

An Object Representation and Methods for Uncertainty-Aware Shape Estimation and Grasping

Von der Fakultät 5: Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde eines Doktors der
Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Stanimir Svetoslav Dragiev

aus Plovdiv

Hauptberichter: Prof. Dr. Marc Toussaint

Mitberichter: Dr. Michael Gienger

Mitberichter: Prof. Dr. Jochen Steil

Tag der mündlichen Prüfung: 18. August 2014

Institut für Parallele und Verteilte Systeme der Universität Stuttgart

2014

In memory of my grandfather who taught me to love knowledge.

In memory of my friend Nikolay Bukoreshtliev who was savouring science.

*Напред! Науката е слънце,
което във душите греј!*

— Стоян Михайловски, *Кирил и Методий*.
Български всеучилищен химн (1892)

The real University is a state of mind. It is that great heritage of rational thought that has been brought down to us through the centuries and which does not exist at any specific location. [...] The real University is nothing less than the continuing body of reason itself.

— Robert M. Pirsig, *Zen and the Art of Motorcycle
Maintenance: An Inquiry into Values* (1974)

Wind extinguishes a candle and energizes fire.

Likewise with randomness, uncertainty, chaos: you want to use them, not hide from them. You want to be the fire and wish for the wind.

— Nassim Nicholas Taleb, Prologue to *Antifragile:
Things That Gain From Disorder* (2012)

Contents

Acknowledgements	xxiii
Summary	xxvii
Prologue	1
1 Introduction	3
1.1 Intelligence	3
1.2 Intelligent between easy and hard	4
1.2.1 Hard made easy	5
1.3 Grasping	7
1.4 Motivating scenario	10
1.5 Novel representation and methods	12
1.5.1 Representing is getting the right features	13
1.5.2 Representation: a learnable, affording grasp shape	14
1.5.3 Methods for estimation and control	15
1.6 Agenda, structure, contributions	15
2 Background	19
2.1 Problems and domains	19
2.1.1 Shape estimation	20
2.1.2 Robotic grasping and manipulation	21

2.1.3	Collision avoidance for grasping	23
2.1.4	Object representations	23
2.1.5	Uncertainty	27
2.1.6	Statistical learning	28
2.2	Methods	30
2.2.1	Forward and inverse kinematics	31
2.2.1.1	Forward	32
2.2.1.2	Inverse	32
2.2.2	Control under multiple objectives	35
2.2.3	Gaussian process	38
2.2.4	Implicit surface	41
3	Gaussian process implicit surface potential	45
3.1	Representations	45
3.2	Related work: geometric shape representations for grasping	48
3.2.1	Fitting and composing primitives	48
3.2.2	Polyhedrons and patches	50
3.2.3	Visual and sub-shape representations	52
3.2.4	Implicit surfaces	53
3.2.5	Summary	54
3.3	Requirements	55
3.4	Glueing GP, IS and P to GPISP	57
3.4.1	Gaussian Process	57
3.4.2	Implicit surface	58
3.4.3	Potential	61
3.4.4	Requirements revisited	62
3.5	Parameters	63
3.5.1	External and internal parameters	63
3.5.1.1	Covariance width	64
3.5.1.2	Observation noise	65

3.5.1.3	Bias and prior variance	65
3.5.1.4	Derivative observations	67
3.5.2	The resulting implicit shape prior	67
3.6	Summary	69
4	Shape estimation	71
4.1	Related work: shape estimation approaches	71
4.1.1	From surface points to superquadrics	72
4.1.2	Volumetric occupancy map	72
4.1.3	Polyhedron walls from tactile points and normals	73
4.1.4	Fusion of heterogeneous datasets	74
4.1.5	Approximating implicit surfaces	75
4.1.6	Visual representations	76
4.1.7	Summary	77
4.2	Shape estimation with GPISP	78
4.3	Data (points and normals)	79
4.3.1	Surface point observation	80
4.3.2	Precision	81
4.3.3	Surface normal observation	81
4.4	Fusion of multi-modal data	82
4.4.1	Preprocessing	82
4.4.2	Combining sources	83
4.5	Performance	83
4.6	Visualisation	86
4.7	Noise distributions	88
4.8	Experiment: estimation of synthetic surfaces	89
4.9	Summary	91
5	Grasp synthesis	93
5.1	Related work: Approaches to grasping	93

5.1.1	Posture	94
5.1.1.1	Grasping with stability measures	95
5.1.1.2	Grasping with visual cues	98
5.1.1.3	Matching the shape	99
5.1.2	Collision avoidance, path finding and planning	101
5.1.2.1	Obstacle avoidance on demand	101
5.1.2.2	Obstacle avoidance by sampling and planning	102
5.1.2.3	Grid search and optimality	103
5.1.2.4	Potential fields for navigation	104
5.1.2.5	Mixing trajectory objectives	107
5.1.2.6	Imitation of trajectories	108
5.1.2.7	Optimal timing	109
5.1.3	Summary	110
5.2	Collision avoidance for manipulation trajectories	111
5.3	Potential field grasping	114
5.3.1	Potential properties	114
5.3.2	Heuristics for feasible grasps	116
5.3.3	General approach	116
5.3.4	Task variables for reach and grasp	117
5.3.4.1	Task variables with GPISP	118
5.3.4.2	Avoidance of collisions and joint limits	120
5.3.5	Objective function	121
5.3.6	Relation to grasp taxonomies	122
5.3.7	Relation to synergies	124
5.3.8	Potential affordance	125
5.3.9	On feed back and feed forward	126
5.4	Potential field feedback controller	127
5.5	Potential field trajectory optimisation	130
5.5.1	Reach and grasp from potential	130
5.5.2	Reach from movement costs, grasp from potential	131

5.5.3	Optimisation heuristics	134
5.6	Validation	135
5.7	Summary	139
5.7.1	Discussion	140
5.7.1.1	Hierarchies, planning	140
5.7.1.2	Learning	140
5.7.1.3	Relation to closure arguments	141
5.7.1.4	Control dependent collision avoidance	141
6	Uncertainty-aware control	143
6.1	Related work: acting under uncertainty and using it	145
6.1.1	Probability, uncertainty	145
6.1.2	Under uncertainty	147
6.1.3	Active learning	150
6.1.4	Summary	155
6.2	Uncertainty-aware grasping	156
6.2.1	From sensor uncertainty to model uncertainty	156
6.2.2	Uncertainty-aware grasp control	158
6.2.2.1	Model variance as motion feature	158
6.2.2.2	Control laws	160
6.3	Demonstration: Variance aware particles	162
6.4	Exploration and exploitation	162
6.5	Model quality measures	165
6.6	On the notions of attempt optimality	167
6.6.1	Constrained exploration	168
6.6.2	The grasping bandit: maximising <i>utility</i>	168
6.6.3	The <i>worth</i> of an explore-grasp	170
6.6.4	Information, learning, interpretation	171
6.7	Summary	173

7	Integrated demonstration systems	175
7.1	Explorative grasp sequence with trajectory planning	176
7.1.1	Grasping sequence	176
7.1.2	Trajectory optimisation	177
7.1.3	Tasks	178
7.1.4	An experiment in simulation	178
7.1.4.1	Measurements	182
7.1.4.2	Results	184
7.2	Estimation of non-convex object with GPISP	185
7.3	Feedback scheme for non-convex object estimation	187
7.3.1	Setup	187
7.3.2	Feedback control	188
7.3.3	Notes on collision avoidance	189
7.3.4	Demonstration	191
7.4	Kinesthetically moved arm with real feedback	192
7.5	Summary	195
8	Conclusions	197
8.1	Summary	197
8.2	Future research suggestions	199
8.2.1	Representations business	199
8.2.2	Tactile array patterns	200
8.2.3	Object recognition as active learning and model selection	201
8.2.4	Transfer to real robots, alternative scenarios	201
8.2.5	Learning surface priors	202
8.2.6	Mixtures of GPs for shape estimation	202
8.2.7	No information left behind	203
A	Derivatives related to GPISP	205
A.1	Basic shape derivatives	205

A.1.1	Gradient	206
A.1.2	Hessian	206
A.1.3	Sphere derivatives	208
A.1.4	Infinite cylinder	208
A.1.5	Finite cylinder	211
A.1.5.1	Between cutting planes and outside the tube	211
A.1.5.2	Between cutting planes and inside the tube and closer to the side than to a lid	212
A.1.5.3	Between cutting planes and closer to a lid than to the side	212
A.1.5.4	Inside tube extension (outside cutting planes)	212
A.1.5.5	Outside extension, outside cutting planes . . .	213
A.2	Gaussian process derivatives	216
A.2.1	Covariances	216
A.2.2	Observations, Gramian, covariance vector	216
A.2.3	Gradient	217
A.2.4	Hessian	217
A.2.5	Definition of covariances	218
A.3	Jacobian of potential field align task variable	222
B	On robot’s self according to Avicenna	225
B.1	Floating man	225
B.2	Floating robot	226
B.2.1	Life in sensory darkness	226
B.2.2	Life as passive witness	227
B.2.3	Life with eternal prior	227
B.2.4	Sensory motor control	228
B.2.5	Abstract actuation	228
	Literature	229

List of Figures

1.1	It is much nicer to walk an uncrumpled landscape.	5
1.2	linearly separable classes	6
2.1	Representations of a bunny for different purposes	24
2.2	Intuition for GP as infinite-variate Gaussian	40
2.3	Blending implicit surfaces	41
2.4	Implicit surface examples	42
3.1	Superquadrics	49
3.2	Discrete shape representations	54
3.3	Implicit surface	54
3.4	Many functions represent the same 1D object implicitly	59
3.5	Implicit surface examples in 1D and 2D	60
3.7	Effects of prior variance choice	66
3.8	Random 2D GPs with varying covariance width	68
4.1	Example GPISP in 1D with two observations	78
4.2	Example GPISP in 2D with 4 observations	79
4.3	Sketch of tactile sensor touching object	81
4.4	Lazy evaluation of a GP	85
4.5	Surface visualisation with mesh vs. surfels	88
4.6	Belief progress for 3D random object estimation	89

