability unchanged.

Sequence of contact formations

In the rest of the experiment, the cube is manipulated in contact with the environment, through a complex sequence of CFs. The demonstrated sequence of CFs and the dimension of the wrench and twist spaces is shown in Table 7.3 and Figure 7.13.

The experiment includes complex CFs and CF transitions, such as:

- adding contact constraints (CF_1 to CF_2 to CF_3),
- removing contact constraints (CF_5 to CF_6),
- adding many contact constraints at once (CF_6 to CF_7),
- removing many contact constraints at once (CF_4 to CF_5), and
- simultaneous contacts with the two planes (CF_4).

Figure 7.15 shows the evolution of the estimated probability on each of the 245 possible CFs. Each CF is one possible value of the discrete state, and its probability is obtained by integrating over all 12 components of the continuous parameter, for each of the possible values of the discrete state. At each time step only a few CFs have a probability greater than zero. The particle filter successfully assigns the highest probability to the CF that corresponds to the true CF in the experiment. Notice that CF_8 and CF_9 have a relevant probability in the respective measurement intervals $[120-150]$ and $[184-186]$. Both these CFs are neighboring to the true CF in the measurement interval.

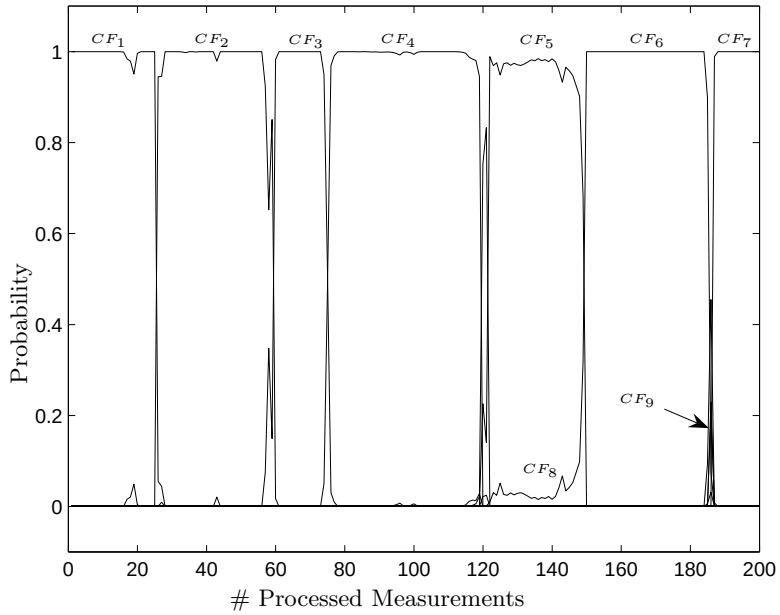


Figure 7.15: The contact formation evolution of a human demonstration where a cube is manipulated in contact with two perpendicular faces. The evolution is shown by the probability on each of the contact formations. The sequence is listed in Table 7.3.

7.3.3 Compliant task generator

The *Compliant Task Generator* automatically converts this demonstrated compliant path into a task specification for the hybrid controller. While from the compliant path planner we only obtain a geometric path description, from a human demonstration we also obtain wrench and twist information. Therefore it is not necessary for the compliant task generator to generate a desired wrench and twist, but only to project the measured demonstration wrench and twist into the wrench and twist space of the estimated CF. The compliant task generator generates the instantaneous setpoints for the hybrid controller online, this is during the execution. At each time step, when the hybrid controller requires a new setpoint, it is automatically generated.

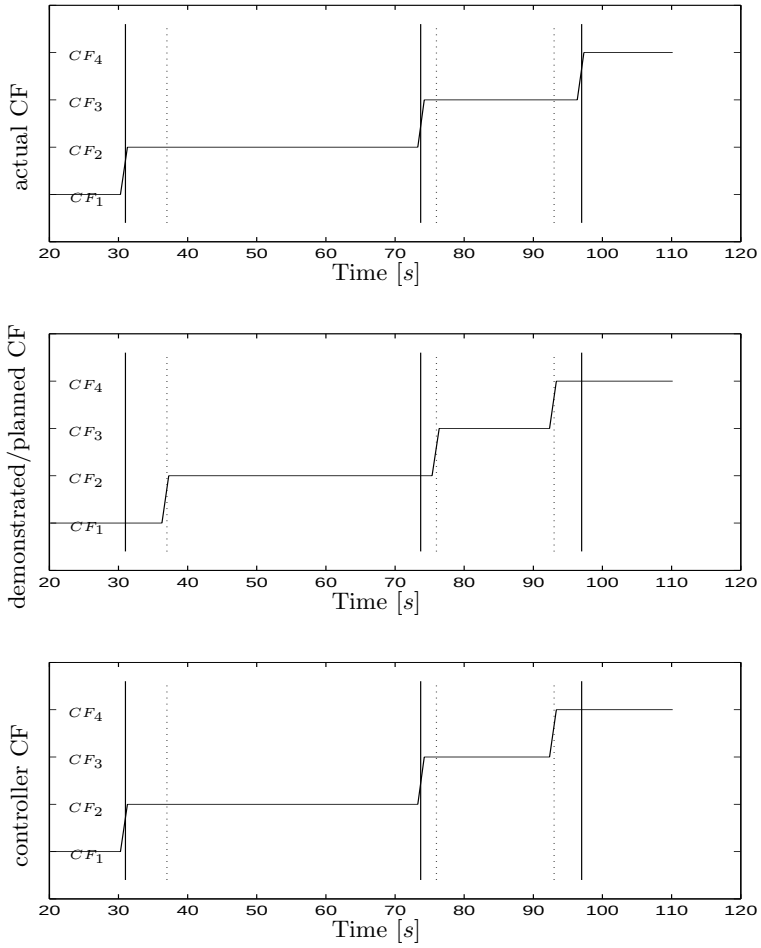


FIGURE 7.16: The actual contact formation, the demonstrated or planned contact formation, and the contact formation applied in the hybrid controller. The actual transitions from CF_1 to CF_2 to CF_3 occur earlier than the planned transitions, while the actual transition from CF_3 to CF_4 occurs later than the planned transition.

7.3.4 Force controlled execution

The hybrid force/velocity controller uses the setpoints from the compliant task generator to execute the demonstrated compliant path under active force control. In contrast to the previous experiment, where the discrete feedback loop to the controller was not closed, this experiment uses a particle filter to estimate the *actual* CFs that occurs, and provides this information in a feedback loop to the controller. This feedback loop allows the selection of an appropriate control law based on real world information, making the force controlled execution more robust, and allows the detection of unexpected events.

For the execution of the compliant path, the actual corner was positioned 20 $[mm]$ higher than during the demonstration phase, without updating the model of the hybrid controller. *Without* a discrete feedback loop, this modelling "error" would results in a collision when the cube approaches the first vertex-face contact with the environment: the first contact is approached under velocity control, and only 1 $[mm]$ before the contact (this is the distance between the last free space setpoint and the first contact setpoint) the controller would have switched to force control. The result would be a velocity controlled collision motion during 19 $[mm]$, physically damaging the experimental setup.

In this experiment however, the actual CFs that occur during the execution are monitored and provided to the controller. Figure 7.16 shows the time evolution of all discrete information provided to the controller to select an appropriate control law. The figure on top shows the *actual* CF that was recognized during the execution, the center figure shows the *desired* CF that was recognized during the demonstration phase, and the bottom figure shows the *controller* CF that was chosen by the controller to specify its control law. The full vertical lines show when the actual CF transitions occurred, while the dotted vertical lines indicate when the CF transitions were expected.

Figure 7.17 shows the measured contact wrench during the execution, reduced to a reference point at the center of the cube, expressed in the corner frame. Figure 7.19 shows the same measured contact wrench, projected onto the wrench space. The projection is based on the wrench space selected by the hybrid controller, and is therefore the measured wrench that is "seen" by the controller. This is clearly illustrated at the time of the first CF transition, at 31 $[sec]$. Before the transition is detected, the measured wrench starts building up, while the projected measured wrench remains zero. Only after the CF transition is detected and the control law is changed, the projected measured wrench is non-zero.

Figure 7.18 shows the contact wrench that was measured during the demonstration, reduced to a reference point at the center of the cube, expressed in the corner frame. This is the desired contact wrench for the ex-

ecution of the compliant motion task. Figure 7.20 shows the same desired contact wrench, projected onto the wrench space. The projection is based on the wrench space selected by the hybrid controller. The desired wrench trajectory is replayed independently of the actual CF or the CF used by the controller. This is illustrated right after the first contact constraint is added, at 31 [*sec*]: according to the task specification this contact constraint was only expected 4 [*sec*] later, and therefore the desired wrench is zero during the first 4 [*sec*] after the first contact. The projection of the desired wrench, which is the input for the hybrid controller, will of course change when the wrench space changes at a CF transition.

Figure 7.21 shows the total twist of the robot manipulator during the execution, expressed in the corner frame. Figure 7.23 shows the same twist, projected onto the twist space. The projection is based on the twist space selected by the hybrid controller, and is therefore the measured twist that is "seen" by the controller.

Figure 7.22 shows the total twist of the demonstration tool, measured during the demonstration of the task and expressed in the corner frame. This is the desired twist for the hybrid controller. Figure 7.24 shows the same desired twist, projected onto the twist space. The projection is based on the twist space selected by the hybrid controller. Similar to the desired wrench, the desired twist is also not adapted to the actual CF of the controller CF. The projected desired twist however, changes together with the controller CF.

7.4 Conclusions

This chapter presents the real world experimental results that show the effectiveness of the presented methods in this thesis. In the first experiment a compliant path generated by a compliant path planner is automatically converted into controller setpoints and executed under active force control by a hybrid force/velocity controller. The allowed geometric uncertainty for this experiment is limited due to the lack of discrete feedback loop to the controller. In the second experiment a task is demonstrated by a human demonstration. A particle filter successfully recognizes the sequence of CFs that is demonstrated, and simultaneously estimates geometrical parameters. The task specification deducted from the demonstration is converted into controller setpoints by the compliant task generator. The force controlled execution by the hybrid force/velocity controller is in this experiment equipped with a discrete feedback loop, feeding the actual CF that occurs during the execution of the task to the controller. The controller uses this information to select an optimal control strategy. The experimental results show the increased robustness of the execution with a discrete feedback loop.

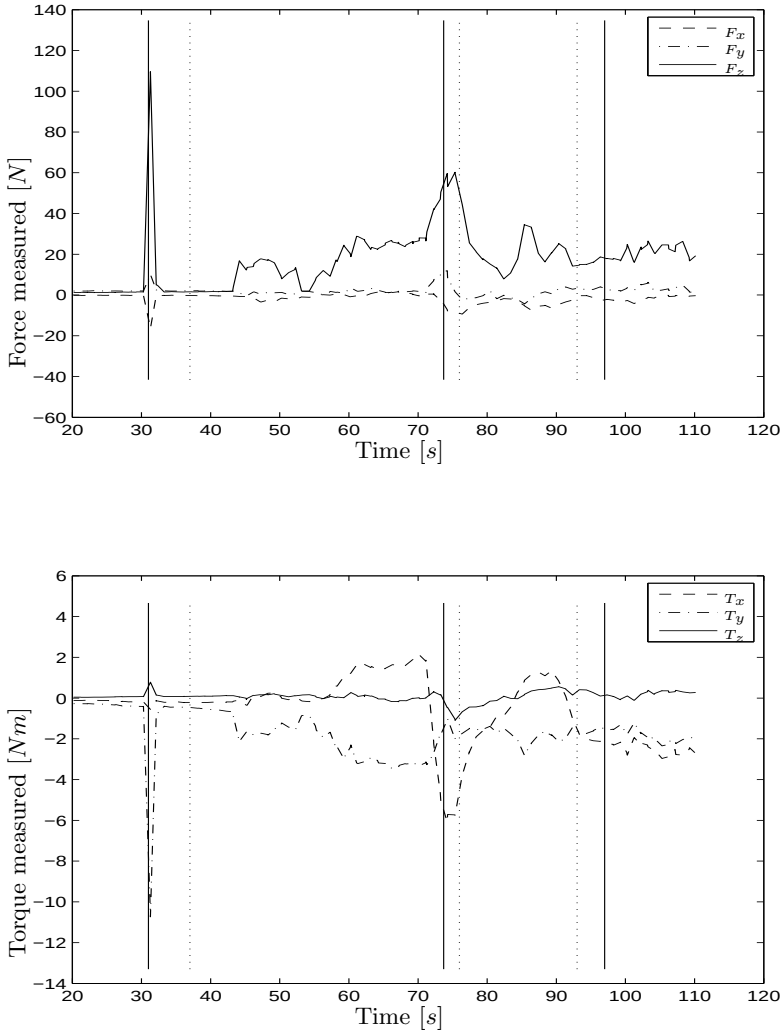


Figure 7.17: The force and torque components of the measured wrench, reduced to a reference point at the center of the cube, expressed in the corner frame. This measured wrench should not be compared to the desired wrench.

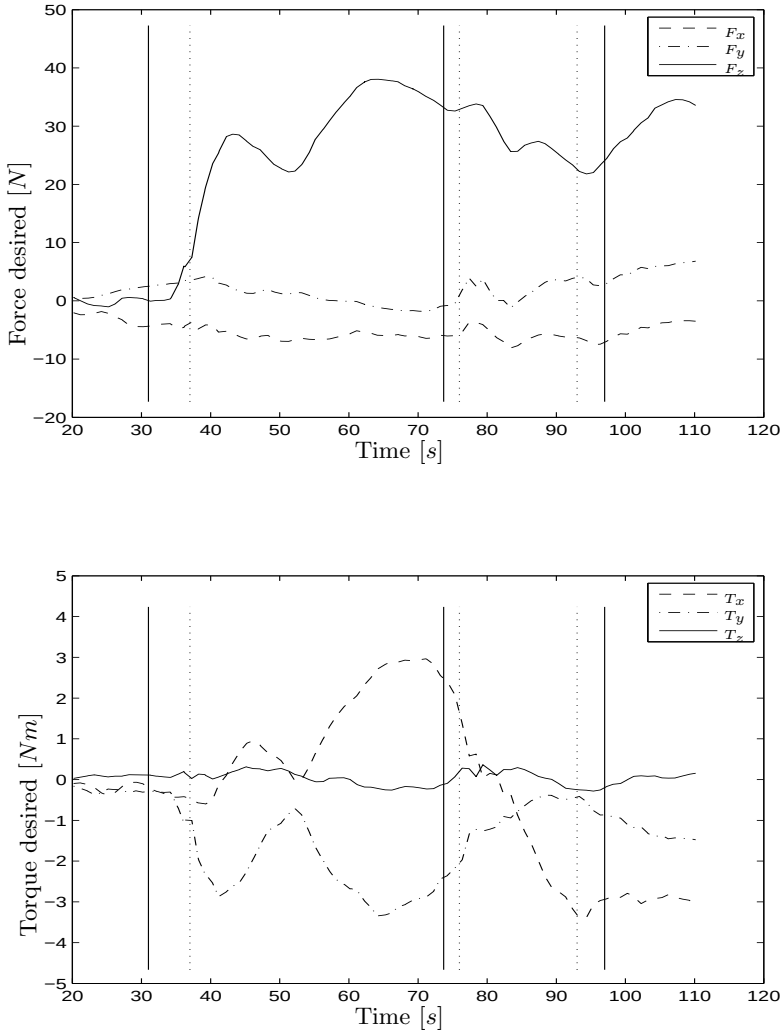


FIGURE 7.18: The force and torque components of the desired wrench, reduced to a reference point at the center of the cube, expressed in the corner frame. This desired wrench should not be compared to the measured wrench.

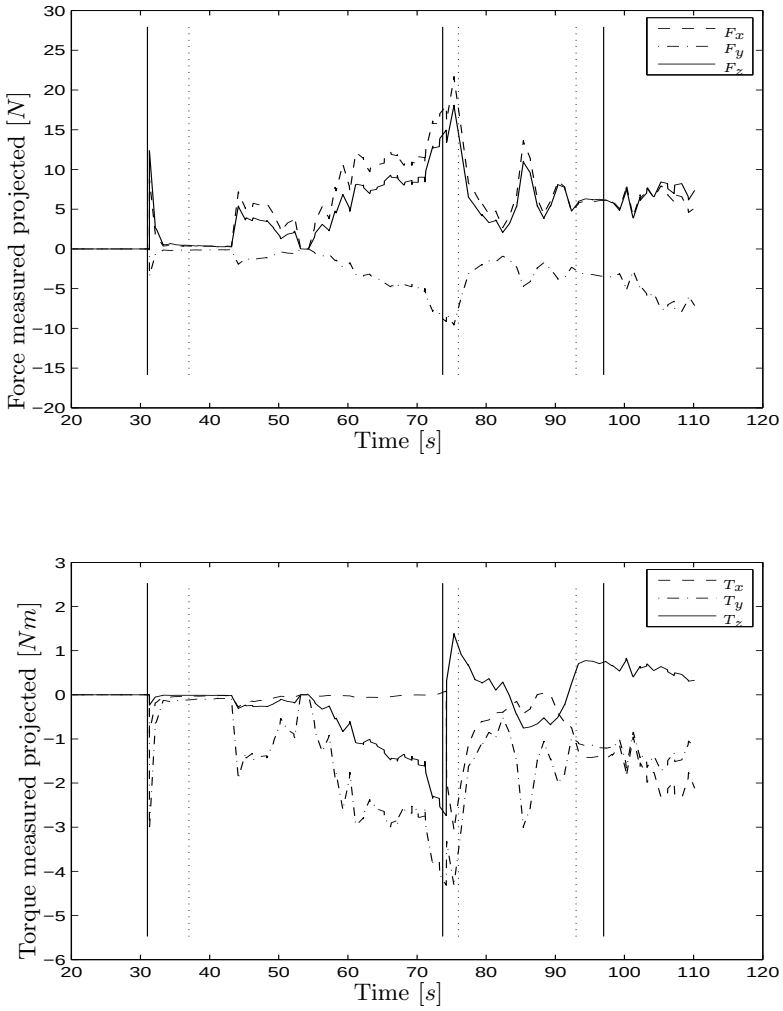


Figure 7.19: The force and torque components of the measured wrench, reduced to a reference point at the center of the cube, expressed in the corner frame, and projected onto the wrench space. This projected measured wrench can be compared to the projected desired wrench on the right hand page.