4.7	Estimation accuracy for random GPISP shapes	90
5.1	The bug algorithm	102
5.2	Two RRTs growing in a plane	103
5.3	Repulsive and attractive potentials in the plane	106
5.4	Typical collision pairs for grasping	112
5.5	Power and precision grasp in terms of GPISP and grasp center	123
5.6	Two stage optimisation	133
5.7	Experimental setup: arm, camera, table top with objects	136
5.8	Grasping real objects on a table top	137
6.1	Color-coded variance on the surface of a 3D GPISP object . . .	158
6.2	Normal and tangential component of the variance gradient of GPISP	161
6.3	Particles flying to certain areas of GPISP surface	163
7.1	Examples of objects and positions on the table	180
7.2	Sequence of explorative grasps with improving belief (in sim- ulation)	182
7.3	Statistics collected for single grasp sequence (the one in Fig- ure. 7.2)	183
7.4	Estimation error and variance measured in simulation experi- ment	184
7.5	Estimation of a T-shaped object	186
7.6	Estimation sequencer state automaton	188
7.7	Barret WAM and valve handle in simulation.	191
7.8	Estimates of a valve handle in simulation	192
7.9	Evolving belief for a non-convex object in simulation	193
7.10	Barret WAM and valve setup	194
7.11	Estimates of a valve handle from sensing with kinesthetic control	195

B.1 The life loop of a robot with and without sensors or/and actu-
ators. 226

List of Algorithms

1	Grasping with ISF.	129
2	Reactive grasping	177
3	Variance feedback control for estimation	190
4	compute Hessian	218
5	compute Hessian	219

Acronyms

AI artificial intelligence

AICO approximate inference control

DMP dynamic motion primitive

ECVD early cognitive visual descriptors

EM expectation-maximisation (algorithm)

FOL first-order logic

GP Gaussian process

GPISP Gaussian process implicit surface/shape potential

GP-UCB Gaussian process upper confidence bound

IS implicit surface

ISF implicit surface function

ISP implicit surface potential

LQG linear-quadratic-Gaussian (control problem)

LWA (Schunk) light-weight arm

MAP maximum a posteriori (estimate, estimation)

MC marching cubes

MDP Markov decision process

ML machine learning

NN (artificial) neural network

PCA principal component analysis

PCL point cloud library

PID proportional-integral-derivative (controller)

POMDP partially observable Markov decision process

PRM probabilistic roadmap (planning algorithm)

RBF radial basis function

RHS right-hand side

RL reinforcement learning

RRT rapidly exploring random tree

SDH Schunk dexterous hand

SVM support vector machine

SVR support vector regression

TV task variable

UCB upper confidence bound

Acknowledgements

‘It takes a village to raise a child,’ I remember a character saying recently in an ethnographic movie with non-professional actors about the life in a Zapotec village in Mexico. Indeed. And it takes many people to *be*. Often it goes unnoticed when someone *gives*.

I would like to thank Honda Research Institute Europe for funding my PhD project and CoR-Lab, Universität Bielefeld, whose structure I was embedded in, for the great organisation.

Considering research, in the order of appearance, I am grateful to Klaus-Robert Müller for infecting me with Machine Learning and encouraging me to indeed do what I wanted – learn more – and eventually connecting me to Marc Toussaint. Marc’s dedication to research, his fearlessness to dive into any kind of problems and formalise them apart are remarkable. It was pleasure to witness and be part of such insightful moments. Some particles of these hopefully jumped over and will stick to me. I appreciate not only the funding by HRI, which gave me the opportunity to work on this thesis, but also my regular visits at HRI in Michael Gienger’s group. Michael’s enthusiasm for robotics is stirring and motivating. I very much hope I have learned from his calm, consequent and systematic approach. Thank you!

Most research discussions I had casually with my spatially nearest colleagues, again in the order of appearance: Nikolay Jetchev, Nils Plath, Tobias Lang, Katja Hansen, Dmitry Zarubin, Marion Lange, Johannes Kulick, Robert

Lieck. You were so inspiring! Apart from the named, same goes to all other people related at different times and places to Marc's MLR lab, Klaus' IDA, Michael's colleagues at HRI, and Oliver Brock's RBO lab (so generous to give us shelter, Oliver). Thanks for the wonderful atmosphere!

My family has believed unconditionally in my every step since I can remember back. This feels like an infinitely renewable source of confidence. Thank you! Мама, Баба, Раденце, благодаря ви, че винаги ме подкрепяте във всичко, което предприемам, и ме зареждате с увереност без край. My Nadja provides me with the everyday confidence and life support. When I am down she kicks me up and when I am heading outer space she is pulling me safe back to Earth. Having someone to believe in what you believe is a treasure. Съкровище, благодаря, че вярваш в измишльотините, в които вярвам аз!

Adding more to the village that raised me: beyond, complementary, and necessary condition for research are my friends. Through them I experience life in most wonderful facets: thank you for Plovdiv's ease (винаги се връщам в Пловдив...), for the clarity of Bulgarian mountains, for chilling at Brandenburg's waters, for himmelstürming, for luftpumping, for pre-smart-world canteen talks, for all forgotten and unforgettable discussions how to save the world!

Det jeht ooch noch rein: Danke, Berlin!

Zusammenfassung

Künstliche und natürliche Intelligenz können wir Menschen besser verstehen, wenn wir versuchen sie nachzubilden, d.h. von uns gebaute Systeme mit ihr zu versehen. Somit spielt Robotik für die Intelligenzforschung eine zentrale Rolle als das Versuchsfeld, auf dem wir unsere Theorien ausprobieren und Erkenntnisse gewinnen. Intelligenz fängt nicht erst beim Schachspielen oder anderen hohen kognitiven Fähigkeiten an, sondern bereits bei alltäglichen, unbewussten, banalen Aktivitäten, wie dem Laufen oder dem Umgang mit der uns umgebenden physikalischen Welt.

Diese Dissertation beschäftigt sich mit dem Greifen von Gegenständen im Kontext der Robotik. Es wird eine neuartige Beschreibung für Gegenstände entwickelt. Dazu gehören auch Methoden, die diese Repräsentation benutzen, um die Form von Gegenständen zu schätzen und sie zu greifen. Anstatt explizit die materielle Erscheinung eines Gegenstandes zu beschreiben, wird er als eine Funktion dargestellt, die seine Form implizit beschreibt. Die Funktion wird durch Bayesianische statistische Methoden von einzelnen Oberflächenpunkten approximiert. Dadurch entsteht eine Verteilung über die möglichen Formen. Die wahrscheinlichste Form wird benutzt, um anhand einfacher Heuristiken den Roboter zu einem erfolgreichen Griff zu bewegen.

In dieser Repräsentation lassen sich die Messwerte verschiedener unpräziser Sensoren mit etwaigem a priori Wissen fusionieren. Die Beschreibung ist nicht nur eng an das sensorische System gekoppelt, sondern die Interpre-

tation der geschätzten Funktion eignet sich auch zum Erzeugen von Greif- und Manipulationsbewegungen. Darüber hinaus wird die Unsicherheit der Sensoren in eine Abschätzung der Verlässlichkeit des Modells übersetzt. Ein Regelungssystem kann sie benutzen, um das Greifen oder das Erkundigen eines Gegenstandes sicherer oder effizienter zu gestalten. Die Berücksichtigung von Unsicherheit in der Bewegungserzeugung stellt eine Verbindung zu der Exploration-Exploitation-Problematik der Entscheidungstheorie her.

Schließlich werden die einzelnen Methoden in simulierte und reale Systeme integriert, um in Experimenten und Demonstrationen die Eigenschaften der Repräsentation zu illustrieren und zu quantifizieren.

Wichtige Überzeugungen und Erkenntnisse, die dieser Arbeit zugrunde liegen, bzw. die sie vermittelt, sind: Sensorik und Motorik gehören zusammen und sollen voneinander profitieren; Die Einschränkungen, gegeben durch die Fähigkeiten eines Agenten und durch seine Umgebung, sind von großer Bedeutung – als Hinweis, was in der Welt (un)wichtig zum Lernen ist; Eine Repräsentation soll Unsicherheiten berücksichtigen und sie darstellen; Die quantifizierte Unsicherheit soll in die Bewegungserzeugung einfließen; Eine sinnvolle Umsetzung in der realen Welt erfordert auch Hardware, die inhärent mit Unsicherheiten umgeht.

Summary

One of the keys to understanding intelligence is the experience of reproducing it, building it into systems we create. Robotics is the natural ground to implement, test, evaluate and realise concepts. It has already taught us that intelligence is not solely a matter of high cognition, but implies understanding of seemingly trivial everyday skills like walking, sentiment detection and interaction with the physical world.

This thesis introduces an internal object representation for the purpose of robotic manipulation. It abstracts the physical appearance of objects and rather considers a function which describes the surface implicitly. The developed methods for building such models employ Bayesian statistical approaches to fuse the information sources – different sensors and a priori knowledge – and estimate the form of an object being aware of the uncertainties.

The functions have such a shape that can be interpreted as potential field generated by the object. A controller uses this to navigate a robot arm for grasping and manipulation. Since the representation translates the uncertainty of the sensors into confidence of the model, an improved controller is able to employ this in order to achieve more robust grasping or more efficient estimation of an object. This links to exploration-exploitation notions related to decision theory.

Finally, the grasp and estimation methods are integrated to systems used to demonstrate or quantify in simulated and real environments the benefits

and limitations of the representation.

The key beliefs and insights the thesis builds on and attempts to convey are that sensing and control must benefit from each other; embodiment – own and environmental limitations – are important for learning; object models need to be aware of uncertainty and expose it; uncertainty is motion feature – must be used to improve control; the real world eventually requires uncertainty-aware hardware.

Prologue

Many researchers share the passion of revealing the mechanisms which govern our thinking. There are plenty of questions to ask. What makes us intelligent? What is intelligence to begin with? Why and when did we develop the skills we have? Probably, questions like these are so appealing because the subject of interest is so much intimately related to the researcher, to its notion of self, like no other. These questions suggest a lot of introspection: we need to reason about ourselves; the reasoning machine needs to reason about the mechanisms and principles of its machinery. The present thesis will not even remotely answer the above questions. But they have been the most reliable internal source of motivation for the time working on it – especially when the motivation was desperately needed. The relation between these questions and the title of the thesis is possibly not obvious. The task of the introductory chapter will be to unveil the connections.

Chapter 1

Introduction

This chapter first discusses some aspects of intelligent systems. The accent is on sensory-motor intelligence. Then we introduce a specific scenario for robot grasping and outline our general approach.

1.1 Intelligence

There are various traits that make a system intelligent. More often than not, *intelligence* has been attributed to a system which meets only one or few aspects – being e.g. autonomous in some respect or responsive to the environment or able to learn or mine data. The overuse has made of *intelligent*, in the strict sense, a trivial category that does not provide useful separation – either due to applying to nearly everything, or due to having infinite requirements. (The latter is the case when one judges something *intelligent* if they believe it goes beyond current technology limits. Such view clearly varies across subjects, and it shifts with the frontiers of technology.) In spite of the ambiguity of the predicate, the colloquial (informal) use carries a common-sense meaning, mostly clear in the context. In these terms, looking at the clauses in the following paragraph, most readers would agree with the first ones and probably

some would disagree with the last ones – if they were read isolated. However, reading in a row, the statements clearly relate to each other by being similar or providing basis to the previous statement. This would probably convince the skeptic that intelligence is not exclusive right to higher cognitive processes.

Thinking about intelligence is intelligent. Understanding fundamental principles is intelligent. Proving theorems and putting forward theories is intelligent. Playing the game of Go is intelligent. Playing football is intelligent. Walking is intelligent. Moving chess pieces is intelligent. Grasping is intelligent.

The following section shifts the focus to a presumed trait of intelligence – complexity of the solved problem –, which may even shuffle the perceived order in the above enumeration.

1.2 Intelligent between easy and hard

In the 1980s, following one of the crises in the belief in Artificial intelligence (the AI winters), Hans Moravec observes the paradox that the computational effort needed to solve a problem does not match our perception of how hard the problem is. For instance, we unconsciously recognise the sentiment of a person in a few seconds, but need to concentrate our attention to make the right move when playing chess. However, algorithmically the judgement of sentiment turns out to be very complex, while machines easily compete with an average chess player. Moravec explains that the evolution has had enough time to develop the solutions of the really complex problems and equip us with them. The instruments to solve the others have been only developed for the last few thousands years – they are not so advanced and this makes us perceive the problems as hard. In his words (Moravec, 1988, p.15):

Encoded in the large, highly evolved sensory and motor portions of the human brain is a billion years of experience about the nature of the world and how to survive in it. The deliberate process we call reasoning is, I believe, the thinnest veneer of human thought, effective only because it is supported by this much older and much more powerful, though usually unconscious, sensorimotor knowledge. We are all prodi-

gious olympians in perceptual and motor areas, so good that we make the difficult look easy. Abstract thought, though, is a new trick, perhaps less than 100 thousand years old. We have not yet mastered it. It is not all that intrinsically difficult; it just seems so when we do it.

This explanation, however, rises a question which is equally interesting. *How* have the skills become easy? Apparently, unconscious skills live in a weird, exotic, cryptic, complicated manifold. What has nature taken its time for? To cluelessly wander around the paths of the scary, dark, and slippery landscape; or the time was needed to unveil fundamental principles and thus discover a hidden clever viewing angle from which the solution appears just over the bridge, simple and elegant (Figure 1.1)?

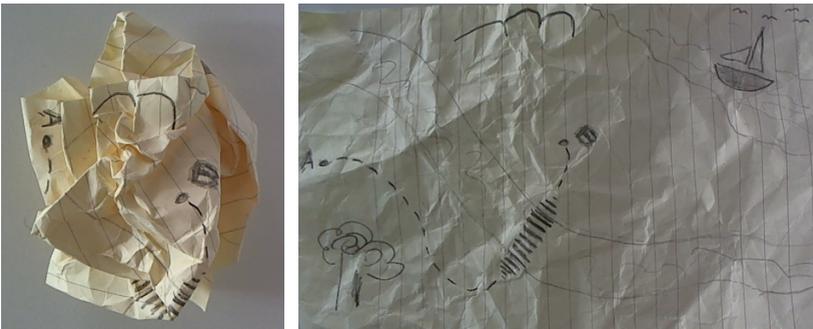


Figure 1.1: It is much nicer to walk an uncrumpled landscape.

A few examples follow, where the solutions are not obvious before changing the way one looks at the problem.

1.2.1 Hard made easy

A toy example from Machine learning illustrates how a simple model can explain the observations if more thought is put into finding the suitable view. Figure 1.2 shows two classes of data points observed in the plane. Obviously, there is no way of separating the two classes with a single line, i.e. the problem

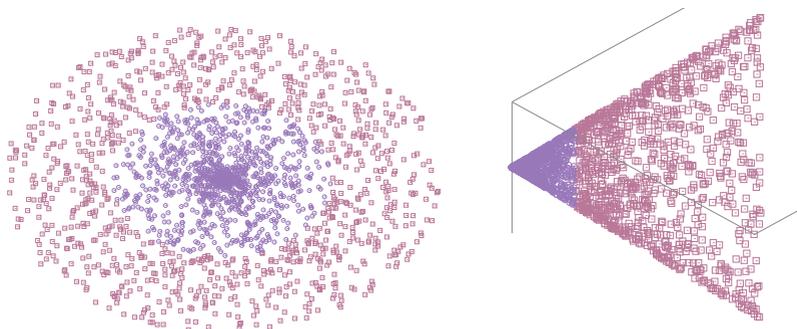


Figure 1.2: linearly separable classes

cannot be explained by a linear model. If one builds quadratic features out of the ‘natural’ ones, the problem lives in $3D$. The dimensionality rises, however, there exists a plane which can separate the classes! Indeed, the observations appear as coming from the inside and outside of a *circle* – a quadratic curve. So, in the hindsight, it is no surprise that working with quadratic features is more suitable and allows to explain the problem with the more simple linear model.

Back to robotics, assume N robots which can have different speeds, however are limited to all move along a single direction at a time. A particular micro-robotic system motivates Bretl to search for the optimal trajectory which brings all robots to their goals. The solution, at first sight, does not seem straightforward. An elaborate derivation of the optimality reveals surprising structures which relate to convex hulls and reachable sets. The optimal trajectory turns out to be very efficiently computable – dominated by the convex hull computation – and despite the limitation of movement is guaranteed to be at most $\frac{1}{2}\pi$ worse than for uncoupled moving robots (Bretl, 2012).

Consider a planar elastic kinematic chain with one end fixed and other attached to a robot arm. Manipulating this chain so that it remains in equilibrium throughout the whole trajectory is another particular problem McCarthy and Bretl (2012) concern themselves with. They show that the task space cor-

responds to the intuition that the problem is three-dimensional – rotation and translation of a robot end effector. Furthermore, they discover the smooth 3-manifold which contains the equilibrium states and thus allow for much simpler planning. Irrespective of the n degrees of freedom of the kinematic chain, planning remains 3D, i.e. constant; only the translations to configuration and task space grow linearly with n .

Thus, choosing the proper viewing point makes above challenges ‘easy problems in robotics’ – Bretl’s talk title at an ICRA 2013 workshop.

The beauty of science is that it shows us such principles and strengthens the belief that the world follows fundamental rules accessible to us. Such thinking might be a cognitive bias. Nevertheless it is a great motivation to keep seeking for these principles. And getting closer to them eventually makes the world more comprehensible and comfortable. This work happens to seek them in the domain of robotic manipulation. Grasping is a skill we use without realising the complexity of the task before we try to implement it on artificial systems. Once aware of the true complexity, we can study the factors involved in it and look for viewing angles which make the landscape nicer. In our terms this means to develop *representations* – ways to abstract and organise knowledge about the world, the own body, its capabilities, about the objects and tasks.

1.3 Grasping

We should ask ourselves what aspects seem to impact grasping, the trivial they can be, and then look for a representation which is capable to capture the relations between them.

Timing. We observe that a grasp is not just a pose at a fixed moment. The final pose of the hand is a result of the way which leads to it. And conversely, the way we want or are supposed to grasp determines which path the hand

should go. Grasping has a time component.

Uncertainty. Even a familiar object will likely not be perfectly known (one usually cannot name the exact dimensions of one's favourite coffee cup). They need to be perceived, even more unfamiliar ones. The sensors through which we perceive the object are imprecise. This introduces uncertainty about the object – its identity, shape, position, etc.

Interaction. Grasping is an interaction with an object. It is not a fixed motion without feedback from the object. This makes sensors inevitable in the loop. Moravec (1988) mentions perception and motion very often together as examples for what we are good at. He even speculates about using that powers where we do not perform so well:

[...] the large highly evolved sensory and motor portions of our brains seem to be the hidden powerhouse behind human thought. By virtue of the great efficiency of these billion-year-old structures, they may embody one million times the effective computational power of the conscious parts of our minds. While novice performance can be achieved using conscious thought alone, master-level expertise draws on the enormous hidden resources of these old and specialized areas. Sometimes some of that power can be harnessed by finding and developing a useful mapping between the problem and sensory intuition.