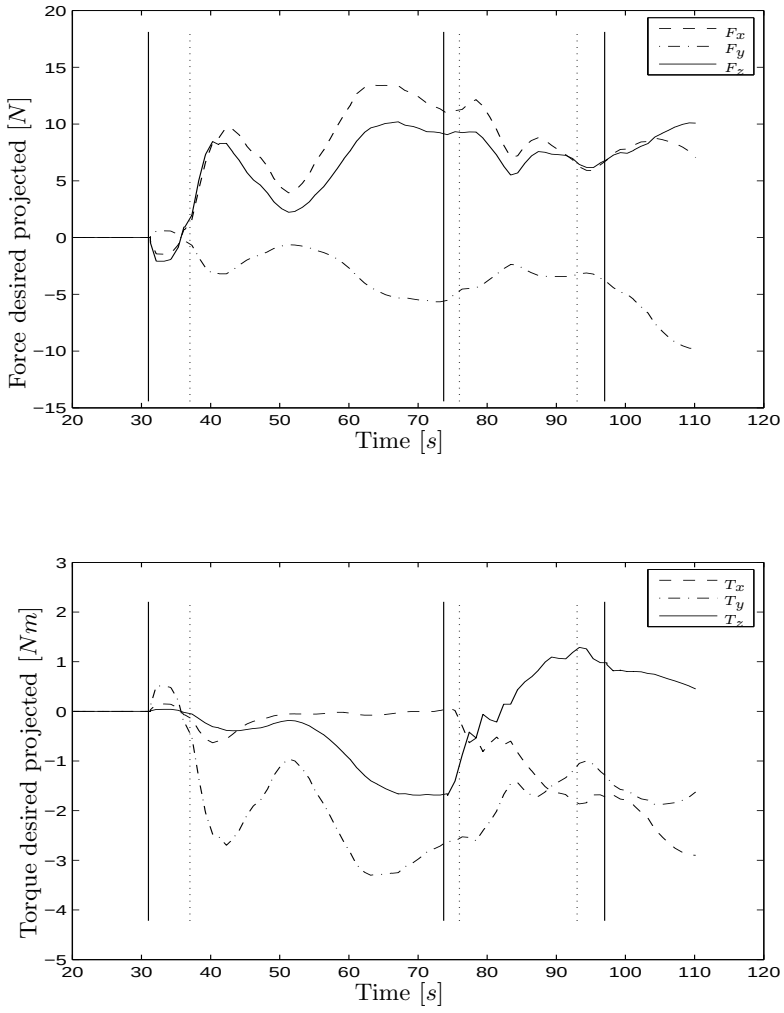


FIGURE 7.20: The force and torque components of the desired wrench, reduced to a reference point at the center of the cube, expressed in the corner frame and projected onto the wrench space. This projected desired wrench can be compared to the projected measured wrench on the left hand page.

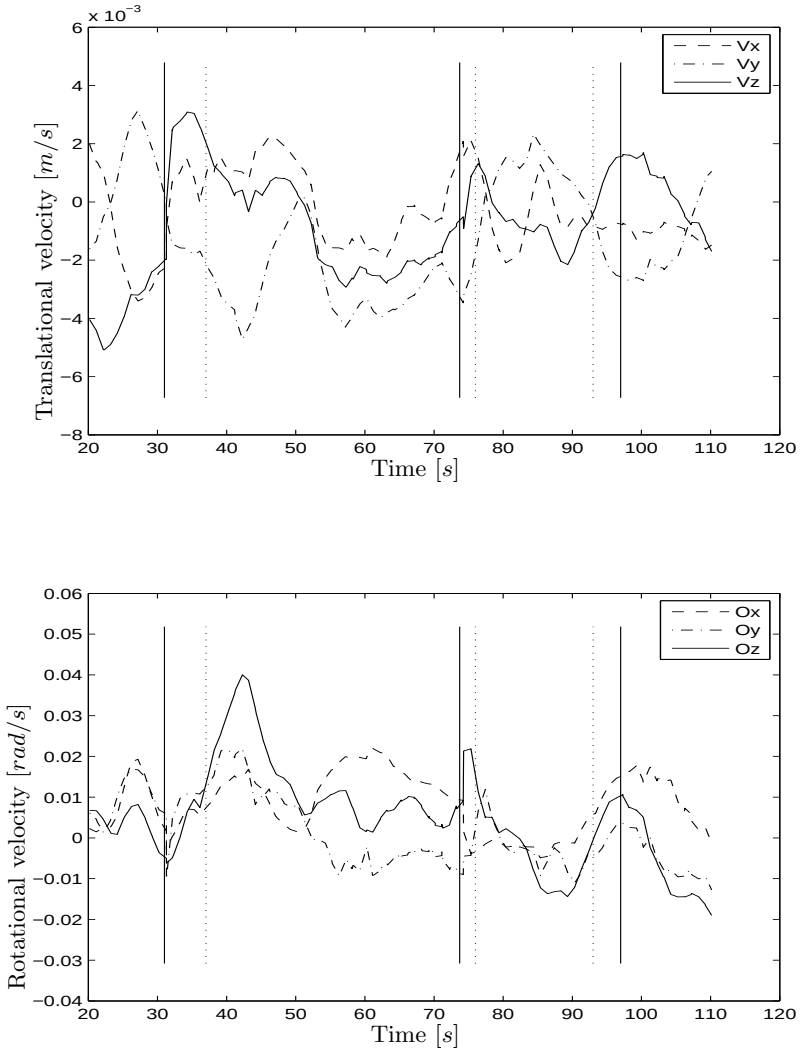


Figure 7.21: The translational and rotational velocity components of the measured twist of a reference point at the center of the cube, expressed in the corner frame. This measured twist should not be compared to the desired twist.

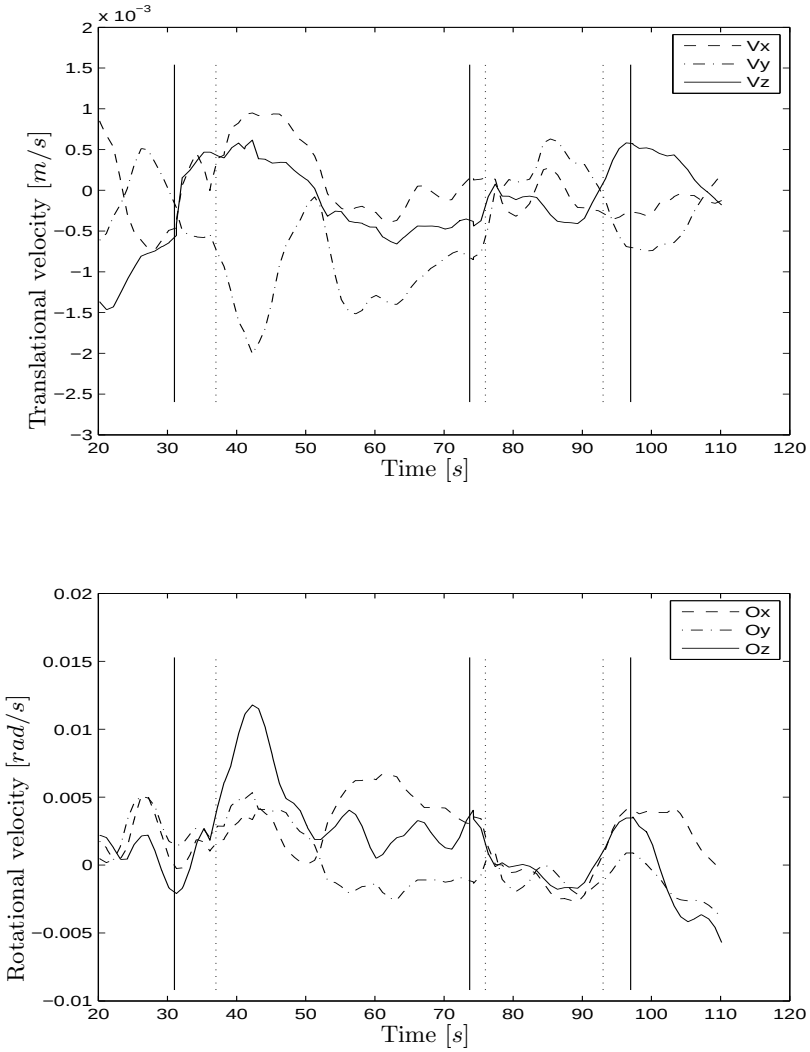


FIGURE 7.22: The translational and rotational velocity components of the desired twist of a reference point at the center of the cube, expressed in the corner frame. This desired twist should not be compared to the measured twist.

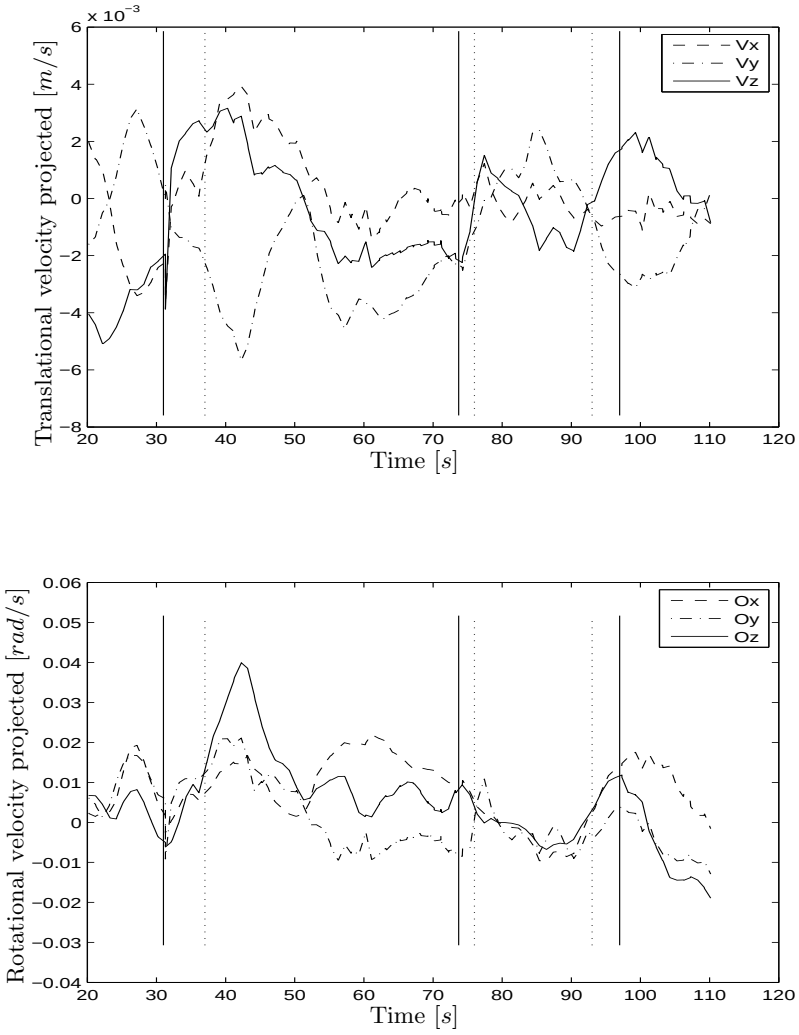


Figure 7.23: The translational and rotational velocity components of the measured twist of a reference point at the center of the cube, expressed in the corner frame and projected onto the twist space. This projected measured twist can be compared to the projected desired twist on the right hand page.

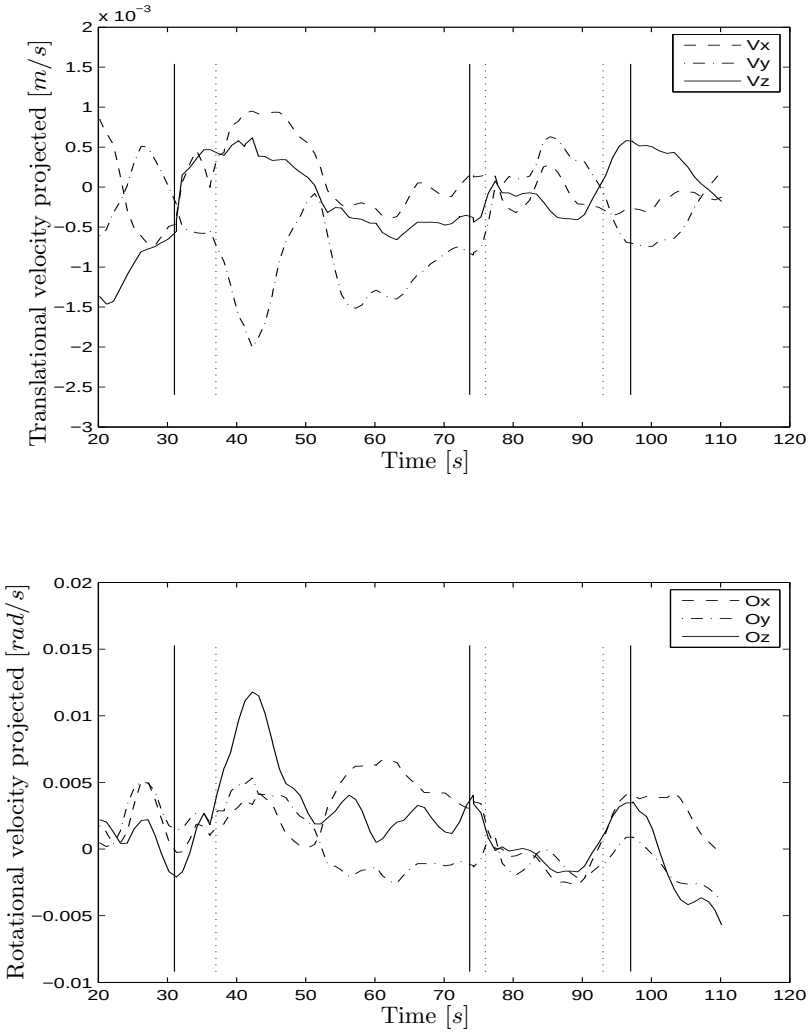


FIGURE 7.24: The translational and rotational velocity components of the desired twist of a reference point at the center of the cube, expressed in the corner frame and projected onto the twist space. This projected desired twist can be compared to the projected measured twist on the left hand page.

Chapter 8

General conclusions

Robots... I think that is a hot topic.

Bill Budge

This chapter presents the general conclusions of this thesis. The first section situates the work in the broader field of compliant motion tasks, the next section presents the contributions of this thesis, and finally the last section discusses the limitations of this thesis and gives suggestions for future work.

8.1 Situation of the work

This thesis aims towards more intelligent, autonomous and flexible robots, that are equipped with multiple sensors to operate in unstructured environments. The sensors allow the robot to observe the unknown environment and to interact with the environment. The work focusses on compliant motion tasks, where an object held by a robot manipulator is manipulated while it maintains contact with the environmental object. For compliant motion tasks, a robot is typically equipped with a force and a position sensor. The force sensor observes the force interaction between the manipulated and the

environmental object, while the position sensor observes the relative position of the two objects. Each sensor observes part of the compliant motion task parameters, and increases the ability of the robot to interact with its environment. This thesis attempts to achieve three main goals for compliant motion tasks:

- make the specification of compliant motion tasks more accessible to non-technical users by integrating *high level task specification* approaches into the compliant motion system,
- *scale* the estimation approach to make it capable of coping with a *high-dimensional* state containing many uncertain geometrical parameters and hundreds of unknown contact formations (CFs), and improving the information extraction from multiple heterogeneous sensors by applying state of the art Bayesian sequential Monte Carlo methods integrated with the topological information contained in a contact state graph, and
- increase the robustness of a compliant motion execution by the *online monitoring* of discrete CFs and selecting an adapted controller strategy accordingly.

To achieve these goals, contributions are made to the state of the art in the task specification component, the generator component and the estimation component, as shown in Figure 8.1. The resulting approach is validated in an experimental setup where a cube is manipulated in contact with a corner, through a complex sequence of CFs. The experimental results can be directly extrapolated to applications such as putting a book on a shelf in between other books, putting a drawer in its place, placing a cell phone battery inside a cell phone, or piling up boxes against a wall. All these applications are examples of high-dimensional estimation problems with uncertainty on many geometrical parameters and many hundreds of possible contact states linked by the discrete topological information in a contact state graph. The presented experiments in this thesis show that it is possible to solve such a complex and high-dimensional estimation problem in realtime. When making abstraction of the type of sensors used, many other fields of research share the same problem of solving a high dimensional estimation problem where topological information about the task is available. Examples of applications in other fields of research are the motion capturing of the human body, where a camera tracks multiple markers on the human body, or the model building using visual features to estimate the features of geometric model. Therefore the achievement of this thesis, which is the approach to solve a complex high-dimensional estimation problem in realtime, based on topological information about the task, is also relevant for other fields of research.