More recently, in his contribution for the Berlin Summit on Robotics, Schaal (2011) observes that

[...] a bigger question for robotics becomes how to start a more comprehensive approach to perception-action-learning systems, an approach that emphasizes the need to address all these topics in an integrated way rather than treating them as independent research topics.[...] Moreover, perception, action, and learning are tightly linked into a functional system, i.e. it is hardly conceivable that the building blocks were developed independently of each other.

This proposition we illustrate more algorithmically in Appendix B in a thought experiment inspired by Avicenna's one from the Middle ages.

Chain link. Grasping is a means, not an end in itself. It is connected to other activities as a part of a higher level plan. We grasp not for the sake of grasping, but in order to do something with an object. And we did something before that. To illustrate this point, imagine how one holds a pencil to write with it and to use it as a lever. For the same object one uses very different hand postures depending on the context and purpose.

Embodiment. Grasping and hand are one. The hand, the hardware, provides clue to the controller, what is it capable of doing; and, the hardware, e.g. human hands, formed along the evolution with respect to what it was controlled to do. The dexterous and precise manipulation which the human hand is capable of started to become possible when million years ago the upper limbs were not used for walking anymore and opposing thumbs started to emerge which facilitated power grasps and some pinch grasps. In the converse direction, our brain could receive feedback for successful grasp only if its control signals are tailored to what the hand can do – and only such can be rewarded and reinforced in the process of learning to use the instrument.

This is not an exhaustive list, however, it gives examples of relevant information in the world model we assume: our approach to grasping is based on shape, it is a geometric one, i.e. we focus on rigid bodies, abstract texture and friction. Here are some examples of how our approach address the above observations:

Timing. We consider the motion of the whole body for grasping – not the hand posture isolated from the kinematic chain. Our heuristics consistently lead the hand over the whole trajectory which results in a smooth and natural motion.

Uncertainty. With our representation we translate the imprecision of the sensors into uncertainty about the object shape. We use this uncertainty to make grasping more informed and focus it on particular object regions.

Interaction. Interaction is a possibility to refine the knowledge about the object, to actively experience the object in the way which is most informative for the given task. Acquiring more precise model makes a successful grasp more probable.

Embodiment. The representation takes account of the embodiment. For instance, the models obtained by sensing can accommodate a prior about the rough size of its details. In other words, a prior can restrict the information w.r.t. capabilities of the body.

An example of what we do not address here is the embedding of grasping in cognitive plans. We stay on the level of primitives on the lower level of hierarchical controllers. Out of our scope are also observations which may play crucial role when dealing with soft and articulated objects where their properties may be used for more sophisticated manipulation.

1.4 Motivating scenario

The following task is so simple that everyone can master it unconsciously, yet it contains challenging aspects when transferred to a robot. We will decompose it, marvel at the ingenious skills nature developed and try to solve some of the challenges.

Alice and Bob play a game. While Bob is outside the room Alice wraps a scarf around an object and puts it in a bag. Bob comes in, gets the bag and has to guess from only feeling around the bag what is the object. If he cannot guess in a while Alice gives him hints in form of associations or the purpose of the object or its appearance.

Purely combinatorially, it is intractable to enumerate all objects in the world, not even only the ones known to the subject, and decide which one is the best guess. Yet Bob is able to recognise the object mostly in a few seconds. In more complex cases, even a very remote hint can lead to the right answer. Why does it work? Without a pretence at completeness or neuroscientific correctness, we speculate that we observe the following. The very fact that Bob sees the bag, gives him an idea of the size of the object, leaving large uncertainty about its full extension and exact shape. With the bag in his hands, he can judge the weight. Starting to feel around, he observes points from the surface, though contaminated with noise (mind the scarf!). However, he does not get all contacts at once, he grasps the object again and again at different places, getting just a few at a time. He can decide where to place each next grasp, as to acquire information he considers most helpful for the recognition. In other words, iterating grasps and updating with each new observation an internal model, it is possible to very quickly reduce these uncertainties and develop a rather precise estimate of the shape and perhaps identify the object from the haptic data only.

If however this information is not enough, Alice tells him that he would normally wear the object on a strap around his neck. With this hint and the rectangular-ish form he already inferred, Bob comes up with the answer: his photo camera. What narrows the search space in a few seconds down to a small number of feasible hypotheses is a fascinating interplay of reasoning, associative memory, motor and sensor skills! The capabilities involved in this example are common for wide range of tasks and skills related to interaction with the world. Their technical, robotic rather than human, implementation is what we consider in this thesis.

1.5 Novel representation and methods

Why do we need another representation? Skilled manipulation, grasping in particular, is still challenging for robots. Apart from controlled environments and objects for which a complete model and thus precomputed grasps are available beforehand, we still have way to go before we know how to reliably grasp. It gets the more obscure, the less we know about the object. The representation – the way the available pieces of information are related to each other and the important is filtered – determines the viewing angle. And an integrated view on grasping calls for a novel representation to capture the important facets. We believe that how a target object can be grasped is constrained to a large extent by its shape. The choice of an adequate way to describe the shape is therefore critical. What is a good choice of shape representation for grasping? It has to account for three key requirements: (i) the representation has to serve directly as a basis to formulate a motion controller to generate a fluent reach and grasp motion; (ii) the shape representation needs to coherently fuse the different information sources (prior knowledge, multi-modal sensors); (iii) it should be able to deal with uncertainty – on the one hand, to judge what model explains the uncertain observations best, on the other, to ‘know what it knows’, i.e. report its confidence to the controller.

To motivate the requirements more clearly, note that information about the shape of an object can be acquired by observing points from the surface. This qualifies different kinds of sensors for the task: light based sensors include cameras (triangulation of points via stereo vision or optical flow), laser, structured light (kinect) or – in direct contact – haptic sensors. They all have different precisions and applications. Another information source, the prior knowledge, could range from concrete initial guess about the shape to assumptions about the connectedness or smoothness of the object. The fusion of all these uncertain channels together with prior knowledge – typically in the Bayesian sense – should provide a shape estimate of the object. For in-

stance, when bad lighting conditions allow for only poor image quality, we should be able to represent a very uncertain shape estimate; when the object gets in reach we get additional more precise haptic feedback and should be able to combine both sensory sources. The need of sensor fusion for shape estimation is only half the story. Ideally, a representation should also provide a convenient basis for manipulation, especially reach and grasp motion generation. Precisely, an accurate shape estimate is not the primary objective; it is rather the combination of all sensory information to support the control of a reach and grasp motion. This includes what we stated in (c) earlier: the controller needs to be aware of the model's confidence.

1.5.1 Representing is getting the right features

Representations are often assumed implicitly given when developing algorithms. They often do not receive the attention and conscious discussion that they deserve. This can be illustrated on a Reinforcement learning (RL) problem. Wide range of problems – where an agent needs to take actions to learn its environment and its own capabilities and eventually reach a goal – can be formalised with RL in terms of statistical learning. In the usual RL setting, an agent can take actions according to its current notion of state, which brings it in a another world state, and it receives some reward for that. The goal is to find the proper actions (policy) so that the accumulated reward is maximal. The agent's experience is encoded as a sequence of such state transitions due to taken action and the encountered immediate reward, (s, a, s', r) , with state current state s , successor state s' , reward r and action a (Sutton and Barto, 1998). From this information, the agent needs to build its world model and/or policy. A common setting is a 2D grid maze in which the agent's possible actions are move to a neighbouring box. The state of the world is the occupancy of the grid and the rewards resulting from acting. In such worlds, effectively, one takes s and a for granted. This means precisely that one knows

the proper way to extract from the *full* state, from the space of all sensor readings \hat{s} , the relevant information for the problem one needs to solve. Similarly, a is assumed to contain only the meaningful actions out of all combinations of controllable degrees of freedom (DoF), \hat{a} . In toy grid worlds, this is feasible, since the ‘sensor’ input is exactly what one wants to know (e.g. grid occupancy), and the available degrees of freedom are exactly the possible actions (e.g. move south, north, west, east), $a = \hat{a}$ and $s = \hat{s}$.

In the context of autonomous complex robots, both assumptions are not trivial to make: selection and combination of available information and actions becomes a problem in itself. Thus, it is the representation which holds the answers and needs care.

1.5.2 Representation: a learnable, affording grasp shape

In this work we propose a scheme which we instantiate with a particular representation, Gaussian process implicit surface potential. The general scheme is to take a continuous representation which gives clue on how to grasp an object, to parametrise it in a way which is suitable to be learned incrementally from experience by interaction. Then statistical methods deal with the uncertain sensory input to estimate a shape and, beyond that, be aware of its uncertainty. In our particular realisation, we reason about functions rather than shapes. The function implicitly describes the shape as a level set – an implicit surface (IS). This allows us to let a Gaussian process (GP) estimate the function, which gives a whole probability distribution over functions and thus probability distribution over shapes. We chose the function so that it can be interpreted as potential field generated by the surface, similar to the gravitational potential. The attraction by the surface provides the convenient basis to navigate the robot’s arm, hand and fingers for the grasping task. This mix of concepts for representation and navigation we call Gaussian process implicit surface potential (GPISP).

1.5.3 Methods for estimation and control

The above scenario is about approximating a shape, i.e. it addresses the accuracy of the internal representation. But more importantly, it points up that interacting with the object is the only way to gain accuracy and reduce the uncertainty. Looking from the perspective of another task – grasping – this corresponds to the most obvious thing one would do if one fails to grasp a thing: try again. However, one would then know more than before. The insight that the object model needs to benefit from the information gained from unsuccessful grasps suggests an iterative scheme for reducing uncertainty for grasping. This means to leverage a motion generating grasp primitive by the uncertainty aspect.