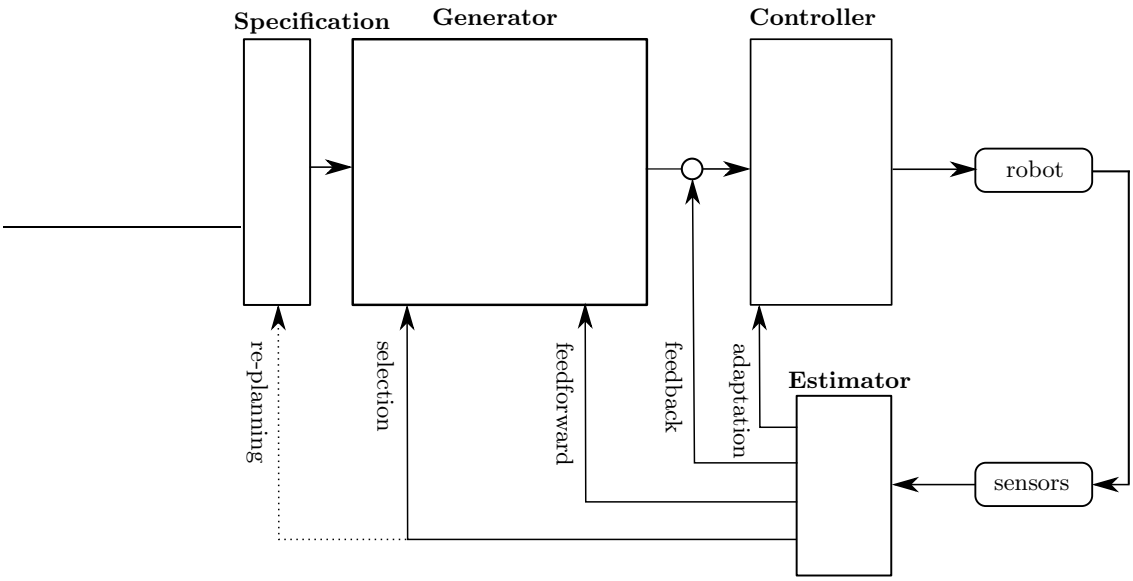


FIGURE 8.1: Overview of the control architecture for sensor based active compliant motion tasks. Figure 1.3 shows the control architecture in more detail. The dotted line shows a possible future extension of the control architecture.

8.2 Contributions

This section enumerates the contributions of this thesis to the state of the art in the field of active compliant motion. As shown in Chapter 7, all presented results are verified in real world experiments, on two different industrial robots. The contributions to the state of the art in active compliant motion are categorized in four major parts:

Manipulator constraints in the contact state graph A contact state graph represents all possible CFs between two rigid polyhedral objects as nodes, and the adjacency relationship between the CFs as arcs. This thesis applies the goal-contact-relaxation algorithm to automatically generate a contact state graph given the geometric model of two objects and a locally most constrained CF. The resulting contact state graph assumes that both objects can move freely in space. However, in many robotics tasks, it is a robot manipulator that moves one of the objects and creates contacts between the manipulated object and the environment. Therefore, whether a CF of the contact state graph can be formed and whether a CF transition of the con-

tact state graph is possible, is subject to the constraints of the manipulator. It is necessary to take into account the manipulator constraints in obtaining a contact state graph so that a compliant motion plan generated based on such a graph can be actually executed by the manipulator.

Chapter 3 presents an approach to find feasible contact states between a polyhedral part held by a manipulator and a fixed polyhedral environment. Given the contact state graph of the two polyhedral objects without the manipulator constraints, the approach checks the reachability for a robot manipulator of each CF and the connection between each two neighboring CFs. First the approach searches a CF-compliant configuration for the contacting objects that is reachable for the robot manipulator. Then, the virtual compliant controller is applied to search a compliant path for the manipulator, connecting the neighboring CFs in the contact state graph. In the resulting modified contact state graph, all nodes and arcs are feasible for a robot manipulating the object in contact with the environment. When using this modified contact state graph as an input for the compliant motion path planner, the resulting path will be feasible for the robot manipulator. The key part of the approach, the automatic verification of a compliant relaxation path between two given configurations in neighboring CFs, is implemented and verified by experimental results. This implementation is applicable to add the constraints of any *serial* robot manipulating a *polyhedral* object in contact with a *polyhedral* environment.

Estimation in programming by human demonstration This thesis presents an approach to obtain a compliant motion task specification using programing by human demonstration. In a demonstration step, a human demonstrates a compliant task by directly manipulating an object in contact with its environment, using a demonstration tool. During the demonstration, the Krypton 6D optical system measures the pose and twist of the manipulated object, while a wrench sensor measures the interaction forces between the contacting objects. Subsequently, in the interpretation step, these measurements are interpreted using sequential Bayesian estimation techniques.

Chapter 4 presents an approach based on the Bayesian sequential Monte Carlo method, or particle filter, to *simultaneously* recognize discrete CF transitions and estimate continuous geometrical parameters. The particle filter is the preferred tool to cope with this hybrid (partly discrete, partly continuous) estimation problem. While previously presented results in this field only allowed certain CFs or certain CF transitions, this approach scales the search space to *all possible CFs* between the contacting objects. To cope with this increased complexity, a more accurate prediction step is used, based on the *topological information* contained in a contact state graph, and the pose of the contacting objects. This extension, in combination with new efficient

algorithms, allows for the *realtime* simultaneous recognition of CFs and estimation of geometric parameters. The approach presented in this thesis is 367 times faster than previously presented methods.

The approach applies to convex as well as to concave *polyhedral* objects with a known geometry. The pose of the objects is however unknown. The approach is able to efficiently recognize the CF at each step of a human demonstration out of many hundreds of possible CFs. The applied measurement models do not consider *friction* forces, *inertia* forces or the *deformation* of the objects in contact. Therefore the approach can only deal with limited ($< \pm$ 20 [%]) inertia and friction forces, and a small ($< \pm$ 2 [%]) deformation of the objects in contact. Due to the accuracy and the maximum load of the used sensors, the allowed *size* of the polyhedral objects is bounded between roughly $> \pm 0.1$ $[m]$ and $< \pm 1.0$ $[m]$.

Compliant task generator The output of a compliant path planner or a human demonstration is given by a geometrical representation of a compliant path, in the form of a set of six-dimensional poses and their corresponding CFs. The hybrid compliant robot controller however, expects an instantaneous desired pose, twist and wrench at each time step, together with their twist and wrench spaces. In order to execute a planned or demonstrated compliant path, the planner or demonstration primitives need to be converted into instantaneous controller primitives.

Chapter 5 presents the *Compliant Task Generator*, the first general and automated approach that links both planning and programming by human demonstration to controller efforts in active compliant motion. It applies to any compliant motion between rigid *polyhedral* objects with a known geometry, and is more general and simple than previously presented ad-hoc or rule-based methods. Moreover, the compliant task generator is invariant with respect to changes of reference frame, scale and physical units. The conversion of the discrete planner and demonstration primitives into a continuous path represented by the controller primitives, is processed separately for the twist and wrench space. A task-specific input of two magnitudes and norms (or four magnitudes for the twist and wrench components) is sufficient to specify the desired dynamic interaction between the manipulated object and its environment. With the task-specific input the approach is able to fully automatically convert the planner or demonstration primitives into the controller primitives. The conversion within the twist space uses the specified desired magnitude and norm for the twist; the conversion within the wrench space uses the specified desired magnitude and norm for the wrench. The result is the immediate execution of an off-line planned or demonstrated compliant path by a robot manipulator, under active force control. In the real world experiment, the approach proved both efficient and effective for all provided

compliant paths, including complex CFs and CF transitions.

Online state estimation A general compliant motion task includes continuous CF-compliant motions and discrete CF transitions. During a compliant motion withing a single CF, the contact constraints between the manipulated object and its environment change in a *continuous* way; a compliant motion that includes a transition between two CFs causes a *discrete* jump in the contact constraints. According to the continuous and discrete changes in contact constraints, the hybrid force/velocity controller of the robot manipulator is adapted. While a CF compliant motion requires a continuous change of the control algorithm, a CF transition requires a discrete change in control algorithm. The desired contact constraints are given in the task description of the compliant motion task. However, the hybrid controller that executes the compliant motion task not only requires the knowledge of the *desired* contact constraints, but even more so requires the knowledge of the *actual* contact constraints that occur during the execution of the task. In an ideal task execution the desired and actual contact constraints are identical throughout the whole execution. In a real world execution however, the uncertainties in the unstructured environment cause the desired and actual contact constraints to differ. The work in this thesis makes the execution of compliant motion tasks more robust to uncertainties in the environment by estimating the CF and feeding it back to the controller.

Chapter 6 explains how the presented particle filter approach is used to monitor discrete contact state transitions online. The efficiency of the underlying algorithms of the particle filter allow the online estimation of CFs and CF transitions during the execution of an active compliant motion task. The CF estimation provides a feedback to the generator and controller components about actual transitions between different sub-tasks, as shown in Figure 8.1. Depending on the number of contact constraints of the actual CF and the desired CF, a different control strategy is selected during the execution of the compliant motion task.

8.3 Limitations and future work

Despite its merits, this work constitutes only a small step towards a fully autonomous compliant motion system. This section presents possible "next steps" towards a fully autonomous compliant motion system.

Recovery by re-planning In this thesis a particle filter is used for the online estimation of the CFs and the CF transitions that occur during the execution of a compliant motion task. The knowledge of the actual CF is used to (i) select an optimal control strategy for the current contact constraints,

and (ii) monitor the CF sequence of the task. When the actual CF sequence differs from the desired CF sequence, the current approach is only able to *detect* the problem (and go to an error state), but is not able to *recover* from the problem. With the knowledge of the actual CF and the desired CF, the next logical step is to use the compliant path planner to generate a path from the actual to the desired CF. Following this path, the execution can recover from the error state and continue the execution of the desired compliant path. This recovery strategy requires an extra feedback link from the estimation component to the task specification component, as shown by the dotted line in Figure 8.1.

Other sensors and active sensing Despite the different sensors used for the execution of a compliant motion task, not all task parameters are always sufficiently observed. A first solution comes from integrating additional sensors that provide measurements complementary to the other sensor measurements. The execution of compliant motion tasks can benefit greatly from the integration of camera vision. While wrench and pose sensors provide very local information, a camera provides a more general "overview" of the task. A camera image can link the different local features that were observed by the other sensors, into the global model. Another solution to improve the observability of the task parameters comes form active sensing, where the compliant path is adapted online to specifically observe certain unknown parameters. Although a general active sensing approach poses many challenges, in an interesting and less challenging first step an active sensing approach can generate specific motions to distinguish between two possible discrete CFs, based on the local twist and wrench spaces of the CFs. When applying a motion that at the same time lies within the twist space of the first CF and the wrench space of the second CF, the resulting sensor measurements contain information that makes it possible to distinguish between the two possible CFs.

Online estimation of continuous parameters The particle filter presented in this thesis simultaneously estimates the continuous geometric parameters and the discrete CF. Currently, only the discrete CF is provided to the controller selection in a feedback loop. The increased knowledge about the continuous geometric parameters already improves the CF recognition, but could also improve the geometric model used by the generator and controller components. The more accurate the geometric models used by these components, the better their performance. An interesting research topic would be the closed loop behavior of the system where the geometric model of the generator and controller is adapted online, based on sensor measurements.

Compliant motion primitives While the research in compliant motion (successfully) focussed on the robustness against ever larger geometrical uncertainties, compliant motion tasks only have a limited robustness against unmodelled effects such as friction and deformation. These effects however, are extremely complex and not easily added to the compliant motion models. An alternative approach—instead of improving the contact models—is to adapt the compliant motion primitives. Currently only one compliant motion primitive, the sliding of the manipulated object along the surface of the environmental object, is applied. New compliant motion primitives such as "hopping" or "tactile compliant motion" can simply avoid the unmodelled effects of friction and deformation. This could allow compliant motion to be introduced in a large range of (now unreachable) real world applications such as construction works.

General shapes The object models used in this thesis are limited to polyhedral objects, limiting their practical use to only a few applications. While modelling objects with an arbitrary shape is a considerable challenge, many objects that are relevant in industrial assembly are designed in a CAD environment, where often only a few basic shapes such as cylinders, spheres or circle segments are used to create the model. Therefore, the relatively small effort of integrating a few basic shapes would already bring the field of active compliant motion a large step closer to many practical and relevant industrial applications. Moreover, current research efforts in compliant motion focus on the contact state description and the automatic generation of a contact state graph between curved objects, moving the field of active compliant motion towards industrial applications.

References

Amato, N., O. Bayazit, L. Dale, C. Jones, and D. Vallejo (1998). Choosing good distance metrics and local planners for probabilistic roadmap methods. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Leuven, Belgium.

Ambler, A. and D. Corner (1982). *RAPT1 users manual*. Edinburgh, Scotland: Department of Artificial Intelligence, University of Edinburgh.

Ambler, A. P. and R. J. Popplestone (1975). Inferring the positions of bodies from specified spatial relationships. *Artificial Intelligence 6*, 157–174.

Asada, H. (1990). Teaching and learning of compliance using neural nets: Representation and generation of nonlinear compliance. In *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, Cincinnati, OH, pp. 1237–1244.

Asada, H. (1993, Dec). Representation and learning of nonlinear compliance using neural nets. *IEEE Transactions on Robotics and Automation 9*(6), 863–867.

Asada, H. and H. Izumi (1989). Automatic program generation from teaching data for the hybrid control of robots. *ITRA 5*(2), 166–173.

Baeten, J. (2001). *Integration of vision and force for robotic servoing*. Ph. D. thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Ball, R. S. (1871). The theory of screws—a geometrical study of the kinematics, equilibrium, and small oscillations of a rigid body. *Trans. of the Royal Irish Academy 25*, 137–217.

Bar-Shalom, Y. and X. Li (1993). *Estimation and Tracking, Principles, Techniques, and Software*. Artech House.

Barraquand, J. and J.-C. Latombe (1991). Robot motion planning: A distributed representation approach. *The International Journal of Robotics Research 10*(6), 628–649.

Bicchi, A., K. Salisbury, and D. L. Brock (1993). Contact sensing from force measurements. *The International Journal of Robotics Research 12*(3), 249–262.

Blajer, W. (1997). A geometric unification of constrained system dynamics. *Multibody System Dynamics 1*(1), 3–21.

Blume, C. and W. Jakob (1985). *PASRO: PASCAL for robots*. Springer Verlag.

Bonner, S. and K. G. Shin (1982). A comparative study of robot languages. *IEEE Computer 15*(12), 82–96.

Breidenbach, A., R. Koeppe, and G. Hirzinger (1996). Skill representation and acquisition of compliant motions using a teach device. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Osaka, Japan, pp. 897–904.

Brock, O. and O. Khatib (1999). High-speed navigation using the global dynamic window approach. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, MI, pp. 341–346.

Brooks, R. and T. Lozano-Pérez (1983). A subdivision algorithm in configuration space for findpath with rotation. In *ICAI*, pp. 799–806.

Bruyninckx, H. (1995). *Kinematic Models for Robot Compliant Motion with Identification of Uncertainties*. Ph. D. thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Bruyninckx, H. (2001). Open RObot COntrol Software. `http://www.orocos.org/`.

Bruyninckx, H. and J. De Schutter (1996). Specification of force-controlled actions in the "Task Frame Formalism": A survey. *IEEE Transactions on Robotics and Automation 12*(5), 581–589.

Bruyninckx, H. and J. De Schutter (1997). Where does the Task Frame go? In Y. Shirai and S. Hiroshe (Eds.), *Robotics Research, The Eight International Symposium*, Shonan, Japan, pp. 55–65. Springer-Verlag.

Bruyninckx, H., J. De Schutter, and S. Dutré (1993a). Kinematic models of rigid body interactions for compliant motion tasks in the presence of uncertainties. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GA, pp. 1007–1012.

Bruyninckx, H., J. De Schutter, and S. Dutré (1993b). The "reciprocity" and "consistency" based approaches to uncertainty identification for compliant motions. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GA, pp. 349–354.

Bruyninckx, H., J. De Schutter, T. Lefebvre, K. Gadeyne, P. Soetens, J. Rutgeerts, P. Slaets, and W. Meeussen (2003). Building blocks for

slam in autonomous compliant motion. In R. Chatila, P. Dario, and O. Khatib (Eds.), *Robotics Research, the 11th International Symposium*, Siena, Italy, pp. 432–441.

Bruyninckx, H., P. Soetens, and B. Koninckx (2003). The real-time motion control core of the Orocos project. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipeh, Taiwan, pp. 2766–2771.

Cabras, S., M. Eugenia Castellanos, W. Meeussen, R. Montes, and E. Staffetti (2006). Using bayesian filtering to simultaneous parameter and state estimation in robot manipulator programming by demonstration. In *Valencia International Meeting on Bayesian Statistics*, Benidorm, Spain.

Castellani, A., D. Botturi, and P. Fiorini (2004). Hybrid hmm/svm model for the analysis and segmentation of teleoperation tasks. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, U.S.A., pp. 2918–2923.

Charniak, E. (1991). Bayesian Networks without tears. *AI magazine 12*(4), 50–63.

Chen, J. (2001). *Coping with demonstration suboptimality in robot programming by demonstration*. Ph. D. thesis, Australian National University.

Chen, J. (2005). Constructing task-level assembly strategies in robot programming by demonstration. *The International Journal of Robotics Research 24*, 1073–1085.

Chiaverini, S. and L. Sciavicco (1993). The parallel approach to force/position control manipulators. *IEEE Transactions on Robotics and Automation 9*, 289–293.

Claassen, W. and K. Serdons (2006). Programmeren van sensorgebaseerde robottaken op basis van menselijk voordoen.

Conti, F. and O. Kathib (2006). A new actuation approach for haptic interface design. In *Proceedings of the International Symposium on Experimental Robotics*, Rio de Janeiro, Brazil. in press.

Corke, P. and R. Kirkham (1993). The arcl robot programming system. In *Robots for Competitive Industries*, Brisbane, pp. 484–493.

Crangle, C., S. Michalowski, and L. Ling (1987). Experimental study of a natural-language interface to an instructable robotic aid for the severely disabled. In *ACRT*, San Jose, CA, USA.

Danthine, A. and M. Geradin (Eds.) (1984). *Advanced software in robotics*. Elsevier Science Publishers B.V. (North Holland).

Davison, A. J., Y. G. Cid, and N. Kita (2004). Real-time 3D SLAM with wide-angle vision. In *Proc. 5th IFAC Symposium on Intelligent Autonomous Vehicles*.

De Luca, A. and C. Manes (1994). Modeling of robots in contact with a dynamic environment. *IEEE Transactions on Robotics and Automation 10*(4), 542–548.

De Schutter, J. (1988). Improved force control laws for advanced tracking applications. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, PA, pp. 1497–1502.

De Schutter, J. and H. Bruyninckx (1996). Force control of robot manipulators. In *The Control Handbook*, pp. 1351–1358. CRC Press.

De Schutter, J. and H. Bruyninckx (2000). Force control of robot manipulators. In W. S. Levine (Ed.), *Control System Applications*, pp. 177–184. CRC Press. Reprint of (De Schutter and Bruyninckx 1996).

De Schutter, J., H. Bruyninckx, S. Dutré, J. De Geeter, J. Katupitiya, S. Demey, and T. Lefebvre (1999). Estimating first order geometric parameters and monitoring contact transitions during force controlled compliant motion. *The International Journal of Robotics Research 18*(12), 1161–1184.

De Schutter, J. and H. Van Brussel (1988a, Aug). Compliant Motion I, II. *The International Journal of Robotics Research 7*(4), 3–33.

De Schutter, J. and H. Van Brussel (1988b). Compliant robot motion II. A control approach based on external control loops. *The International Journal of Robotics Research 7*(4), 18–33.

Debus, T., P. Dupont, and R. Howe (2002). Contact State Estimation using Multiple Model Estimation and Hidden Markov Models. In *Proceedings of the International Symposium on Experimental Robotics*, Sant'Angelo d'Ischia, Italy. in press.

Debus, T., P. Dupont, and R. Howe (2004). Contact State Estimation using Multiple Model Estimation and Hidden Markov Models. *The International Journal of Robotics Research 23*(4–5), 399–413.

Delson, N. and H. West (1993). Robot programming by demonstration: Subtask compliance controller identification. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Yokohama, Japan, pp. 33–41.

Delson, N. and H. West (1996). Robot programming by human demonstration: Adaptation and inconsistency in constrained motion. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 30–36.

Demey, S. (1996). *Shape identification and shape matching for compliant motion based on invariant differential shape descriptions*. Ph. D. thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Desai, R. S. (1989). *On fine motion in mechanical assembly in presence of uncertainty*. Ph. D. thesis, Department of Mechanical Engineering, University of Michigan.

Desai, R. S. and R. A. Volz (1989). Identification and verification of termination conditions in fine motion in presence of sensor errors and geometric uncertainties. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, pp. 800–807.

Dillmann, R. (2004). Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems 2-3*(47), 109–116.

Dillmann, R., M. Kaiser, and A. Ude (1995). Acquisition of elementary robot skills from human demonstration. In *SIRS*, Pisa, Italia, pp. 185 – 192.

Donald, B. R. (1985). On motion planning with six degrees of freedom: solving the intersection problems in configuration space. In *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, St. Louis, MS, pp. 536–541.

Doty, K. L., C. Melchiorri, and C. Bonivento (1993). A theory of generalized inverses applied to robotics. *The International Journal of Robotics Research 12*(1), 1–19.

Doucet, A., N. J. Gordon, and V. Krishnamurthy (2001, march). Particle Filters for State Estimation of Jump Markov Linear Systems. *IEEE Transactions on Signal Processing 49*(3), 613–624.

Duffy, J. (1990). The fallacy of modern hybrid control theory that is based on "orthogonal complements" of twist and wrench spaces. *Journal of Robotic Systems 7*(2), 139–144.

Dupont, P. E., T. M. Schulteis, P. Millman, and R. D. Howe (1999). Automatic identification of environment haptic properties. *Presence: Teleoperators and Virtual Environments 8*(4), 392–409.

Dutré, S., H. Bruyninckx, and J. De Schutter (1996). Contact identification and monitoring based on energy. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 1333–1338.

Eberman, B. (1997). A model-based approach to Cartesian manipulation contact sensing. *The International Journal of Robotics Research 16*(4), 508–528.

Eberman, B. S. and J. K. Salisbury, Jr. (1994). Application of change detection to dynamic contact sensing. *The International Journal of Robotics Research 13*(5), 369–394.

Ehrenmann, M., R. Zöllner, O. Rogalla, S. Vacek, and R. Dillmann (2002). Programming service tasks in household environments by human demonstration. In *Proceedings of the IEEE International Workshop on Robot and Human Interactive Cummunication*, Berlin, Germany, pp. 25–27.

Ekvall, S., D. Aarno, and D. Kragic (2006). Online task recognition and real-time adaptive assistance for computer-aided machine control. *IEEE Transactions on Robotics 22*(5), 1029–1033.

Farahat, A. O., B. S. Graves, and J. C. Trinkle (1995). Identifying contact formations in the presence of uncertainty. In *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Pittsburgh, PA.

Fedele, A., A. Fioretti, C. Manes, and G. Ulivi (1993). On-line processing of position and force measures for contour identification and robot control. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GA, pp. 369–374.

Feng, P. and S. Joseph M. (2003). Efficient contact state graph generation for assembly applications. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Volume 2, Taipeh, Taiwan, pp. 2592–2598.

Finkemeyer, B., T. Kröger, and F. M. Wahl (2003). Placing of objects in unknown unvironments. In *MMAR*, Miedzyzdroje, Poland, pp. 975–980.

Fisher, W. D. and M. S. Mujtaba (1992). Hybrid position/force control: A correct formulation. *The International Journal of Robotics Research 11*(4), 299–311.

Fox, D., W. Burgard, F. Dellaert, and S. Thrun (1999, July). Monte Carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99).*

Fukuda, T., K. Kosuge, and H. Asads (1991). Acquisition of human skill for robotic systems. In *Proceedings of the International Symposium on Intelligent Control*, Arlington, VA, USA, pp. 469–474.

Gadeyne, K. (2001). BFL: Bayesian Filtering Library. `http://people.mech.kuleuven.ac.be/~kgadeyne/bfl`.

Gadeyne, K. (2005, September). *Sequential Monte Carlo methods for rigorous Bayesian modeling of Autonomous Compliant Motion*. Ph. D. thesis,

Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Gadeyne, K., T. Lefebvre, and H. Bruyninckx (2005). Bayesian hybrid model-state estimation applied to simultaneous contact formation recognition and geometrical parameter estimation. *The International Journal of Robotics Research 24*(8), 615–630.

Geschke, C., B. Shimano, and C. Spalding (1984). Val-ii: a robot programming language and control system. In *SME Robots Conference*, Detrait, MI, pp. 103–119.

Gilbert, E. (1988). A fast procedure for computing the distance between complex objects in three-dimensional space. *IJRA 4*(2), 193–203.

González-Linares, J., N. Guil, P. Perez, J. Ehrenmann, and R. Dillmann (1999). An efficient image processing algorithm for high level skill acquisition. In *ISATP*, Porto, Portugal.

Groover, M. P., M. Weiss, R. N. Nagel, and N. G. Odrey (1986). *Industrial Robotics: Technology, Programming, and Applications*. McGraw-Hill Companies.

Gupta, K. (1995). Motion planning for re-orientation using finger tracking: landmarks in $so(3) \times \omega$. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, pp. 446–451.

Halperin, D. and M. Sharir (1996). Near-quadratic algorithm for planning the motion of a polygon in a polygonal environment. *DCG 16*, 121–134.

Hannaford, B. and P. Lee (1991). Hidden Markov Model analysis of force/torque information in telemanipulation. *The International Journal of Robotics Research 10*(5), 528–539.

Hara, K. and R. Yokogawa (1992). Recognition of state in peg-in-hole by fuzzy schema. *Journal of Advanced Automation Technology 4*(3), 134–139.

Harima, T. and H. West (1992). Natural robot programming system. In *Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Raleigh, NC, pp. 750–756.

Higham, N. J. (1986). Computing the polar decomposition—with applications. *SIAM J. Sci. Stat. Comput. 7*(4), 1160–1174.

Hirai, R., H. Noguchi, and K. Iwata (1996). Human-demonstration based approach to the recognition of process state transitions in insertion of deformable tubes. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN.

Hirai, S. and H. Asada (1990). A model-based approach to the recognition of assembly process states using the theory of polyhedral convex cones.

In *Japan USA Symposium on Flexible Automation*, Kyoto, Japan, pp. 809–819.

Hirzinger, G., B. Brunner, J. Dietrich, and J. Heindl (1993). Sensor based space robotics—ROTEX and its telerobotic features. *IEEE Transactions on Robotics and Automation 9*(5), 649–663.

Hogan, N. (1985). Impedance control: An approach to manipulation. Parts I-III. *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control 107*, 1–24.

Hogan, N. (1987). Stable execution of contact tasks using impedance control. In *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, Raleigh, NC, pp. 1047–1054.

Hovland, G. and B. J. McCarragher (1996). Frequency-domain force measurements for discrete event contact recognition. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 1166–1171.

Hovland, G. and B. J. McCarragher (1998a). Controlling sensory perception for indoor navigation. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Leuven, Belgium, pp. 2211–2216.

Hovland, G. E. and B. J. McCarragher (1998b, Feb). Hidden Markov Models as a Process Monitor in Robotic Assembly. *The International Journal of Robotics Research 17*(2), 153–168.

Ikeuchi, K. and T. Suehiro (1994). Toward an assembly plan from observation. *IEEE Transactions on Robotics and Automation 10*, 368–385.

Ji, X. and J. Xiao (2001a, July). Planning motions compliant to complex contact states. *The International Journal of Robotics Research 20*(6), 446–465.

Ji, X. and J. Xiao (2001b). Random sampling of contact configurations in two-pc contact formations. In K. L. B.R. Donald and D. Rus (Eds.), *New Directions in Algorithmic and Computational Robotics*, Boston. Kluwer.

Jordan, M. I. (Ed.) (1999). *Learning in Graphical Models*. Adaptive Computation and Machine Learning. London, England: MIT Press. ISBN 0262600323.

Kaiser, M. and R. Dillmann (1996). Building elementary robot skills from human demonstration. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 2700–2705.

Kaiser, M., H. Friedrich, and R. Dillmann (1995). Obtaining good performance from a bad teacher. In *ICML*. Morgan Kaufmann.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering 82*, 34–45.

Kato, K. and S. Hirose (2000). Proposition and basic experiments of shape feedback master-slave arm -on the application for the demining robots-. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp. 2334–2339.

Kavraki, L. and J. Latombe (1998). Probabilistic roadmaps for robot path planning. In K. Gupta and A. P. del Pobil (Eds.), *Practical Motion Planning in Robotics*, pp. 33–53. John Wiley & Sons.

Kavraki, L. and J.-C. Latombe (1994). Randomized preprocessing of configuration space for fast path planning. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, pp. 2138–2145.

Kazerooni, H. (1990). Human-robot interaction via transfer of power and information signals. *ITSMC 20*(2), 450–463.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research 5*(1), 90–99.

Kitagaki, K., T. Ogasawara, and T. Suehiro (1993). Methods to detect contact state by force sensing in an edge mating task. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GA, pp. 701–706.

Klein, F. (1871). Notiz betreffend den Zusammenhang der Linien-Geometrie mit der Mechanik starrer Körper. *Mathematische Annalen 4*, 403–415.

Knoop, S., S. Vacek, and R. Dillman (2006). Sensor fusion for 3d human body tracking with an articulated 3d body model. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, Orlando, U.S.A., pp. 1686– 1691.

Knoop, S., S. Vacek, K. Steinbach, and R. Dillman (2006). Sensor fusion for model based 3d tracking. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, pp. 524–529.

Koga, Y. and J.-C. Latombe (1994). On multi-arm manipulation planning. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, pp. 945–952.

Kröger, T., B. Finkemeyer, M. Heuck, and F. M. Wahl (2004). Adaptive implicit hybrid force/pose control of industrial manipulators: Compliant motion experiments. In *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, pp. 816–821.

Kröger, T., B. Finkemeyer, and F. M. Wahl (2004). A task frame formalism for practical implementations. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, U.S.A., pp. 5218–5223.

Kuniyoshi, Y., M. Inaba, and H. Inoue (1994). Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE Transactions on Robotics and Automation 10*, 799–822.

Latombe, J. C. (1991). *Robot motion planning*, Volume 124 of *Int. Series in Engineering and Computer Science*. Boston, MA: Kluwer Academic Publishers.

Latombe, J. C. (1999). Motion planning: A journey of robots, molecules, digital actors, and other artifacts. *The International Journal of Robotics Research 18*(11), 1119– 1128. Invited paper.

Latombe, J. C., C. Laugier, J. Lefebvre, and E. Mazer (1984). The lm robot programming system. *The International Journal of Robotics Research 1*(1), 377–391.

Latombe, J. C. and R. Motwani (1997). Path planning in expansive configuration spaces. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, pp. 2719–2726.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge University Press.

Lefebvre, T., H. Bruyninckx, and J. De Schutter (2003). Polyhedral contact formation modeling and identification for autonomous compliant motion. *IEEE Transactions on Robotics and Automation 19*(1), 26–41.

Lefebvre, T., H. Bruyninckx, and J. De Schutter (2005). Online statistical model recognition and state estimation for autonomous compliant motion. *IEEE Transactions on Systems, Man, and Cybernetics: Part C 35*(1), 16–29.

Lieberman, L. and M. Wesley (1977). Autopass: an automatic programming system for computer controlled mechanical assembly. *IBM Journal of Research and Development 21*(4), 321–333.

Lipkin, H. and J. Duffy (1988). Hybrid twist and wrench control for a robotic manipulator. *Transactions of the ASME, Journal of Mechanisms, Transmissions, and Automation in Design 110*, 138–144.

Lloyd, J., J. S. Beis, D. K. Pai, and D. G. Lowe (1999). Programming contact tasks using a reality-based virtual environment integrated with vision. *IEEE Transactions on Robotics and Automation 15*(3), 423–434.

Lozano-Pérez, T. (1976a). The design of a mechanical assembly system. Technical Report Technical Report 397, MIT Artificial Intelligence Laboratory.

Lozano-Pérez, T. (1976b). Spatial planning: A configuration space approach. *Trans. on Computers 32*(2), 108–120.

Lozano-Pérez, T. and T. Winston (1977). Lama: A language for automatic mechanical assembly. In *ICAI*, pp. 710–716.

Luo, Q. and J. Xiao (2004). Physically accurate haptic rendering and virtual assembly. In *Transactions of North America Manufacturing Research*, UNC Charlotte, Charlotte, NC USA.

Luo, Q. and J. Xiao (2005, June). Modeling complex contacts involving deformable objects for haptic and graphic rendering. In *RSS*, Cambridge, USA.

Manes, C. (1992). Recovering model consistence for force and velocity measures in robot hybrid control. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, Nice, France, pp. 1276–1281.

Marcelo, H., W. Lin, and S.-Y. Lim (1999). A walk-through programmed robot for welding in shipyards. *The Industrial Robot 26*(5), 377–388.

Mason, M. T. (1979). Compliance and force control for computer controlled manipulators. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA.

Mason, M. T. (1981). Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics SMC-11*(6), 418–432.

McCarragher, B. J. and H. Asada (1992). The discrete event control of robotic assembly tasks. In *Proceedings of the 31st Conference on Decision and Control*, Tucson, AZ, pp. 1406–1407.

McCarragher, B. J. and H. Asada (1993). Qualitative template matching using dynamic process models for state transition recognition of robotic assembly. *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control 115*, 261–269.

McCarragher, B. J. and H. Asada (1995). The discrete event control of robotic assembly tasks. *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control 117*(3), 384–393.

McCarragher, B. J. and G. Hovland (1998). Hidden Markov Models as process monitor in robotics assembly. *The International Journal of Robotics Research 17*(2), 153–158.

Meeussen, W., J. De Schutter, H. Bruyninckx, J. Xiao, and E. Staffetti (2005a). Integration of planning and execution in force controlled compliant motion. In *Proceedings of the 2005 IEEE/RSJ International Con-*

ference on Intelligent Robots and Systems, Edmonton, Canada, pp. 2550–2555.

Meeussen, W., J. De Schutter, H. Bruyninckx, J. Xiao, and E. Staffetti (2005b). Integration of planning and execution in force controlled compliant motion. *IEEE Transactions on Robotics*. in review.