The notion of uncertainty in the model assigns levels of interest to particular surface regions. Then we can instruct grasp controllers to *grasp at known regions* or go to the uncertain regions and thus *more effectively collect information*. These correspond to a primitive which more successfully grasps, and another primitive which is better at collecting information. In addition, exploring with the latter primitive collects information which is relevant especially for grasping. For instance, the fingers of a hand would approach from opposite directions and explore opposing regions, just as if they were about to grasp. Such a primitive *gains useful information with respect to its own constraints*, rather than sampling the world randomly or only exploring a neighbourhood at a time – yet another manifestation of the observation that the embodiment of a learning system provides the right restrictions to the search space to make learning feasible.

1.6 Agenda, structure, contributions

In the following we will develop a representation and methods for a robot with tasks in mind which are similar to Bob's – manipulation of previously

unknown/uncertain objects using haptic data and an iterative exploration and grasping strategy. In contrast, we will not integrate more complex to interpret associative information, and not go into object recognition. The main demonstration at the end of the thesis addresses estimation of the shape from multiple grasps with active selection of contact points. This involves several issues: (i) how to represent the current estimate and uncertainty (belief) about the object, (ii) how to update the model incrementally with new data, (iii) how to use it to produce the motion needed to contact the object, (iv) how to design active learning strategies for the robot to decide on subsequent grasps, and (v) how to integrate these in a closed sensory-motor control loop. The thesis contributions are

- a formulation of the requirements for dealing with the complexity of grasping and a representation which answers these requirements and captures information relevant for shape-based grasping;
- a translation of sensor feedback to data points to approximate in terms of the above representation an object from multi-modal data and prior information;
- heuristic grasp criteria for generating grasping motion – given a shape estimate – and their formalisation as an optimisation problem;
- methods to use the uncertainty in the model for better informed object manipulation;
- the integration of motion and sensing into a sequential grasping policy to reduce uncertainty and facilitate grasping and shape estimation, as well as measures to support automated decisions.

The object representation and the methods around it depart from some traditional assumptions about grasping. Drawing links between different paradigms, they take a wholistic approach to reach and grasp motion, connect sensory and motor paths, relate them to hand synergies and grasp affordances.

The task space suggested by GPISP supports well both interpreting observations into a model and using it to reason and interact with the world, aware of the imperfectness of the model. This ability to naturally account for motion, sensing and uncertainty announces that many aspects of grasping in particular – and low level intelligence in general – can be jointly dealt with. This nears us to the understanding of the principles behind the truly complex problems we began this chapter with.

Below we proceed with the Background chapter. The sections in it can be safely ignored by a reader familiar with the concepts suggested by their titles. They are meant to only give high level intuition and help the reader to locate this work in his context.

Then the contribution chapters are built around the following skeleton: We first describe the idea behind our representation in Chapter 3. Next we show how a model of an object emerges from prior information and data (sensory information). The representation's connection to the motor system is elaborated on in Chapter 5 where we formalise heuristics for motion generation based on GPISP. Chapter 6 shows how the uncertainty of a model can be used to improve the controlling of the robot for manipulation tasks. In Chapter 7 we demonstrate a particular scenario – blind estimation – in which we integrate the developed tools. Each of these chapters starts with a review of existing literature related to the particular problem and ends with a discussion. Finally, Chapter 8 concludes and suggests directions for future research.

Note that substantial ideas found in this work as well as part of the text have been published in conference proceedings (Dragiev et al., 2011a, 2013b), as workshop contributions (Dragiev et al., 2011b, 2013a) and in internal reports.

Chapter 2

Background

This chapter provides a wide-angle perspective on problems our contributions are related to as well as on fields we borrow techniques from to build upon in the following chapters. First we discuss shape estimation and robotic grasping, comment on ways to represent the shape of an object and emphasize a particular one we use later, implicit surface; we touch on the problem of collision-free trajectories; further, we comment on uncertainty and statistical learning; then we introduce concepts like inverse kinematics and optimal control under multiple objectives – techniques involved in moving a robot; then, we review Bayes’ theorem as a general framework to reduce uncertainty, and introduce briefly statistical learning and Gaussian process as a particular learning approach.

2.1 Problems and domains

The methods developed in this thesis are related to the topics we naïvely introduce in the following: shape estimation, object representations, motion generation, grasping and manipulation, handling uncertainty in perception and control.

2.1.1 Shape estimation

Applications from very different domains need to estimate object shapes from a set of probes: in the context of, among others, radars (Das and Boerner, 1978), medical imaging (Chen et al., 1994), underwater inspection (Hollinger et al., 2012), terrain modelling (Vasudevan et al., 2011). Devices can sense at certain temporal and spacial resolution. If the resolution is lower than a target application expects, the raw data needs to be augmented, enriched to match the requirement. For instance, the points on a surface scanned by a rotating laser sensor at an angular resolution of 1° on $1m$ distance will be more than $1.7cm$ away from each other. Scanning from $10cm$, the gap remains more than $1.7mm$. Assume one needs a model of the object for rigid body simulation. Typically one wants to close the gaps between the points and obtain a connected surface. How can this be done, keeping in mind that the sensors are usually noisy, thus even the measured points are only approximation, rather than ground truth? Bolle and Vemuri (1991) put the methods in two categories according to whether they describe the surface implicitly or explicitly. For a particular sensor type – camera – and a use case which exposes most of the challenges Seitz et al. (2006) review comprehensively methods for stereo vision reconstruction from multiple views; many examples can be found as references therein.

Regardless of the used object description, a related question is how to select from a set of known objects the one that fits the observations best, i.e. retrieval from shape databases. When the database contains just a small number of parametric regular shapes, this amounts to reducing objects, or their parts, to these primitive shapes; Miller et al. (2003), for instance, use primitives for grasping. Later we review in more detail related approaches (Zha et al., 1998; Ohtake et al., 2005). Problems in the context of shape database retrieval, in particular dissimilarity metrics, have been discussed in a survey by Tangelder and Veltkamp (2008). Quantifying dissimilarity allows to formulate

both retrieval and reconstruction as optimisation problems.

2.1.2 Robotic grasping and manipulation

Robots already play a big role in our society – without the automation level we’ve reached in production, the quality of life wouldn’t be maintainable. However, industrial robots act in very well structured and controlled environment. This is the main reason why the achievements in speed and precision haven’t been transferred to everyday life. The more we move away from industrial robots, the interaction between the machines and the world becomes less structured and unpredictable: things are not always at the same place, not properly positioned. This calls for more flexible methods for interaction. Different developments answer this demand: for instance, recent developments in locomotion allow to cope well with arbitrary terrain (Raibert et al., 2008); more relevant for this thesis are the research efforts of the last decades devoted to hardware and methods for grasping – realising its importance for goal-directed manipulation. Classically, dexterous manipulation is considered separately from grasping and from object estimation. Bicchi and Kumar (2000) review work on robotic grasping looking at how contacts and forces are modelled; how to analyse grasps by force and form closure and their contribution to fixturing of objects; as well as how to measure grasp performance with respect to criteria like stability, controllability, visibility. Okamura et al. (2000) survey dexterous manipulation which involves planning not only the static grasp, the way the robot comes to the grasp but also how to move the robot according to a desired trajectory of the grasped object. They point out that manipulation and estimation ‘go hand in hand’ and identify a still valid key limitation: sensor feedback, its obtaining, processing and integrating in the control. The diversity of forms, materials, surfaces, goals for which a robot needs to be equipped in a non-controlled environment is challenging. An approach to deal with this is (Okamura et al., 2000) to break the problems into

small ones, research them extensively – e.g. the laws of pushing by Mason (1986) – and build specialised hardware and algorithms. Luckily, most everyday objects exhibit certain (not necessarily comprehensible for us) structure: they are built with human technology and are built for human hands. Thus, ‘the majority of robot hands designed for dexterous manipulation are anthropomorphic in design’ (Okamura et al., 2000). This and the copying or learning human ways of grasping are en vogue not because they look chic, but because they capture the variability of every day objects and situations. While most of the research is concerned with rigid bodies, a survey by Jiménez (2012) categorises approaches to manipulation of deformable objects – according to the type of objects and to the desired deformation. Notably, however not surprisingly, pioneering work in their modelling has been done in the context of Computer graphics (Terzopoulos et al., 1987; Gibson and Mirtich, 1997).

Bohg et al. (2013), in a very recent survey, attend to the joint problem of perceiving an object and synthesise a grasp for it – as they term it, data-driven grasping. They differentiate by how familiar the object is, by grasp sampling techniques, by grasp quality estimation and by the representation of good grasps. The approaches considered in the survey’s section on unknown objects (notably, they are quite recent – from the last five years) bridge the previous and the current section. If one were to relate the current thesis’ contribution in such terms, it would be along the lines of *model-based* grasping of *unknown* objects with *heuristic* quality measure. (A model of the object is acquired from data, rather than seeking patterns in the data, which would be model-free.) Relating it to the division scheme in (Sahbani et al., 2012) it would fall under approaches which extract data from object features (as opposed to data from human demonstrations).

2.1.3 Collision avoidance for grasping

In the context of grasping, an *obstacle* is part of the environment which is not important for the carried out task in the sense that the grasping does not aim at interacting with it. Examples of obstacles include the ground, the table on which lies an object the robot have to grasp, other objects in the surrounding of the target, a human operator. In the presence of obstacles, the importance of having a collision-free trajectory grows with the stiffness of the hardware: the more compliant, the more one can tolerate collisions; the stiffer, the more critical are collisions. The consequences of a collision are usually undesirable and need to be avoided. They can be unfavourable for both obstacle and robot – for safety and security reasons: the impulse of stiff hardware can displace or break an obstacle or damage itself. Even without causing damage, the trajectory may become unfeasible if during execution a contact causes a large perturbation. Another source of harm for the robot is a collision between its own body parts. All these cases need to be taken care of when moving a robot so that the trajectory is *collision-free*. Approaches to this problem have been studied in various contexts: starting from mobile robots, where collision avoidance is the essential part of navigation, later extended to mobile manipulators, both in continuous or discrete working areas; examples (which we review later in more detail in Section 5.1.2) include potential navigation (Khatib, 1986), path search in occupancy graph (Hart et al., 1968) (e.g. A^*), the family of BUG algorithms (Lumelsky and Stepanov, 1986).