Meeussen, W., J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter (2006a). Bayesian contact state segmentation for programming by human demonstration in compliant motion tasks. In *Proceedings of the International Symposium on Experimental Robotics*, Rio de Janeiro, Brazil. in press.

Meeussen, W., J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter (2006b). Contact state segmentation using particle filters for programming by human demonstration in compliant motion tasks. *IEEE Transactions on Robotics*. in press.

Meeussen, W., J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter (2006c). Particle filters for hybrid event sensor fusion with 3d vision and force. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, pp. 518–523.

Meeussen, W., J. Xiao, J. De Schutter, H. Bruyninckx, and E. Staffetti (2004). Automatic verification of contact states taking into account manipulator constraints. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, U.S.A., pp. 3583–3588.

Mihaylova, L., T. Lefebvre, E. Staffetti, H. Bruyninckx, and J. De Schutter (2002). Contact transitions tracking during force-controlled compliant motion using an interacting multiple model estimator. *Information and Security 9*, 114–129.

Mimura, N. and Y. Funahashi (1994). Parameter identification of contact conditions by active force sensing. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, pp. 2645–2650.

Miura, J. and K. Ikeuchi (1998). Task-oriented generation of visual sensing strategies in assembly tasks. *PAMI 20*, 126–138.

Montemerlo, M. and S. Thrun (2003). Simultaneous localization and mapping with unknown data association using fastslam. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipeh, Taiwan, pp. 1985–1991.

Montemerlo, M., S. Thrun, D. Koller, and B. Wegbreit (2003). Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization

and mapping that provably converges. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico.

Murray, R. M., Z. Li, and S. S. Sastry (1994). *A mathematical introduction to robotic manipulation*. Boca Raton, FL: CRC Press.

Nakamura, Y. (1991). *Advanced robotics: redundancy and optimization*. Reading, MA: Addison-Wesley.

Noyes, M. and T. Sheridan (1984). A novel predictor for telemanipulation through a time delay. In *ACMC*, NASA Ames Research Center, CA.

Nuttin, M. and H. Van Brussel (1996). Applications of neural nets in robot control, illustrated with some experiments. *Jounal A 37*(3), 28–33.

Nuttin, M., H. Van Brussel, J. Peirs, and S. Soembagijo, A. S.and Sonck (1995). Learning the peg-into-hole assembly operation with a connectionist reinforcement technique. In *31 International CIRP Workshop on Learning in Intelligent Manufacturing Systems*, pp. 335– 357.

Ogata, K. and T. Takahashi (1994). A geometrical approach to task understanding and playback: compact and robust task description for complex robot environments. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, pp. 848–854.

Ohwovoriole, M. S. and B. Roth (1981). An extension of screw theory. *Transactions of the ASME, Journal of Mechanical Design 103*(4), 725–735.

Pelich, C. and F. Wahl (1997). Zero++: An oop environment for multiprocessor robot control. *IEEE Journal of Robotics and Automation 12*(2), 49–57.

Pembeci, I. and G. D. Hager (2002). A comparative review of robot programming languages. Technical report, Johns Hopkins University. CIRL Lab Technical Report.

Penrose, R. (1955). A generalized inverse for matrices. *Proc. Cambridge Philos. Soc. 51*, 406–413.

Popplestone, R. (1978). Rapt, a language for describing assemblies. *IR 5*(3), 131–137.

Popplestone, R. J., R. Weiss, and Y. Liu (1988). Using characteristic invariants to infer new spatial relationships. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, Philadelphia, PA, pp. 1107–1112.

Princen, P. and B. Stallaert (2004). Ontwerp van een demonstratietool voor robotprogrammatie door voordoen.

Quinlan, S. and O. Khatib (1993). Elastic bands: Connecting path planning and control. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Volume 2, Atlanta, GA, pp. 802–807.

Raibert, M. and J. J. Craig (1981). Hybrid position/force control of manipulators. *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control 102*, 126–133.

Raju, G., G. C. Verghese, and T. B. Sheridan (1989). Design issues in 2-port network models of bilateral remote manipulation. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, pp. 1316–1321.

Ránky, C. (1985). *Robot modelling: Control and applications with software*. Kempston, Bedford, England: IFS Ltd./Springer.

Rutgeerts, J., T. De Laet, R. Smits, H. Bruyninckx, and J. De Schutter (2006). Constraint-based task specification and estimation for sensor-based robot systems: a practical framework. Technical report.

Rutgeerts, J., P. Slaets, F. Schillebeeckx, W. Meeussen, B. Stallaert, P. Princen, T. Lefebvre, H. Bruyninckx, and J. De Schutter (2005). A demonstration tool with Kalman Filter data processing for robot programming by human demonstration. In *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, pp. 3918–3923.

Schimmels, J. M. and M. A. Peshkin (1992). Admittance matrix design for force-guided assembly. *IEEE Transactions on Robotics and Automation 8*(2), 213–227.

Schulz, D., W. Burgard, and D. Fox (2003). People tracking with mobile robots using sample-based joint probabilistic data association filters. *The International Journal of Robotics Research 22*(2), 99–116.

Schwarz, J. and M. Sharir (1983). On the 'piano movers' problem: Ii. general techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics 4*, 298–351.

Sciavicco, L. and B. Siciliano (1996). *Modeling and Control of Robot Manipulators*. McGraw-Hill.

Sikka, P. and B. McCarragher (1996). Monitoring contact using clustering and discriminant functions. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 1351–1356.

Simons, J., H. Van Brussel, J. De Schutter, and J. Verhaert (1982). A self-learning automaton with variable resolution for high precision assembly by industrial robots. *IEEE Transactions on Automatic Control AC-27*(5), 1109–1113.

Skubic, M. (1997). *Transferring assembly skills to robots: learning force sensory patterns and skills from human demonstration*. Ph. D. thesis, Texas A&M University.

Skubic, M. and R. Volz (1996). Identifying contact formations from sensory patterns and its applicability to robot programming by demonstration. In *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Osaka, Japan, pp. 458–464.

Skubic, M. and R. Volz (2000, October). Identifying single-ended contact formations from force sensor patterns. *IEEE Transactions on Robotics and Automation 16*(5), 597–603.

Slaets, P., T. Lefebvre, H. Bruyninckx, and J. De Schutter (2006). Incremental building of a polyhedral feature model for programming by human demonstration of force controlled tasks. *The International Journal of Robotics Research*.

Smits, R., H. Bruyninckx, W. Meeussen, J. Baeten, P. Slaets, and J. De Schutter (2006). Model based position-force-vision sensor fusion for robot compliant motion control. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, pp. 501–506.

Soetens, P. (2006, May). *A Software Framework for Real-Time and Distributed Robot and Machine Control*. Ph. D. thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Sorenson, H. W. (1985). *Kalman filtering: theory and application*. New York, NY: IEEE Press.

Staffetti, E., W. Meeussen, and J. Xiao (2005). A new formalism to characterize contact states involving articulated polyhedral objects. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, pp. 3630–3635.

Steinhage, A. and T. Bergener (2000). Learning by doing: a dynamic architecture for generating adaptive beharioral sequences. In *NC*, Los Angeles, California, pp. 813–820.

Sturges, Jr., R. H. and S. Laowattana (1995). Fine motion planning through constraint network analysis. In *ISATP*, Pittsburg, PA, pp. 160–170.

Takahashi, T. and H. Ogata (1992). Robotic assembly operation based on task-level teaching in a virtual environment. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, Nice, France, pp. 1083–1088.

Takeo, K. K. and T. Fukuda (1995). Unified approach for teleoperation of virtual and real environment: Manipulation based on reference dy-

namics. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, pp. 938–943.

Tang, P. and J. Xiao (2006, June). Generation of point-contact state space between strictly curved objects. In *RSS*, Cambridge, USA, pp. 31–38.

Tanizaki, H. (1996). *Nonlinear Filters. Estimation and Applications*. Springer-Verlag.

Tarn, T.-J. (1996). Path-based approach to integrated planning and control for robotic systems. *Automatica 32*(12), 1675–1687.

Taylor, R., P. Summers, and J. Meyer (1982). Aml: A manufacturing language. *The International Journal of Robotics Research 1*(3), 19–41.

Taylor, R. H. (1976). *Synthesis of manipulator control programs from task-level specifications*. Ph. D. thesis, Department of Computer Science, Stanford University, Stanford CA.

Thrun, S., W. Burgard, and D. Fox (2005). *Probabilistic Robotics*. MIT Press.

Tsujimura, T. and M. Yabuta (1989). Object detection by tactile sensing method employing force/torque information. *IEEE Transactions on Robotics and Automation 5*(4), 444–450.

Unimation Inc. (1979). *Guide to VAL, version 11*. Unimation Inc.

Vacek, S., S. Bergmann, U. Morh, and R. Dillmann (2006). Rule-based tracking of multiple lanes using particle filters. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, pp. 203–208.

Van de Poel, P., W. Witvrouw, H. Bruyninckx, and J. De Schutter (1993). An environment for developing and optimizing compliant robot motion tasks. In *Proceedings of the 1993 International Conference on Advanced Robotics*, Tokyo, Japan, pp. 713–718.

Wang, Q. (1999, November). *Programming of compliant robot motion by human demonstration*. Ph. D. thesis, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Wang, Q. and J. De Schutter (1998). Towards real-time robot programming by human demonstration for 6d force controlled actions. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, Leuven, Belgium, pp. 2256–2261.

Wang, Q., J. De Schutter, W. Witvrouw, and S. Graves (1996). Derivation of compliant motion programs based on human demonstration. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN, pp. 2616–2621.

Xiao, J. (1993). Automatic determination of topological contacts in the presence of sensing uncertainty. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GA, pp. 65–70.

Xiao, J. and X. Ji (2000). A divide-and-merge approach to automatic generation of contact states and planning of contact motions. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp. 750–756.

Xiao, J. and X. Ji (2001). On automatic generation of high-level contact state space. *The International Journal of Robotics Research 20*(7), 584–606.

Xiao, J., Q. Luo, and Y. Song (2003). Haptic modeling of contact formations and compliant motion. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipeh, Taiwan, pp. 2605–2610.

Xiao, J. and L. Zhang (1995). A general strategy to determine geometrically valid contact formations from possible contact primitives. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, pp. 2728–2733.

Xiao, J. and L. Zhang (1996). Towards obtaining all possible contacts—Growing a polyhedron by its location uncertainty. *IEEE Transactions on Robotics and Automation 12*(4), 553–565.

Xu, Y., J. Yang, and C. Chen (1994). Hidden markov model approach to skill learning and its applications to telerobotics. *ITRA 10*, 621–631.

Yoshikawa, T. and A. Sudou (1993). Dynamic hybrid position/force control of robot manipulators—on-line estimation of unknown constraint. *IEEE Transactions on Robotics and Automation 9*(2), 220–226.

Zhang, J., Y. von Collani, and A. Knoll (1999). Interactive assembly by a two arm robot agent. *RAS 1*(29), 91–100.

Zieliński, C. (1995). Control of a multi-robot system. In *MMAR*, Miedzyzdroje, Poland, pp. 603–608.

Zieliński, C. (2002). Motion generators in MRROC++ based robot controller. In *Proceedings 14th CISM-IFToMM Symposium on Robotics (RoManSy)*, Udine, Italy, pp. 299–306.

Index

Resume

Personal data

Wim Meeussen
July 10 1978, Leuven, Belgium
wim.meeussen@mech.kuleuven.be
http://people.mech.kuleuven.be/∼wmeeusse/

Education

- **2007-**: **Postdoctoral fellow** at the autonomous compliant motion group of the Katholieke Universiteit Leuven, Belgium.
- **2003 - 2006**: **Ph.D. in mechanical engineering** at the Katholieke Universiteit Leuven, Belgium.

  My research is situated in the area of autonomous compliant motion for force controlled robot tasks, such as deburring and assembly. The aim of this research is to make industrial robots work autonomously in less structured environments where they have to deal with inaccurately positioned tools and work pieces.

  2006: **Visiting Ph.D. student** at the Department of Statistics, Universidad Rey Juan Carlos, Madrid, Spain.
  2003 - 2004: **Visiting Ph.D. student** during one year at the Department of Computer Science, University of North Carolina at Charlotte, NC, USA.
- **2000 - 2002**: **Master of science in mechanical engineering** specialization mechatronics and machine design at the Katholieke Universiteit Leuven, Belgium.

  2000 - 2001: **ECTS** (European Credit Transfer System) one year study exchange with the Graz University of Technology, Austria.

  2001: **Athens** program at the Université de Marne la Valleé, Paris, France.
- **1997 - 2000**: **Bachelor in engineering** at the Katholieke Universiteit Leuven, Belgium.

List of Publications

Bruyninckx, H., J. De Schutter, T. Lefebvre, K. Gadeyne, P. Soetens, J. Rutgeerts, P. Slaets, and W. Meeussen (2003). Building blocks for slam in autonomous compliant motion. In R. Chatila, P. Dario, and O. Khatib (Eds.), *Robotics Research, the 11th International Symposium*, Siena, Italy, pp. 432–441.

Cabras, S., M. Eugenia Castellanos, W. Meeussen, R. Montes, and E. Staffetti (2006). Using bayesian filtering to simultaneous parameter and state estimation in robot manipulator programming by demonstration. In *Valencia International Meeting on Bayesian Statistics*, Benidorm, Spain.

Meeussen, W., J. De Schutter, H. Bruyninckx, J. Xiao, and E. Staffetti (2005a). Integration of planning and execution in force controlled compliant motion. *IEEE Transactions on Robotics*. in review.

Meeussen, W., J. De Schutter, H. Bruyninckx, J. Xiao, and E. Staffetti (2005b). Integration of planning and execution in force controlled compliant motion. In *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, pp. 2550–2555.

Meeussen, W., J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter (2006a). Bayesian contact state segmentation for programming by human demonstration in compliant motion tasks. In *Proceedings of the International Symposium on Experimental Robotics*, Rio de Janeiro, Brazil. in press.

Meeussen, W., J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter (2006b). Contact state segmentation using particle filters for programming by human demonstration in compliant motion tasks. *IEEE Transactions on Robotics*. in press.

Meeussen, W., J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter (2006c). Particle filters for hybrid event sensor fusion with 3d vision

and force. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, pp. 518–523.

Meeussen, W., J. Xiao, J. De Schutter, H. Bruyninckx, and E. Staffetti (2004). Automatic verification of contact states taking into account manipulator constraints. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, U.S.A., pp. 3583–3588.

Rutgeerts, J., P. Slaets, F. Schillebeeckx, W. Meeussen, B. Stallaert, P. Princen, T. Lefebvre, H. Bruyninckx, and J. De Schutter (2005). A demonstration tool with Kalman Filter data processing for robot programming by human demonstration. In *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, pp. 3918–3923.

Smits, R., H. Bruyninckx, W. Meeussen, J. Baeten, P. Slaets, and J. De Schutter (2006). Model based position-force-vision sensor fusion for robot compliant motion control. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Heidelberg, Germany, pp. 501–506.

Staffetti, E., W. Meeussen, and J. Xiao (2005). A new formalism to characterize contact states involving articulated polyhedral objects. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, pp. 3630–3635.

Robotbeweging in contact: van padplanning of programmatie door menselijk voordoen tot krachtgecontroleerde uitvoering

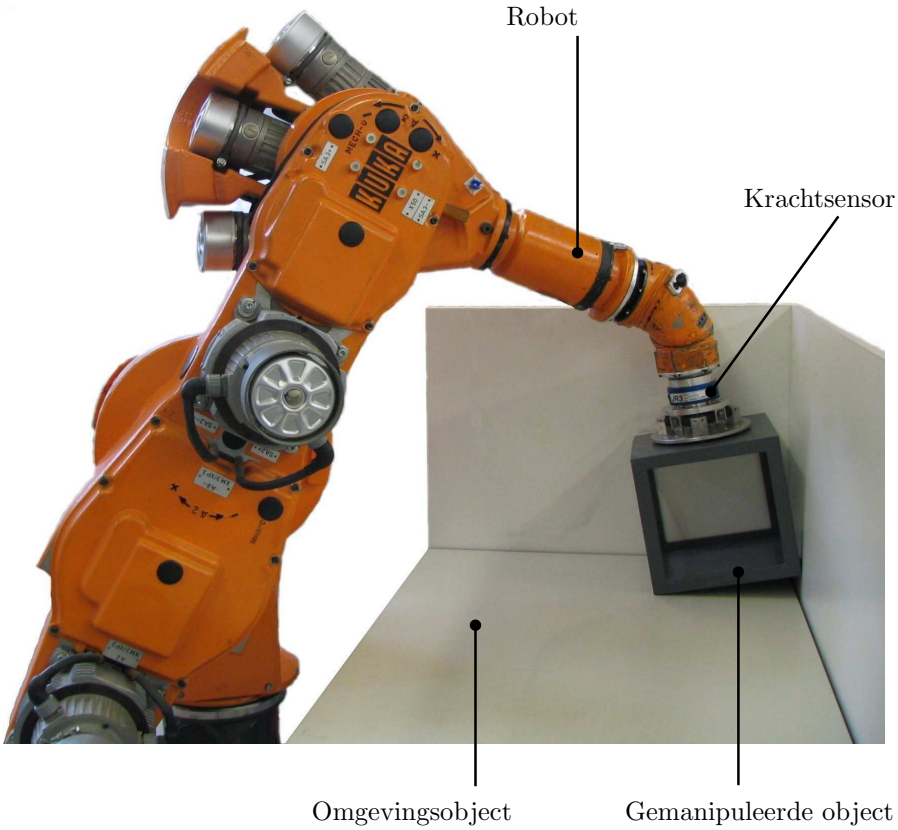
Nederlandse samenvatting

1 Inleiding

1.1 Situering

Industriële robots worden met succes ingezet voor industriële toepassingen, als flexibele positionering machines. Robots worden gebruikt voor het besparen van kosten, het verhogen van de productiviteit en de kwaliteit, of voor het uitvoeren van gevaarlijke en arbeidsintensieve taken. Meestal voeren robots echter eenvoudige positiegestuurde taken uit, zoals bijvoorbeeld puntlassen of het verplaatsen van objecten. De robot voert deze taken uit in een gestructureerde en gekende omgeving, speciaal aangepast aan de robot; alle objecten en obstakels in een gestructureerde omgeving zijn gekend, zodat de robot zijn taak blindelings kan uitvoeren. Voor elke nieuwe taak moet echter de omgeving opnieuw aangepast worden aan de taak, wat veel tijd vraagt en aanzienlijke kosten met zich meebrengt. Het gebruik van robots is hierdoor tot op vandaag beperkt tot zich herhalende taken in de massaproductie, zoals bijvoorbeeld in de autoindustrie.