2.1.4 Object representations

An agent needs to organize its knowledge if it is to interact autonomously and learn from the interactions with the perceived reality. They need a way to track changes and effects in order to enrich their knowledge. At different abstraction levels and aspects of cognition (e.g. in explaining his ‘frames’ for knowledge organisation Minsky (1974) goes through vision, language, prob-

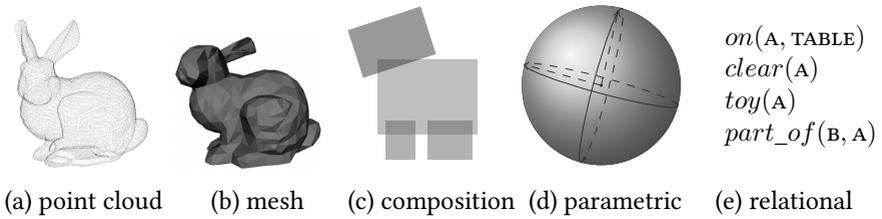


Figure 2.1: Different representation of a bunny. Abstraction from appearance and data grows from left to right.

lem solving, etc.), the relevant knowledge has different nature. When planning sequences of complex actions for achieving a goal, the individual objects may be of less interest, and individual manipulation actions can be considered as available primitive building blocks. For instance, adding milk to the cup of tea involves finding and opening the fridge, finding and grabbing a bottle of milk, moving to the table, pouring into the cup of tea. The particular bottle is indeed irrelevant to the algorithm which produces the high level plan, it rather works with information which relates the objects and their properties (Pasula et al., 2007). Thus, the representations such algorithms employ need to compactly capture these aspects of the world. However, if we zoom in on the primitives, relevance changes. In this thesis, we act on the geometric level – we estimate object shapes and poses and produce robot trajectories which operate on them –, thus, we need rich geometric representations. The world consists of objects, moreover, when physically interacting with the world, what we change are properties of the objects: their position, shape, relations. Expectedly, the sensible way to organise knowledge is in geometric object representations. Beyond the purpose question, technical aspects are addressed in surveys like the one by Besl and Jain (1985), who focus on volumetric representations. Bolle and Vemuri (1991) look at surfaces and point out the importance of the chosen frame of reference for the geometric representations – arguing in favour of object-centric frames due to invariance to views.

Figure 2.1 shows, for a single object, several examples for representations suitable for different purpose. The most left image is a point cloud of a terra cotta figure, the Stanford bunny, obtained by a laser scan. A point cloud holds the 3D coordinates of all points. Common practice is to process these points to a polygon mesh (Fabio, 2003; Rusu et al., 2008), like in 2.1b. Meshes are widely used in visualisation, they have several advantages, including: images can be rendered on their faces (Hernández Esteban and Schmitt, 2004); one can grow the faces and discard points which lie inside faces (especially useful for large flat surfaces); meshes implicitly contain the orientation of the manifold, i.e. what is the inside and what is the outside; one can detect intersections of polygons with other objects easily, etc. Meshes still need to hold the 3D coordinates of their nodes, as well as the triangulation information, i.e. which face they belong to. Consider a mobile platform (Rimon and Koditschek, 1992) which moves in a room with some bunnies it should avoid to collide with. Then the fine details of the bunny surface are too small scale to be relevant for the task. We can inscribe the bunny in a sphere, e.g. 2.1d, and use this sphere to represent it for the purpose of collision avoidance. (As we see later, a point source of repulsive potential does this.) We effectively trade a lot of accuracy off for a very compact representation using four numbers, the 3D position and the radius of the sphere. The latter might be too aggressive simplification, but we can remain close to the original shape and still largely reduce the representation by using primitive shapes to approximate it. For instance, consider an arm with a gripper which can very reliably grasp boxes (Huebner and Kragic, 2008). Then it could be helpful to fit the shape as a set of boxes, like 2.1c naïvely illustrates. The parameters – the dimensions, orientations and positions of the boxes – are still compact but retain more structure and provide hints to a box grasping algorithm. Finally, a representation is not only about appearance; a representation may contain relations to other objects and completely abstract appearance, e.g. the example in 2.1e states that the object of interest is on the table, has nothing on top of it, the robot has nothing in the

hand. From these facts an algorithm may deduce, for instance, that the robot can grab the object and plan sequences of actions (Lang and Toussaint, 2010).

It is common that better informed algorithms have better performance, but if they are flooded with information, the performance drops due to processing limitations, i.e. the designer always needs to trade rich information content off against performance. What makes a representation successful for particular purpose is the right degree of abstraction; a representation can only be evaluated related to an application. Large variety of object representations have been developed having different focus in mind. Depending on application, they abstract information which is not vital for the approach and compactly store what is.

Where do the good representations live? While the above paragraphs deal with man-made ones, there are naturally emerging ones studied by neuroscientists. Neurophysiology looks at the brain and finds areas in primates' cortex and even single neurons which are associated with the visual appearance of objects; others are responsible for specific movements; and some even associate actions (grasping) to specific object structures (Murata et al., 1997). Further brain imaging evidence suggests that distinct visual representations are used for the planning and for the execution of actions (Glover, 2004). On this background, developmental psychologists and developmental roboticists suggest that the compact and efficient representations evolve through the embodiment of the intelligence (Pfeifer et al., 2007). Our bodies have certain capabilities, we live in a concrete reality and have specific sensors to perceive the reality. The embodiment carves the representations in the sense that it provides the measure for what is possible and sensible with the body's capabilities and in the current environment. We will come back to discussing representations and go deeper into grasping related once in Chapter 3.

2.1.5 Uncertainty

The sensors deliver all the empiric information about the true world. Sensor readings are tainted with noise which makes the derived knowledge uncertain. But with collecting more evidence the uncertainty decreases, i.e. we gain more confidence about the true state of the world. Bayes' rule is a general framework to describe the accumulation of knowledge using conditional probabilities (Bayes is credited for the idea used in a special case published posthumously by Price (Bayes and Price, 1763), Laplace (1814) is credited for the general formulation; refer to Russell and Norvig (2010) for a broader picture of the history and application):

$$P(C|E) = \frac{P(E|C)P(C)}{P(E)}. \quad (2.1)$$

Often, we observe the effects of a known cause $P(E|C)$, but are interested to predict the cause. The rule relates the belief in C to observing $P(E|C)$. This *posterior* belief in C when seeing E is proportional to the *prior* probability $P(C)$ and the *likelihood* of E being caused by C , $P(E|C)$. If the posterior needs to be interpreted as probability, it has to be normalized to sum to 1 for all E (Kolmogorov, 1950) (s. next section), hence the division by $P(E)$. Thus, Bayes' theorem formalises the process of how a robot iteratively gets to know the world. In order to work inside this framework, we need internal representations which are capable to deal with such notions – such probabilistic representation is the topic of the present thesis.

We might need to take a step back and justify why we use probabilities in the first place: life is not a game of chance, is it? – one could argue. Regardless on one's view on the determinism of the world, one is trapped alone within an oblique view of the reality. Russell and Norvig (2010) introduce the probability assertion notation as an extension to first order logic (FOL) assertions. They also discuss the different interpretations of probability: being property of the events – and thus objective – or being matter of subjective belief. Working with probabilities coupled with the subjective view is more powerful than FOL

in the sense that it allows to compare worlds not only in terms of whether they are possible, but also express how much, given the available information, we believe that they are possible: ‘Logic can tell us that it is unknown whether there is a pit in [2, 2], but we need probability to tell us how likely it is.’ Quantifying likelihood turns out to be useful not only in indeterministic settings, but also for modelling ignorance (information *is* incomplete by the finiteness of agent’s presence in space and time).

Today’s rigorous formalisation of probability theory is due to Kolmogorov (1950) who derives it from simple axioms, e.g. that the probabilities for all possibilities sum to one. Contributors to probability theory from ancient scholars through middle ages include Laplace, Pascal, Huygens. Research has been often linked to games of chance. This context provides another justification for using probabilities to deal with uncertainty: de Finetti (1993) shows that the only way to be consistent with rational decision making is to adhere to Kolmogorov’s axioms.

Probability theory provides the instruments to manipulate propositions with different levels of belief and use them for making decisions. There are other well developed formalisms which aim at similar problems – fuzzy logic for instance – however, we believe that they focus on slightly different aspects of propositions, e.g. imprecision, inexactness, ambiguity.

The notions of prior and conditional probability distribution, (conditional) independence and Bayes’ rule underlay most approaches to learning machines, i.e. to algorithms which learn from data rather than being pre-programmed, often under the names machine learning or statistical learning.

2.1.6 Statistical learning

Statistical learning provides methods to extract information from available evidence and use it to draw conclusions about unseen relations. Langley (1996) gives a practical definition of learning as

the improvement of performance in some environment through the ac-

quisition of knowledge resulting from experience in that environment.

A systematisation of learning problems goes often along the available supervision (Hastie et al., 2009). In *unsupervised* learning one seeks for the structure in a bunch of data points and the result are a few clusters which group similar data or other insights about the distribution of the data. *Classification* and *regression* are examples of *supervised* learning – the supervisor provides labels to the inputs and thus explicitly says what to learn. In *reinforcement learning* (Sutton and Barto, 1998), one tries to figure out appropriate actions which maximise the reward an agent receives upon acting. Thus, it is placed somewhere between supervised and unsupervised learning – giving reward is a form of guiding the learning, but it does not necessarily praise the immediate action, it rather encodes a relation that the agent needs to figure out. Other forms of *semi-supervised* learning take advantage of both explicitly labeled and unlabeled data to improve the prediction. Dividing classically along supervision addresses typical use cases rather than radically different methods. Eventually, the model that fits the data is searched for by optimisation in the space of acceptable models.

The more complex models are used the more precisely can be explained the available data. Usually, the input is contaminated with noise or is uncertain (which we express, again, by assuming noisy input). As consequence, if every available evidence is explained perfectly by a learning method, it ‘explains’ the noise, too. This is disadvantageous, because, in the end, the aim of the learning is to capture the structure of the problem and be useful on unknown data, rather than to predict perfectly the evidence which is known anyway. Thus, simpler models are preferred (Occam’s razor). To protect learning methods from choosing unnecessarily complex models we usually apply restrictions – penalise complexity to some degree, forcing the method to be sane, try to find the best fitting model for the available data but staying within a family of models (Müller et al., 2001). The choice of restriction has to be based on knowledge available prior to evaluating the particular evidence. There are

different ways to specify the restrictions, depending on the formalism used for learning. Specifying *priors* in probabilistic methods is largely equivalent to *regularisation* in optimisation terms.

Models can be evaluated in terms of their performance on different data. The training error shows how good the model captures the training set, but is of limited usefulness, since what matters is the performance on unseen data. The true test error shall be using a subset of the available data only for the purpose of evaluation (Kohavi, 1995). Much on the relations between training error, estimated and real test error has been contributed by Vapnik (1999).

There are various approaches to the above problems, which differ in the way they formalise model families, the way they achieve generalisation, etc. *Probabilistic* formalisms provide interpretation of the learned model as distribution over labels for a given input – a feature we treasure and use here. For one such formalism, Gaussian process, we give an intuition in the following section.

The perspective of the overview by Mitchell (2006) on Machine learning is still largely valid: he is pointing how it is shaping Statistics and Computer science and how in turn is shaped by them; and he also frames the important short and long term research questions. In addition, recently, with the penetration of machine learning applications and rising awareness in the society, some ethical questions gain importance: whether and under what circumstances applications previously impossible need to be regulated. On the technical side this raises questions to understanding and interpretability of datasets, features and algorithms.

2.2 Methods

A basic idea of the tools we use can help in following the explanations later and the contributions of the thesis. In this section we introduce – to whom is unfamiliar with these concepts – forward and inverse kinematics, Gaussian

processes, implicit surfaces and an approach to control under multiple objectives.

2.2.1 Forward and inverse kinematics

In the common usage of the word *robot* – in fiction and reality – one assumes a system which has physical actuators which can move and thus impact the physical world, i.e. change its state: move or deform objects, or move itself, e.g. (Čapek, 1928; Asimov, 1950; Sakagami et al., 2002; Seok et al., 2013; Metta et al., 2008; Rooks, 2006). The actuators are necessarily made of links of certain shapes, connected by joints. We can assume that the links are rigid bodies, i.e. do not change their shape. This is not too restrictive, though, since many deformations can be thought of as result of virtual joints. (Virtual robots, e.g. chat bots, impact the world by processing information, not directly changing the Newtonian world; these paragraphs do not apply to them.)

The different types of connections between rigid bodies have been described by Reuleaux (1876) and a century later studied extensively by Hartenberg and Denavit (1964). Commonly used in hardware nowadays are *revolute* and *prismatic* joints which allow one *degree of freedom* (DoF), i.e. their state can be unambiguously described by a single real number, $q_i \in \mathbb{R}$. The links connected with joints constitute a kinematic tree. Controlling the robot means to control its impact on the world, i.e. control its actuators. Formally, this means to be able to set the vector describing all controllable DoF, \mathbf{q} . A non-controllable DoF is one which exists, yet we cannot deliberately influence. Some DoF are coupled, so that changing one directly changes the other or changes some constraints of the other DoF. Two examples: we operate a pair of scissors by changing its configuration, the single degree of freedom given by the revolute joint connecting the two blades; opening an umbrella involves sliding a prismatic joint up, which causes multiple other joints to move and spread the cloth – effectively a single DoF.

2.2.1.1 Forward

In such kinematic chains, one important question is, how, given the configuration of the joints, one can determine the position of some point attached to the chain. This is answered by applying a chain of coordinate transformations which correspond to links and joints in the order they build the kinematic chain: translation for link and prismatic joint, and rotation for revolute joint (Waldron and Schmedeler, 2008; Siciliano and Khatib, 2008). In our notation forward kinematics maps the n -dimensional configuration space to the 3D position of a point: $\phi_{T,F} : \mathbb{R}^n \mapsto \mathbb{R}^3$, $\phi(\mathbf{q}) = \mathbf{x}$, where the index T is the transformation which defines how the point of interest is attached to frame F of the kinematic chain. While the transformations corresponding to links are fixed (since links do not change), the joint transformations depend on the current joint parameter, \mathbf{q}_i . Similarly, by ϕ^z we denote the function which yields the orientation of a vector attached to a coordinate frame of the kinematic chain, $\phi_{T,F}^z : \mathbb{R}^n \mapsto \mathbb{R}^3$, $\phi_{T,F}^z(\mathbf{q}) = \mathbf{z}$, indices meaning the same as above. Mostly, it is clear how the point is attached to the kinematic chain and we will drop the indices or subsume them to some descriptive name of the point ('finger tip'). Adding the rate of change of the configuration to the state ($\mathbf{q} := \{\mathbf{q}, \dot{\mathbf{q}}\}$), we can similarly define ϕ^v , the velocity of a point in Euclidean space.

2.2.1.2 Inverse

For controlling a robot, one has access to the configuration space, i.e. is capable of setting joint positions (\mathbf{q}) or velocities ($\dot{\mathbf{q}}$) or accelerations ($\ddot{\mathbf{q}}$) or torques (Waldron and Schmedeler, 2008). However, 'put the finger tip at the lid switch' and most other tasks one wants accomplished by the robot suggest other coordinate frames, for instance Euclidean positions, distances in Euclidean space, polar frames, etc. One would typically formalise a task in such common frame and would then want to instruct the robot to fulfill it. Know-

ing the position of the tool tip of a robot arm, how to come up with the corresponding joint configuration? This is what inverse kinematic computes – for given $\mathbf{x} := \phi(\mathbf{q})$, compute the parameters \mathbf{q}_i which give the position \mathbf{x} . This problem is often not easy to solve exactly. Furthermore, it is of limited practical usefulness: typically the robot is in a particular start posture \mathbf{q}_0 and needs to go to the desired $\mathbf{q}_T : \phi(\mathbf{q}_T) = \mathbf{x}$, i.e. we look for a valid path in configuration space to the desired position, not only the position itself. Note that there are usually multiple configurations \mathbf{q} for a single position of the end effector, i.e. the manipulator is redundant w.r.t. the task: this translates to the ability to take different paths in different situations; Furthermore, not every configuration is reachable without violating kinematic constraints like joint limits. Thus, often the inverse kinematic problem is solved iteratively by gradually nearing the desired position. First compute a rough approximation for \mathbf{q}_T , then make a small step in this direction and repeat the procedure until close enough to the goal.

A way to roughly estimate the target configurations is to assume that at small scale, the kinematic function is linear, i.e. to locally linearise around the current configuration. The Jacobian matrix generalises the notion of gradient to vector valued function. Each column of a Jacobian can be thought of as the gradient of a component of the vector. Given the function $\phi(\mathbf{q}) = \mathbf{x}$, the Jacobian at point \mathbf{q}_0 can be used to locally linearize ϕ around \mathbf{q}_0 as in

$$\phi(\mathbf{q}_0 + \epsilon) = \phi(\mathbf{q}_0) + \mathbf{J}_\phi(\mathbf{q}_0)\epsilon, \quad (2.2)$$

$$\mathbf{J}(\mathbf{q}) = \frac{\partial \phi(x)}{\partial \mathbf{q}}. \quad (2.3)$$

Letting this crude linearisation go to the target,

$$\mathbf{x} := \phi(\mathbf{q}) = \phi(\mathbf{q}_0) + J_\phi(\mathbf{q}_0)(\mathbf{q} - \mathbf{q}_0), \quad (2.4)$$

can be used to obtain the rough estimate of \mathbf{q} .

There are several ways to approach this problem – a comprehensive introduction is compiled by Buss (2005). In summary, the kinematic chain is redundant, i.e. the Jacobian is not square and not full rank, and thus not invertible. If it were invertible one could exactly solve (2.4), but this is barely relevant. One approach is to use the Moore-Penrose pseudoinverse instead: \mathbf{J}^+ . It generalises the notion of inverse, exists for any matrix and solves the problem in least squares sense: it yields a vector which is optimal in least squares terms – even if there is no solution. In the here common case of redundant kinematic chain, i.e. full column rank Jacobian, the pseudoinverse is $\mathbf{J}^+ = (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T$. (Note that then $\mathbf{J}^+ \mathbf{J} = \mathbf{I}$, i.e. \mathbf{J}^+ is left inverse of \mathbf{J} .) Applying this to the above equation, the estimated joint space direction is given by $\Delta \mathbf{q} = \mathbf{J}^+ (\mathbf{x} - \phi(\mathbf{q}_0))$. A small step $\alpha \Delta \mathbf{q} / \|\Delta \mathbf{q}\|_2$ should decrease the distance to the goal configuration and can be used in the next iteration (typically, the Jacobian of $\phi(\mathbf{q} + \alpha \Delta \mathbf{q} / \|\Delta \mathbf{q}\|_2)$ is different).

In practice, the pseudoinverse can be numerically unstable near singularities. A way to overcome such problems is the Levenberg-Marquardt minimisation algorithm (Nakamura and Hanafusa, 1986). It introduces an additional tunable (Tikhonov and Arsenin (1979)) regularisation term which penalises the norm of $\Delta \mathbf{q}$. Effectively, the matrix to be inverted is then $(\mathbf{J}^T \mathbf{J} + \lambda \mathbf{I})$ which is guaranteed to be nonsingular. The damping factor λ trades precision off for nice behaviour near singularities of the Jacobian.