Robots kunnen een taak sneller en nauwkeuriger uitvoeren dan een mens, maar kunnen zich onmogelijk meten met de menselijke intelligentie. Dit onderzoek richt zich daarom op het verhogen van de intelligentie en flexibiliteit van robots; door een robot uit te rusten met meerdere sensoren zoals bijvoorbeeld een camera, een krachtsensor of een afstandssensor, kan hij zijn omgeving waarnemen en ontstaat er een interactie tussen de robot en zijn omgeving. Dit laat een robot toe om ook in niet gestructureerde omgevingen nuttige taken uit te voeren, waardoor robots in de toekomst langzaam geïntroduceerd kunnen worden in opslagplaatsen, laboratoria, energiecentrales, ziekenhuizen, en zelfs in de ruimtevaart. In de realiteit echter is de intelligentie van een robot nog zeer beperkt, wat het op grote schaal inzetten van robots in ongestructureerde omgevingen nog belemmert. Dit geldt voor vele industriële toepassingen, en in het bijzonder voor assemblagetaken, waar er fysische contacten zijn tussen de robot en zijn omgeving.



Figuur 1: Actieve beweging in contact: de Kuka 361 met zes vrijheidsgraden manipuleert een kubus in contact met een hoek. De krachtinteractie is actief gecontroleerd.

1.2 Actieve beweging in contact

Dit proefschrift richt zich voornamelijk op taken in contact, waarbij een robot een object manipuleert in contact met de omgeving. De krachtinteractie leidt het gemanipuleerde object langs het oppervlak van het omgevingsobject; op deze manier kan een taak nauwkeurig uitgevoerd worden, zelfs in een onzekere omgeving. Figuur 1 toont een voorbeeld van een taak in contact, waarbij de robot een kubus manipuleert in contact met drie loodrecht op elkaar staande vlakken. Bijna alle industriële implementaties van taken in contact zijn nog steeds volledig positiegecontroleerd, en maken gebruik van een passieve flexibiliteit tussen de robot en het gemanipuleerde object. Deze passieve

flexibiliteit voorkomt dat kleine positioneringsfouten resulteren in grote ongewenste contactkrachten. Een passieve flexibiliteit maakt het mogelijk om een robot te gebruiken in een omgeving met kleine onzekerheden. In tegenstelling tot dit passief controleren van de contactkrachten, gebruikt *Active Compliant Motion* (ACM) een krachtsensor om de contactkrachten te meten, en het pad van de robot bij te sturen opdat de contactkrachten niet te groot zouden worden. In ACM simuleert het systeem dus een flexibiliteit tussen de robot en het gemanipuleerde object. Deze actieve controle laat de robot toe om in een omgeving met veel grotere onzekerheden een taak uit te voeren. De metingen van de krachtsensor kunnen ook gebruikt worden om de toestand van een taak op te volgen tijdens de uitvoering, en eventuele problemen in een vroeg stadium te detecteren. Niettegenstaande de mogelijkheid om met sensoren zijn omgeving waar te nemen, is een robot nog niet in staat de oorzaak van een probleem te interpreteren en mogelijke oplossingen voor te stellen.

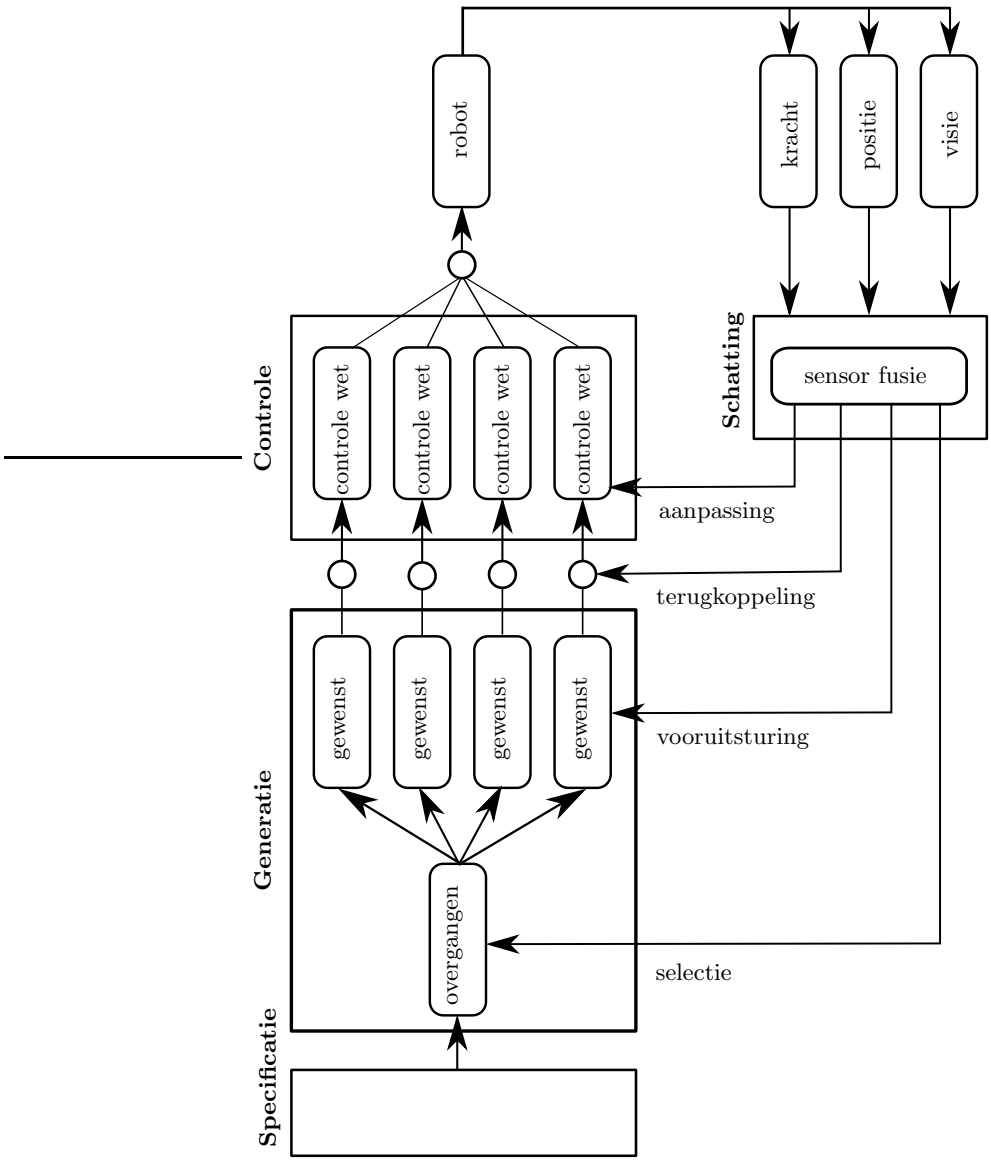
Componenten Het algemene controleschema voor een ACM-systeem bestaat uit vier belangrijke componenten: de taakspecificatie, het genereren van gewenste waarden, de controle en de schatting. Figuur 2 toont de componenten van het controleschema, en de gegevensstroom tussen de verschillende componenten.

- De specificatiecomponent geeft een operator de mogelijkheid om op een eenvoudige manier een taak voor de robot te specificeren. In het ideale geval hoeft de operator slechts weinig details te specificeren, maar volstaat een specificatie zoals bijvoorbeeld *steek pen in gat*.

- De generatiecomponent berekent op elk ogenblik de gewenste waarden voor de beweging van de robot, zoals bijvoorbeeld de gewenste assnelheden. Elke deeltaak heeft een eigen specifieke generatiecomponent.

- De controlecomponent vergelijkt op elk ogenblik de gewenste waarden van de generatiecomponent, met de gemeten waarden van de sensoren, en stuurt de robot bij als deze waarden van elkaar afwijken. Elke deeltaak heeft naast een eigen generatiecomponent ook een eigen controlecomponent.

- De schattingcomponent combineert meteringen van meerdere sensoren om taakspecifieke parameters te schatten. Voorbeelden zijn de kansdichtheid van de discrete parameters van de taak (bijvoorbeeld welke contacten er zijn tussen de gemanipuleerde objecten) en de kansdichtheid van de continue geometrische parameters van de taak (bijvoorbeeld de lengte van een zijde van een vlak).



Figuur 2: Het controleschema voor de uitvoering van actief gecontroleerde, sensorgebaseerde taken in contact.

1.3 Bijdragen van dit proefschrift

Dit proefschrift levert verschillende bijdragen aan de generatiecomponent en de schattingcomponent, getoond in Figuur 2. De bijdragen zijn onderverdeeld in vier categorieën:

- Dit proefschrift stelt een methode voor om de beperkingen van een robotmanipulator te integreren in de contactgrafe. De methode verifieert of elk knooppunt van de grafe, en de overgang tussen elke twee knooppunten van de grafe, kan bereikt worden met een robot. Onbereikbare knooppunten en overgangen worden uit de grafe verwijderd. Hierdoor kan deze grafe gebruikt worden om een contactpad te plannen dat uitvoerbaar is door een robot.

- Dit proefschrift past stochastische methodes toe op programmeren door menselijk voordoen, zoals in de eerder voorgestelde methode voor het gelijktijdig schatten van geometrische parameters en herkennen van contacten, op basis van een deeltjesfilter. De eerder voorgestelde methode kan slechts een beperkt aantal verschillende contacten onderscheiden; de methode die in dit proefschrift wordt voorgesteld kan *alle mogelijke* contacten tussen twee polyhedrische objecten onderscheiden. Om met deze toegenomen complexiteit om te gaan, maakt dit proefschrift gebruik van de *topologische informatie* die vervat zit in een contactgrafe. Verder stelt dit proefschrift ook efficiënte algoritmes voor om de meetmodellen op basis van het sluiten van de positielus en de consistentievergelijkingen te implementeren. Deze algoritmes laten toe de deeltjesfilter in *ware tijd* te gebruiken tijdens de uitvoering van een contacttaak.

- Dit proefschrift stelt ook een methode voor om op een geautomatiseerde manier een contactpadbeschrijving om te zetten in gewenste waarden voor de controlecomponent. De contactpadbeschrijving kan afkomstig zijn van een automatische contactpadplanner, of van de demonstratie van een taak door een mens. De voorgestelde methode laat toe om op basis van de geometrische beschrijving van een contactpad, onmiddellijk de uitvoering van dit pad onder actieve krachtcontrole te realiseren.

- Dit proefschrift maakt gebruik van Bayesiaanse schatters om discrete toestandsveranderingen tijdens de uitvoering van een contact taak te herkennen, en als terugkoppeling aan de controlecomponent te geven. Deze terugkoppeling laat de controlecomponent toe om op elk moment tijdens de uitvoering een aangepaste controlewet te selecteren.

2 Literatuurstudie

2.1 Programmeren van robots

Het programmeren van robots, het specificeren van acties en deeltaken die de robot moet uitvoeren om een taak tot een goed einde te brengen, was altijd al een uitdaging. Sinds de eerste robotprogrammeertalen in de jaren 70, zijn er in de roboticaliteratuur vele methodes voorgesteld om robots te programmeren. De methodes verschillen in de taak waarop ze gericht zijn, in het niveau van detail waarop een taak gespecificeerd wordt, en in hun praktische bruikbaarheid. Een uitgebreid overzicht is te vinden in (Ránky 1985; Latombe 1991; Chen 2001). De methodes kunnen opgedeeld worden in vier grote groepen:

- naspeelmethodes,
- programmeertalen,
- taakniveau programmeren, en
- programmeren door menselijk voordoen.

Tabel 1 toont deze vier methodes, opgedeeld in hoog- en laagniveau specificatie methodes, en specificeer- of demonstreermethodes.

	specificeer taak	**demonstreer taak**
hoog niveau	taakniveau programmeren	menselijk voordoen
laag niveau	programmeertalen	naspeelmethodes

Tabel 1: Methodes voor het specificeren van een contact taak.

Naspeelmethodes Bij het gebruik van naspeelmethodes leidt de operator de robot manueel door een reeks van posities en acties (bijvoorbeeld open/sluit de grijper) om een taak uit te voeren. Door alle instructies van de operator op te nemen, en later weer af te spelen, voert de robot een taak uit. De eerste naspeelmethodes lieten de operator enkel toe om de robot op asniveau te bewegen; latere methodes maakten het mogelijk om de robot in de Cartesische ruimte te bewegen. De robot speelt het resulterende programma af in een open lus, zonder een sensorterugkoppeling over de werkelijke interactie met de omgeving.

Programmeertalen Doorheen de jaren zijn er verschillende programmeertalen ontwikkeld voor het programmeren van robots. Een vergelijkende studie van deze talen is te vinden in (Bonner and Shin 1982; Pembeci and Hager 2002). Een programmeertaal stelt de operator in staat om manueel programmacode te schrijven die de gewenste taak beschrijft aan de hand van commando's zoal *beweeg naar positie*, *sluit grijper*, of complexere specificaties zoal *beweeg in de x-richting, en behoud contact in de y-richting.* Metingen van verschillende sensoren beïnvloeden de uitvoering van een robotprogramma. Het programmeren van robots met behulp van een programmeertaal geeft de operator veel flexibiliteit, maar vergt ook een zekere expertise en is zeer tijdsrovend voor complexere taken.

Taakniveau programmeren Bij het programmeren van een robot op taakniveau, hoeft de operator slechts een aantal hoogniveau commando's te geven zoals *steek* pen in gat, *grijp* dit object, of *puntlas* deze objecten aan elkaar. De operator specificeert *wat* de robot moet doen, maar niet *hoe*. Het onderliggende planningsysteem berekent alle noodzakelijke laagniveau robotcommando's nodig om de taak te realiseren. Hoewel deze manier van robotprogrammeren al veel aandacht kreeg van verschillende onderzoeklabo's, toch is het uiteindelijke doel van een flexibel en eenvoudig systeem voor het programmeren van robots nog niet binnen bereik. Een overzicht van het onderzoek over deze manier van taak specificatie is te vinden in (Latombe 1999; LaValle 2006).

Programmeren door menselijk voordoen Om complexe robot taken te programmeren, maakt de *programmeren door menselijk voordoen* (PbD) methode gebruik van de sterk ontwikkelde menselijke motorische vaardigheden. In PbD wordt een robottaak gespecificeerd door de operator de taak te laten demonstreren. De demonstratie kan gebeuren met behulp van de robot zelf, in een virtuele omgeving, of gebruik makende van een speciale demonstratiehulpmiddel. Tijdens de demonstratie registreren verschillende sensoren de acties van de operator. Na de demonstratie, in de interpretatiefase, worden verschillede taakparameters uit de sensormetingen geschat. PbD laat zelfs niet technische gebruikers toe om een robottaak te specificeren. PbD heeft echter ook een inherente beperkingen: de operator moet acties demonstreren die informatief zijn over de vaardigdelen die in de taak vervat zitten.

2.2 Schatten van geometrische parameters en herkennen van contacten

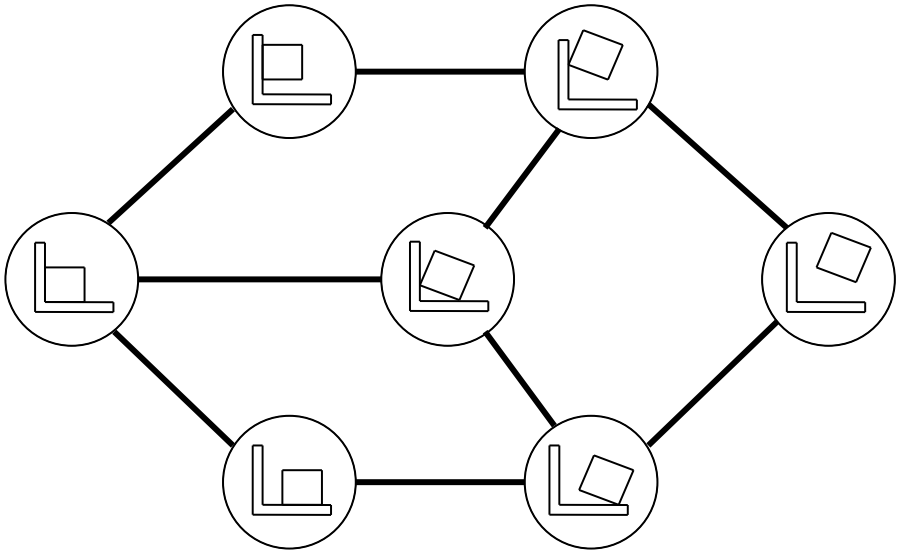
Voor het uitvoeren van taken in contact in een onzekere omgeving, maakt de robot gebruik van verschillende sensoren om zijn omgeving waar te nemen. De meest voorkomende sensor is de krachsensor om de krachtinteractie in de

contacten op te meten, maar ook sensoren zoals een camera of een laserafstandssensor kunnen gebruikt worden. Door de informatie van verschillende sensoren te combineren en te interpreteren, kan een robot gebeurtenissen in zijn omgeving herkennen en hierop gepast op reageren. De voornaamste uitdagingen in de interpretatie en combinatie van sensorgegevens voor contacttaken zijn: (i) de contacten tussen de verschillende objecten herkennen, (ii) de geometrische parameters van de contacten schatten (bijvoorbeeld de positie van een contactpunt, de richting van de contactnormaal, enz.), en (iii) detecteren wanneer er een transitie tussen verschillende contacten optreedt. Verschillende methodes lossen elk van deze deelproblemen afzonderlijk op, gebruik makende van technieken zoals verborgen Markov modellen, neurale netwerken, of toepassingspecifieke ad-hoc technieken. Meer recente methodes proberen de verschillende deelproblemen gelijktijdig op te lossen.

3 Planner voor beweging in contact

3.1 Contactgrafe

Om een taak met twee objecten in contact met succes te voltooien, is de kennis van de mogelijke contactsituaties tussen de objecten en de relatie tussen de onderlinge contactsituaties belangrijk. Daarom maken padplanners voor contacttaken gebruik van de contactgrafe. In de contactgrafe stelt een knooppunt een mogelijke contactsituatie voor, en een link tussen twee knooppunten toont dat er een directe overgang tussen de twee contactsituaties mogelijk is, zonder door een derde contactsituatie te gaan. Figuur 3 toont een vereenvoudigde voorstelling van een contactgrafe. Vaak is de informatie in een contactgrafe manueel ingevoerd, wat een tijdrovende en moeilijke opgave is. Xiao and Ji (2000) ontwikkelden en implementeerden daarom een methode voor het automatisch genereren van een contactgrafe, gebaseerd op enkel de geometrische informatie van de objecten in contact. Deze methode houdt echter geen rekening met een robotmanipulator die de objecten manipuleert; de robotmanipulator beperkt de bewegingsvrijheid van het gemanipuleerde object, waardoor er minder mogelijke contactsituaties zijn, en minder mogelijke overgangen tussen contactsituaties. Dit proefschrift stelt de grote lijnen voor van een methode om de beperkingen van een robot manipulator automatisch toe te voegen aan een bestaande contactgrafe. Deze methode maakt gebruik van een virtuele contactmanipulator, om verschillende mogelijke paden tussen naburige contactsituaties uit te proberen met de manipulator.



FIGUUR 3: Een contactgrafe toon alle mogelijke contactsituaties (knooppunten) en overgangen tussen contactsituaties (verbindingen tussen knooppunten). Deze vereenvoudigde figuur bevat slechts een aantal mogelijke contactsituaties, maar een werkelijke contactgrafe bevat honderden contactsituaties.

3.2 Padplanner

Een padplanner genereert automatisch een pad in contact uit een gegeven startpositie in een startcontactsituatie, en een gegeven eindpositie in een eindcontactsituatie. De planner verbindt dan de start- en eindpositie met een pad dat botsingvrij is voor zowel de robotmanipulator als het gemanipuleerde object. Het pad is gegeven door een reeks van opeenvolgende posities van de objecten in contact, en de reeks van corresponderende contactsituaties. Ji and Xiao (2001a) ontwikkelden een hybride aanpad om dit planningprobleem op te lossen. In een eerste stap maakt de planner gebruik van de contactgrafe om de gegeven begincontactsituatie te verbinden met de gegeven eindcontactsituatie. Dit hoogniveau pad bestaat uit een opeenvolging van naburige contactsituaties die voor zowel de robotmanipulator als voor het gemanipuleerde object mogelijk zijn. In een tweede stap genereert de planner een reeks van relatieve posities tussen de twee objecten om de discrete contactsituaties uit het hoogniveau pad met elkaar te verbinden. Het resultaat is een geometrische voorstelling van een contactpad dat de gegeven startpositie verbindt met de gegeven eindpositie.

4 Programmeren door menselijk voordoen

In tegenstelling tot de methode voorgesteld in de vorige sectie, waar een padplanner automatisch een contactpad genereert, maakt deze sectie gebruik van *programmeren door menselijk voordoen* (PbD) om een contactpad te bekomen. In PbD demonstreert de operator de gewenste contacttaak. Tijdens de demonstratie registreren sensoren verschillende parameters van de contacttaak. In een interpretatiestap worden de opgemeten parameters verwerkt tot een taakbeschrijving van het gedemonstreerde contactpad.

4.1 Demonstratiehulpmiddel

De operator kan een contacttaak demonstreren gebruik makende van de robotmanipulator zelf, van een haptische interface naar een virtuele omgeving, of van een demonstratiehulpmiddel dat op het gemanipuleerde object is gemonteerd. Dit proefschrift maakt gebruik van de voor de mens meest intuïtieve methode, namelijk PbD met een demonstratiehulpmiddel, waarbij de operator het object zelf manipuleert in de werkelijke taakomgeving. Het demonstratiehulpmiddel, zoals getoond in Figuur 4, is uitgerust met een krachtsensor tussen het demonstratiehulpmiddel en het gemanipuleerde object, om de contactkrachten te registreren. Op de zijwanden van het demonstratiehulpmiddel zijn meerdere LEDs bevestigd, tot 4 LEDs per zijde van het demonstratiehulpmiddel. De positie van elke LED wordt opgemeten door een camerasysteem bestaande uit drie camera's. Als de drie camera's gelijktijdig een LED in beeld hebben, kan de positie van de LED in de ruimte bepaald worden, tot op een nauwkeurigheid van 0.1 $[mm]$, aan 100 $[Hz]$. Als 4 of meer LEDs gelijktijdig voor alle drie de camera's zichtbaar zijn, kan uit de positie van de LEDs de positie en oriëntatie van het demonstratiehulpmiddel berekend worden.

4.2 Schatting van geometrische parameters en herkenning van contactsituaties

Tijdens de demonstratie van een contacttaak meten sensoren de contactkrachten tussen de objecten, en de positie en de oriëntatie van het demonstratiehulpmiddel. Bayesiaanse schatters gebruiken deze metingen om enerzijds geometrische parameters van de objecten te schatten, en anderzijds de contactsituatie tussen de objecten te herkennen. Deze twee deelproblemen worden gelijktijdig opgelost omdat ze niet onafhankelijk zijn van elkaar: de schatting van de contacten verbetert als de geometrische parameters van de objecten gekend zijn, en om de geometrische parameters te schatten is de kennis van de huidige contacten noodzakelijk. De gelijktijdige schatting van continue geometrische parameters en discrete contactsituaties, is een hybride (deels

FIGUUR 4: De operator gebruikt een demonstratiehulpmiddel om een contacttaak te demonstreren. Sensoren op het demonstratiehulpmiddel registreren verschillende taakparameters.

continu, deels discreet) probleem. Omdat Kalman filter varianten niet kunnen omgaan met een hybride toestand, maakt dit proefschrift gebruik van een deeltjesfilter. In een deeltjesfilter stelt elk deeltje een mogelijke waarde van de hybride toestand voor; een waarde voor de geometrische parameters, en een waarde voor de contactsituatie. Met voldoende deeltjes is het mogelijk om een goede benadering te maken van elke continue, discrete of hybride kansdichtheidfunctie. Er bestaat echter telkens een afweging tussen het aantal gebruikte deeltjes, en de snelheid van de deeltjefilter.

Na elke sensormeting tijdens de demonstratie past de deeltjesfilter de schatting van de geometrische toestand en de contactsituatie aan, gebruik

makende van een systeem- en een meetmodel. Het systeemmodel maakt voor elk deeltje een voorspelling van de contactsituatie in de volgende tijdsstap, gebaseerd op de contactsituatie van het deeltje en de topologische informatie in de contactgrafe. het systeemmodel past dus de *waarde* aan van elk deeltje. Het meetmodel kent vervolgens een waarschijnlijkheid toe aan elk van de voorspelde deeltjes, en past daarmee het *gewicht* van elk deeltje aan. Voor elke sensormeting is een kansdichtheidfunctie (PDF) gegeven, die aangeeft hoe nauwkeurig deze meting is. Hoewel de keuze van deze PDF deels arbitrair is, is het mogelijk om een fysisch gebaseerde en probleemspecifieke keuze te maken, gebaseerd op (i) de gebruikte sensoren, (ii) de calibratie van de sensoren, en (iii) de gebruikte modellen om de wereld te beschrijven.

5 Contacttaakgenerator

De uitvoer van een contactpadplanner of een programmering door menselijk voordoen, bestaan uit een geometrische beschrijving van een contactpad, in de vorm van een reeks gewenste configuraties, en een corresponderende reeks contactsituaties. Deze geometrische beschrijving bevat onvoldoende informatie, en informatie in de foute vorm, om direct gebruikt te kunnen worden door een hybride controlealgoritme. De *contacttaakgenerator*, een automatische manier om een geometrische beschrijving van een contactpad om te zetten in ogenblikkelijke waarden voor een hybride controlealgoritme, zorgt voor een automatische koppeling tussen een hoogniveau taakspecificatie en een laagniveau controlealgoritme.

Primitieven van de planner De contactpadplanner vertrekt van een gegeven start- en eindconfiguratie en contactsituatie. Gebruik makende van een geometrisch model en de contactgrafe, berekent de padplanner dan een pad dat de begin- en eindconfiguratie verbindt. Het resulterende pad is beschreven aan de hand van enkel geometrische parameters zoals de configuratie en de contactsituatie. De planner genereert geen informatie over de gewenste snelheid, kracht of dynamische interactie.

Primitieven van het menselijk voordoen Bij het programmeren van een taak door menselijk voordoen, demonstreert een operator de gewenste taak. Uit de sensormetingen tijdens de demonstratie worden de taakparameters zoals de geometrie van de objecten en de contactsituatie geschat. De beschrijving van het resulterende contactpad is identiek aan dat van een contactplanner. Daarenboven geeft de demonstratie ook informatie over de gewenste contactkrachten en de gewenste bewegingssnelheid.

Primitieven van de controle Dit proefschrift maakt gebruik van het hybride kracht/snelheid controlealgoritme, dat uitgaat van een eerste orde benadering van de ogenblikkelijke kinematische vrijheidsgraden van een object in contact met zijn omgeving. Hierbij worden de 6 vrijheidsgraden van een object in de ruimte onderverdeeld in s snelheidgecontroleerde vrijheidsgraden, en $(6 - s)$ krachgecontroleerde vrijheidsgraden, voorgesteld door een s-dimensionale snelheiddeelruimte en een $(6 - s)$-dimensionale krachtdeelruimte. In elke deelruimte gebruikt de hybride controle een eigen, onafhankelijke controlelus. De hybride controle verwacht voor op elke tijdstap een gewenste waarde voor elk van de deelruimtes; een gewenste snelheid voor de snelheidgecontroleerde deelruimte en een gewenste kracht voor de krachtgecontroleerde deelruimte. Ook verwacht de hybride controle op elk moment een voorstelling van de deelruimtes. Deze controleprimitieven stellen hetzelfde contactpad voor als de plannerprimitieven of de primitieven van een menselijke demonstratie, maar dan in een vorm die begrijpbaar is voor de controle.

De omzetting van het geometrische pad naar ogenblikkelijke waarden voor de controle gebeurt door een gewenste magnitude voor de snelheid en kracht te specificeren als invoer voor de contacttaakgenerator. Dit maakt het mogelijk om een complexe contacttaak te genereren met een padplanner, of deze taak te demonstreren met een demonstratiehulpmiddel, en deze taak dan onmiddellijk uit te voeren onder actieve krachtcontrole op een echte robotmanipulator.

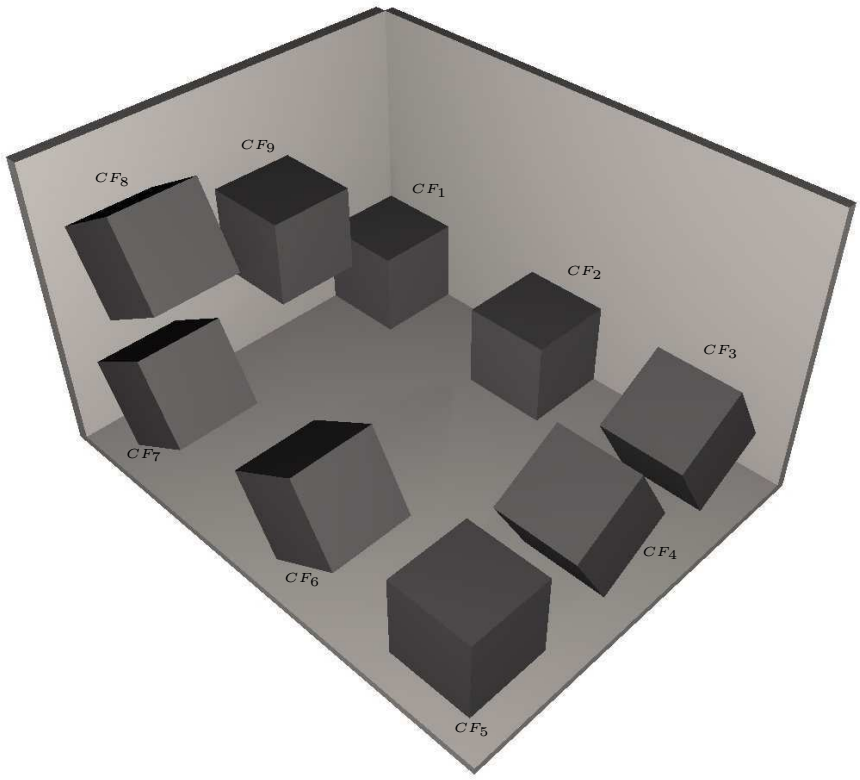
6 Experimenten

Dit proefschrift voert verschillende experimenten uit om de doeltreffendheid en toepasbaarheid van de voorgestelde methodes te verifiëren. De testopstelling bestaat uit een robotmanipulator met zes vrijheidsgraden, die een kubus manipuleert in contact met drie loodrecht op elkaar staande vlakken. De eerste experimenten gebruiken de *Kuka 361*, terwijl de volgende experimenten gebruik maken van zijn grotere broer, de *Kuka 160*. De originele controle over beide robots is overgenomen door een computer uitgerust met dataacquisitiekaarten. Het gebruikte softwareplatform is gebaseerd op het waretijdraamwerk voor robot controle: *Open RObot COntrol Software* (Orocos), en de *Real-Time Application Interface* (RTAI) uitbreiding van de Linux kernel. De robotmanipulator is uitgerust met de zes-componenten *JR3* kracht/moment sensor.

6.1 Contactpad planning

De eerste experimenten verifiëren de methode voor het automatisch omzetten van de geometrische beschrijving van een contactpad, in ogenblikkelijke waarden voor de hybride kracht/snelheid controle. In een eerste stap genereert de

contactpadplanner een geometrische beschrijving van een pad, vertrekkende van een door de gebruiker opgegeven begin- en eindconfiguratie en contactsituatie. De planner maakt gebruik van de contactgrafe, die voor dit voorbeeld bestaat uit 245 knooppunten. Het gegenereerde pad is voorgesteld in Figuur 5.



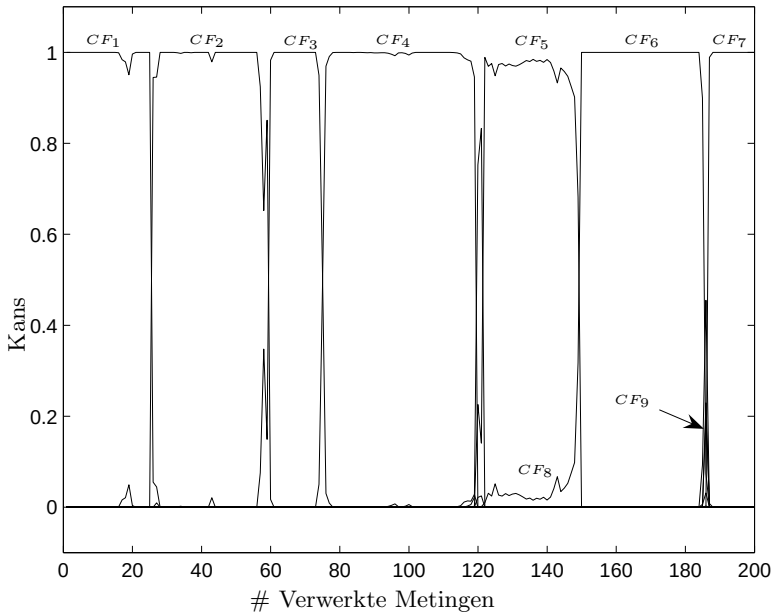
Figuur 5: Het contactpad gegenereerd door de contactplanner bestaat uit een opeenvolging van configuraties en hun overeenkomstige contactsituaties.

In een tweede stap zet de contacttaakgenerator de geometrische beschrijving van het pad, aangevuld met de gewenste magnitudes voor de contactkracht en manipulatorsnelheid, om in ogenblikkelijke waarden voor de hybride controle. Ook berekent de taakgenerator een beschrijving van de locale kracht- en snelheiddeelruimtes die de eerste orde kinematica van de contactsituatie beschrijven. In een laatste stap voert de Kuka 361 het geplande pad uit, onder actieve krachtcontrole. De experimenten tonen aan dat het geplande pad nauwkeurig wordt uitgevoerd door de robot, zelfs in een niet nauwkeurig

gekende omgeving.

6.2 Programmeren door menselijk voordoen

De volgende experimenten verifiëren de methode voor het programmeren van een contact taak door middel van menselijk voordoen. In een eerste stap gebruikt een operator het voorgestelde demonstratiehulpmiddel om een kubus te manipuleren in contact met drie loodrecht op elkaar staande vlakken. Tijdens de demonstratie meet een krachtsensor de contactkrachten en -momenten op. Een camerasysteem meet de positie en oriëntatie van het demonstratiehulpmiddel. In een tweede stap gebruikt een deeltjesfilter deze sensormetingen om de geometrische parameters van de taak te schatten, en om gelijktijdig de contactsituatie op elk moment te herkennen. Het resultaat is getoond in Figuur 6, waar de kans op elke contactsituatie voorgesteld is gedurende het verloop van de demonstratie.



FIGUUR 6: De kans op elke contactsituatie tijdens de demonstratie van een contacttaak door een operator.

6.3 Schatting tijdens de uitvoering

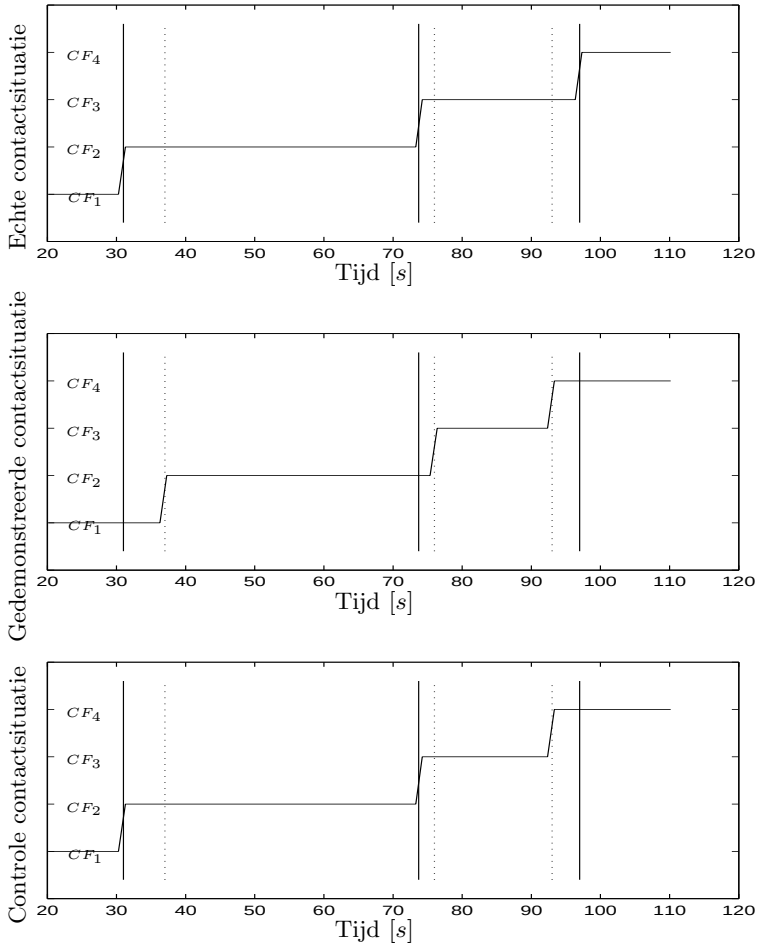
In het eerste voorgestelde experiment zijn de krachtmetingen de enige vorm van terugkoppeling van de omgeving naar de uitvoering. De actieve krachtterugkoppeling maakt de uitvoering meer robuust tegen onnauwkeurigheden in de omgeving, maar laat nog niet toe om toe te zien op de correcte uitvoering van de taak. De uitvoering in dit experiment is wel voorzien van een surveilleringsmechanisme, om toe te zien op de correcte uitvoering van alle contactsituaties. Hiervoor wordt dezelfde deeltjesfilter ingezet als gebruikt tijdens de programmering door menselijk voordoen. Gebruik makende van kracht-, snelheids- en positiemetingen, bepaalt de deeltjesfilter de echte contactsituatie die optreedt op elk moment van de uitvoering. Gebaseerd op deze echte contactsituatie en de gewenste contactsituatie die uit de demonstratie volgde, kan de hybride controle een aangepaste controlewet selecteren, om de taak ook in niet nauwkeurig gekende omgevingen succesvol te kunnen uitvoeren. Figuur 7 toont de echte en de gedemonstreerde contactsituatie tijdens de uitvoering van een contacttaak. Ook toont deze figuur de contactsituatie die de hybride controle gebruikt als controlewet.

7 Algemeen besluit

7.1 Situering

Dit proefschrift zet een stap in de richting naar meer intelligente, autonome en flexibele robots. Met sensoren neemt de robot zijn omgeving waar, en reageert op gebeurtenissen in de omgeving, waardoor er een interactie ontstaat tussen de robot en de omgeving. Dit proefschrift richt zich voornamelijk op contacttaken, waarbij de robotmanipulator een object manipuleert in contact met de omgeving; hiervoor is de robot uitgerust met sensoren om de contactkrachten, de manipulatorsnelheid en de manipulatorpositie op te meten. Elke sensor geeft informatie over een deel van de taakparameters, en verhoogt zo de mogelijkheden van de robot om te reageren op gebeurtenissen in zijn omgeving. Dit proefschrift probeert drie doelen te bereiken voor contacttaken:

- het toegankelijker maken en vereenvoudigen van de taakspecificatie van een contacttaak, door hoogniveau taakspecificatiemethodes te integreren in het algemene controleschema voor contacttaken,
- het verbeteren van de informatieextractie uit de ruwe data afkomstig van meerdere heterogene sensoren, door Bayesiaanse methodes te combineren met de topologische informatie in een contactgrafe,
- de robuustheid van de uitvoering van contacttaken verhogen door de contactsituaties op te volgen tijdens de uitvoering, en deze informatie



FIGUUR 7: De echte, gedemonstreerde en controle contactsituatie, tijdens de krachtgecontroleerde uitvoering van een contacttaak. De echte overgangen tussen contactsituaties van CF_1 naar CF_2 naar CF_3 gebeuren vroeger dan gedemonstreerd, terwijl de echte overgang van CF_3 naar CF_4 later gebeurd dan gedemonstreerd.

door te spelen aan de controle voor de selectie van een optimale controlewet.

7.2 Bijdragen

Deze sectie beschrijft kort de belangrijkste bijdragen van dit profschrift, opgedeeld in vier onderdelen:

Manipulatorbeperkingen in de contactgrafe Een contactgrafe stelt alle mogelijke contactsituaties tussen twee polyhedrische objecten voor als knooppunten, en alle mogelijke overgangen tussen naburige contactsituaties als verbindingen tussen knooppunten. Dit proefschrift gebruikt het "doel contact verbreking" algoritme om automatisch een volledige contactgrafe te genereren. Dit algoritme gaat er echter van uit dat beide objecten vrij kunnen bewegen in de ruimte. In een robottaak is het echter een robotmanipulator die een van de objecten manipuleert, waardoor dit object beperkt is in zijn bewegingen door de bewegingvrijheidsgraden van de robot. Daardoor kan een automatisch gegenereerde contactgrafe niet onmiddellijk gebruikt worden in een contactplanner.

Dit proefschrift stelt een methode voor om de beperkingen van een manipulator te integreren in de contactgrafe. De methode verifieert elk knooppunt van een gegeven contactgrafe, en zoekt een mogelijke configuratie van de twee objecten die bereikbaar is voor de robotmanipulator. Als binnen de contactsituatie van een knooppunt een dergelijke configuratie niet bestaat, wordt dit knooppunt uit de contactgrafe verwijdert. Vervolgens verifieert de methode alle verbindingen tussen de overblijvende knooppunten, gebruik makende van een virtuele contactmanipulator. Deze stuurt een gesimuleerd kinematisch model van een robot aan, en zoekt zo een pad in contact van het ene knooppunt naar het andere knooppunt. Als er geen pad gevonden wordt tussen de knooppunten, verdwijnt dit uit de grafe. Het uiteindelijke resultaat is een contactgrafe waarin alle contactsituaties en overgangen tussen contactsituaties mogelijk zijn voor een robotmanipulator.

Schatting in programmeren door menselijk voordoen Bij het programmeren van een contacttaak door menselijk voordoen demonstreert de operator de gewenste taak gebruik makende van een demonstratiehulpmiddel. Tijdens de demonstratie registreert het Krypton 6D camera systeem de positie, oriëntatie en snelheid van het demonstratiehulpmiddel, en de JR3 kracht/moment sensor registreert de krachtinteractie tussen de objecten in contact. Vervolgens, in een interpretatiestap, worden Bayesiaanse technieken gebuikt voor de interpretatie van de sensorsignalen.

Dit proefschrift stelt een methode voor op basis van een deeltjesfilter voor de gelijktijdige schatting continue geometrische parameters en de discrete contactsituatie. De deeltjesfilter geniet de voorkeur voor dit hybride (deels continue en deels discrete) en sterk nietlineaire probleem. Eerder voorgestelde methodes gebaseerd op deeltjesfilters waren beperkt tot enkele mogelijke contactsituaties tussen de objecten in contact. Dit proefschrift breidt de methode uit tot *alle mogelijke contactsituaties* tussen de objecten (245 mogelijke contactsituaties in de voorgestelde experimenten), door gebruik te maken van de *topologische informatie* in de contactgrafe. Deze uitbreiding, in combinatie met nieuwe en efficiënte algoritmes, maken het mogelijk om in *ware tijd* geometrische parameters te schatten en tegelijkertijd contactsituaties te herkennen uit honderden mogelijke contactsituaties.

Contacttaakgenerator Een automatische contactpadplanner, of een programmatie door menselijk voordoen, resulteert in een geometrische beschrijving van het pad in contact. De beschrijving bestaat uit een reeks configuraties tussen de twee objecten, en de overeenkomstige contactsituaties. Deze beschrijving van de taak is echter niet onmiddellijk bruikbaar door een hybride controle, die op elk tijdstip een gewenste kracht en snelheid verwacht, samen met de kracht- en snelheidgecontroleerde deelruimtes. Om een geometrisch padbeschrijving te kunnen uitvoeren, moet deze omgezet worden in de ogenblikkelijke controleprimitieven.

Dit proefschrift stelt de *contacttaakgenerator* voor, de eerste algemene en automatische methode die planning en demonstratie verbindt met krachtgecontroleerde uitvoering. De methode is toepasbaar op contact taken tussen onvervormbare, polyhedrische objecten, en is algemener dan eerder voorgestelde ad-hoc of regelgebaseerde methodes. Daarenboven is de contacttaakgenerator invariant ten opzichte van veranderingen van referentieassenkruis, scalering en verandering van fysische eenheden. De omzetting van de geometrische beschrijving naar ogenblikkelijke waarden voor de controle, wordt apart verwerkt voor de snelheid- en de krachtdeelruimtes. De operator specificeert de gewenste magnitudes voor de snelheid en de kracht, waaruit de contacttaakgenerator dan de gewenste waarden voor de controle berekent. Het resultaat is dat een complexe contact taak met meerdere gelijktijdige contactsituaties en verschillende overgangen tussen contactsituaties, kan gepland of gedemonstreerd worden, en vervolgens onmiddellijk kan uitgevoerd worden op een robotmanipulator, onder actieve krachtcontrole.

Contactsituatieherkenning tijdens de uitvoering Een algemene contacttaak bevat een aantal continue bewegingen in contact, en een aantal discrete veranderingen tussen verschillende contactsituaties. Tijdens de uitvoering van een beweging in een bepaalde contactsituatie veranderen de contact-

beperkingen op een continue manier, terwijl een verandering in contactsituatie een discrete verandering van de contactbeperkingen met zich meebrengt. De controlewet in de hybride controle past zich gelijktijdig met de contactbeperkingen aan. Tijdens de uitvoering van een contacttaak moet de controlewet dus aangepast worden aan de contactsituatie die werkelijk optreedt. Hiervoor is het noodzakelijk dat de werkelijke contactsituatie op elk moment gekend is.

Dit profschrift gebruikt een deeltjesfilter om tijdens de uitvoering van een contacttaak de werkelijke contactsituatie te herkennen. De algoritmes van de deeltjesfilter is voldoende efficiënt om de deeltjesfilter in ware tijd uit te voeren tijdens de demonstratie. De schatting van de meest waarschijnlijke contactsituatie wordt aan de controlecomponent doorgegeven, die dan op zijn beurt een optimale controlewet kan selecteren. Dit vergroot de robuustheid van de taakuitvoering, en maakt het mogelijk om fouten te detecteren tijdens de uitvoering.

7.3 Beperkingen en toekomstig onderzoek

Ondanks zijn bijdragen zet dit proefschrift maar een kleine stap in de richting van een volledig autonoom en intelligent systeem voor het uitvoeren van taken in contact, zoals getoond in Figuur 2. Deze sectie bespreekt de beperkingen van het huidige systeem en geeft suggesties voor toekomstig onderzoek.

Herstellen van een fout door herplanning Dit proefschrift gebruikt een deeltjesfilter om tijdens de uitvoering van een taak de werkelijke contactsituatie te herkennen. De kennis van de werkelijkecontact situatie wordt gebruikt door (i) de controle om een aangepaste controlewet te selecteren, en (ii) om eventuele fouten tijdens de uitvoering te *herkennen*. Een volgende logische stap is het niet enkel herkennen van een fout tijdens de uitvoering, maar deze fout ook te *herstellen*. Met de kennis van de werkelijke contactsituatie en de gewenste contactsituatie kan de contactpadplanner een nieuw pad berekenen om vanuit een ongewenste contactsituatie terug naar de gewenste contactsituatie te bewegen; na de herstelling van de fout kan de uitvoering van de taak hervat worden.

Andere sensoren en actief waarnemen Zelfs de verschillende sensoren (kracht, positie en snelheid) die tijdens de uitvoering ingezet worden, kunnen niet op elk moment informatie geven over alle taakparameters. Een eerste oplossing voor dit probleem ligt in het inzetten van nieuwe sensoren, zoals bijvoorbeeld een camera of een laser afstandssensor. De tot nu toe gebruikte positiemetingen geven enkel locale informatie; een camera kan een beter algemeen overzicht van de taak geven en de onderlinge relatie tussen verschillende deelobjecten die waargenomen werden met andere sensoren met elkaar ver-

binden. Een andere oplossing voor de beperkte informatie van sensoren komt van het actief waarnemen; het pad van de robot kan tijdens de uitvoering aangepast worden om zo meer informatie uit de sensormetingen te halen. Hoewel een algemene methode voor het actief waarnemen van bepaalde parameters nog niet direct binnen het bereik ligt, is het wel mogelijk om robotbewegingen te genereren die het mogelijk maken om het onderscheid te maken tussen twee verschillende contactsituaties. Door een beweging aan te leggen die in de krachtdeelruimte van een eerste contactsituatie ligt en gelijktijdig in de snelheiddeelruimte van de tweede contactsituatie ligt, is het mogelijk om aan de hand van de resulterende sensormetingen een onderscheid te maken tussen de twee contactsituaties.

Schatten van continue parameters tijdens uitvoering De deeltjesfilter die in dit proefschrift ingezet wordt tijdens de uitvoering van een contacttaak, herkent de contactsituatie en schat tegelijkertijd de geometrische parameters van de taak. De schatting van de geometrische parameters verbetert nu al de herkenning van de contactsituatie, maar verbetert de parameters van het geometrisch model in de controle- en generatiecomponent nog niet. Een nauwkeuriger model in de generator- en controlecomponent kan leiden tot een verbeterede performantie van het hele systeem.

Taakprimitieven voor beweging in contact Onderzoek in contacttaken richtte zicht tot nu toe vooral op een grotere robuustheid tegen grote geometrische onzekerheden. De robuustheid tegenover nietgemodelleerde effecten zoals wrijving of vervorming is minder groot, en vaak ook zeer moeilijk te modelleren. In plaats van te investeren in betere modellen, is het ook mogelijk om de taakprimitieven voor de beweging in contact aan te passen; de gebruikte taakprimitieven beperking zich nu tot het *glijden* van een object langs het oppervlak van een ander object. Nieuwe taakprimitieven kunnen het glijden vervangen door bijvoorbeeld *springen* of *tasten*. Deze taakprimitieven vermijden het probleem van wrijving en vervorming, wat het mogelijk kan maken om contacttaken te automatiseren in nieuwe domeinen zoals bijvoorbeeld in de bouwsector.

Algemene vormen De vorm van de objecten in dit proefschrift is beperkt tot polyhedrische vormen, wat het praktisch gebruik beperkt tot aan klein aantal applicaties. Door de modellen uit te breiden met een aantal basisvormen die vaak gebruikt worden in CAD systemen, zoals cilinders of bollen, kunnen een groot aantal praktische problemen geautomatiseerd worden. Huidig onderzoek in contacttaken werkt ook aan contactbeschrijvingen en contactgrafen van objecten met een arbitraire vorm.