A more exotic approach to find $\Delta \mathbf{q}$ is to exploit the fact that $\langle \mathbf{J} \mathbf{J}^T \mathbf{a}, \mathbf{a} \rangle \geq 0$ for any \mathbf{a} and any \mathbf{J} (Wolovich and Elliott, 1984). Now if one updates the position in direction $\Delta \mathbf{q} = \mathbf{J}^T \Delta \mathbf{x}$, the new end effector position becomes $\mathbf{J} \mathbf{J}^T \Delta \mathbf{x}$. The inequality above means that the step's projection on $\Delta \mathbf{x}$ is non-negative, i.e. even if it is very crude, it roughly goes into the 'right' direction. The step scaling factor in $\lambda \Delta \mathbf{q}$ can be appropriately chosen to make a step as far as to land as close to the target as it gets along the direction $\Delta \mathbf{q}$. Working with the transpose is imprecise, but has the advantage of low computational cost.

Finally, an unknown mapping can be estimated if enough data is available: there are learning approaches to inverse kinematics, e.g. by D'Souza et al. (2001). They are especially useful where the kinematic modelling is problematic in the first place (Rolf et al., 2010) and fit very well the philosophy of developmental robotics.

The map ϕ which is used to transform configurations into meaningful tasks may not necessarily be a position in Euclidean space, it could encode any relation of a robot and its surrounding, i.e. be a motion feature. The function, the goal value of the function, ϕ^* , and the Jacobian contain the information needed to move the robot to a configuration which reaches the goal. We call these differentiable motion features *task variables* (TVs). The next section shows ways to combine such features.

2.2.2 Control under multiple objectives

As noted in Section 2.2.1, we usually do not formalise the goals of a robot in its configuration space. Inverse kinematics allows to translate abstract goals into control commands. The abstract frames are usually more convenient, comprehensible and compact for expressing ideas. We've just reviewed how a single objective encoded through a forward function $\phi(\mathbf{q})$ can be iteratively reached. What about multiple objectives, each pulling in its own interest on a redundant system which has multiple ways of fulfilling the goals? Most relevant problems are indeed so and there is a large body of research (Chiaverini et al., 2008). For an everyday example, bringing a cup of tee implicitly suggests (at least) that the robot (i) does not collide with objects on the way, (ii) the cup keeps its orientation upwards, (iii) the movement is smooth enough to keep the tee in the cup. A natural approach is to ask how important is each objective compared to the others and do one's best to fulfill them accordingly. We attach to each TV its relative importance ρ , also referred to as *precision* to convey the intuition that the respective goal needs to be reached that precisely.

This yields the optimisation problem below.

Let $q \in \mathbb{R}^n$ be the n -dimensional vector describing the current joint angles of the robot. We formulate the control problem in the dynamic domain, based on the current state $(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{2n}$. Assume we have m different task variables $y_i \in \mathbb{R}^{d_i}$ (with the i th task variable being d_i -dimensional) and we are given the respective forward function $y_i = \phi_i(\mathbf{q})$ to map the joint angles to a task and its Jacobian $\mathbf{J}_i = \frac{\partial \phi_i(\mathbf{q})}{\partial \mathbf{q}}$ at the current \mathbf{q} . For each task variable one can have desired values y_i^* , desired velocities \dot{y}_i^* as well as associated precisions ρ_i and ν_i for values and velocities, respectively. These task variables define a cost function:

$$c(\mathbf{q}, \dot{\mathbf{q}}) = \sum_{i=1}^m \rho_i [y_i^* - \phi_i(\mathbf{q})]^2 + \nu_i [\dot{y}_i^* - J_i \dot{\mathbf{q}}]^2$$

Using a local linearization of $\phi_i(\mathbf{q})$ this can be expressed as a quadratic form in $(\mathbf{q}, \dot{\mathbf{q}})$.

Apart from this task cost term we have a control cost term penalizing accelerations – proportional to $a^\top H a$ with $a = \dot{\mathbf{q}}_{t+1} - \dot{\mathbf{q}}_t$ – or velocity. The sum of the task and control cost terms is a quadratic term in the new state. At each time step the controller computes an acceleration that minimizes these quadratic costs. Refer to Toussaint (2010a) for more details on the dynamic controller with multiple task variables.

Preceding the above, a family of alternatives for enforcing hierarchy over tasks have been developed – commonly known as operational space control. An extensive discussion and practical comparison is provided by Nakanishi et al. (2008). They use the pseudoinverse to find the desired controls for a kinematic chain. The particular methods differ, first, in what they aim to provide eventually: velocities, accelerations or forces; and second in the details of tasks prioritisation. For a redundant kinematic chain w.r.t. a task, the matrix $(\mathbf{I} - \mathbf{J}^+ \mathbf{J})$ maps into the null space of the pseudoinverse. This exploits the fact that the configuration space is redundant w.r.t. task, there is a whole

manifold of configurations which satisfy the constraints. Then its complement can be used to satisfy further ‘secondary’ objectives, without clashing with the task. The first variant was resolved on velocity level by Liegeois (1977) who introduces the null space idea. Another notable alternative among the discussed is the original force-based formulation (Khatib, 1987). Nakanishi et al. provide a common notation for the different approaches, outline consistently the differences and similarities and propose extensions which cure some deficiencies. In their notation, with \mathbf{x} living in task space, \mathbf{q} in configuration space; subscript d denoting ‘desired’ (position or configuration), and with:

$$\mathbf{x} = \phi(\mathbf{q}), \quad (2.5)$$

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}, \quad (2.6)$$

$$\ddot{\mathbf{x}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}, \quad (2.7)$$

and with $\boldsymbol{\xi}$ an arbitrary vector (velocity or acceleration) to be projected in null space, velocity control amounts to

$$\dot{\mathbf{q}}_d = \mathbf{J}^+ \dot{\mathbf{x}}_d + (\mathbf{I} - \mathbf{J}^+ \mathbf{J}) \boldsymbol{\xi}_1.$$

Acceleration control is similarly given by

$$\ddot{\mathbf{q}}_d = \mathbf{J}^+ (\ddot{\mathbf{x}}_d - \dot{\mathbf{J}}\dot{\mathbf{x}}_d) + (\mathbf{I} - \mathbf{J}^+ \mathbf{J}) \boldsymbol{\xi}_2.$$

For force based control the inertia-weighted pseudoinverse is used:

$$\bar{\mathbf{J}} = \mathbf{M}^{-1} \mathbf{J}^T (\mathbf{J} \mathbf{M}^{-1} \mathbf{J}^T)^{-1}.$$

When using such an approach it is important to have sane, redundant enough description of the task, in order to leave enough room for the null space: The task, i.e. a map ϕ as in Section 2.2.1 is not necessarily a fully described Euclidean position and orientation (6 DoF). The gaze direction of the

robot's head and the hand orientation for grasping a symmetric shape (cylinder), for instance, can be described by two DoF and use the null space to avoid collisions and escape joint limits (Gienger et al., 2010, 2008).

In our experiments and simulation we deal with velocity and position controlled systems. In one case we use the approach we started the present section with – on the Schunk arm; on the Barret WAM, we employ a null space mapping similar to Gienger et al. (2010). The former approach can be viewed as more flexible way to task prioritisation. Separating into task and null space hierarchy a la Liegeois demands to think explicitly in terms of two segregated task classes. Instead, the precisions ρ assign relative importances to task variables without assuming a hierarchy; the hierarchy is gradual and the prioritisation is symmetric in its interpretation. The approaches are eventually equivalent technically, though.

To define our controllers we employ motion features derived from the representation we develop. Within it, we use Gaussian processes, a probabilistic machine learning formalism, to deal with uncertainty.

2.2.3 Gaussian process

A Gaussian process (Rasmussen and Williams, 2006) puts all uncertainly observed evidence together with prior assumptions about a function and predicts how the function behaves – near and far from observed spots. Heterogeneous observations – values and derivatives of a function at different certainty levels – can be combined in a GPs model. The prediction for a function at a query point is not a single value, it is a probability distribution around the most probable estimate: thus, the uncertainty of the observations is transformed to confidence of the estimate.

Gaussian Process regression exploits the idea that the estimate at a certain query point should be a properly weighted average of all the observed points – each contributing according to its similarity to the query.

Formally, a Gaussian process is a stochastic process for which any finite number of random variables are jointly Gaussian distributed,

$$(f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)) \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma),$$

with $\boldsymbol{\mu} \in \mathbb{R}^N$; $\Sigma \in \mathbb{R} \times \mathbb{R}$. This can be used to compute the conditional distribution

$$\Pr(f(\mathbf{x}^*) \mid f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_N)),$$

which means to predict the value at \mathbf{x}^* given the values at $\mathbf{x}_1, \dots, \mathbf{x}_N$. How does an observed value or prior or some uncertainty come into the prediction, what is the intuition behind the formulas? Following the notation from (Rasmusen and Williams, 2006) the predictive distribution for

$$f(\mathbf{x}^*) \sim \mathcal{N}(\bar{f}_*, \mathbb{V}[f_*])$$

has the mean and variance

$$\bar{f}_* = \mathbf{k}_*^T (K + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y} \quad (2.8)$$

$$\mathbb{V}[f_*] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T (K + \sigma_n^2 \mathbf{I})^{-1} \mathbf{k}_* \quad (2.9)$$

where k is the covariance function, \mathbf{k}_* is the vector of covariances between the test point and all observation points, K is the Gram matrix and \mathbf{y} denotes the observed function values. The covariance, referred to as *kernel* in related methods (Müller et al., 2001), measures the relatedness of two locations. In this work we use squared exponential covariance. How strong the covariance decreases with the distance between points is governed by the width of the squared exponential. When querying the value of the function at \mathbf{x}^* the observed values \mathbf{y} are weighted by the ‘distance’ at which they are observed and contribute accordingly to the prediction (2.8). The uncertainty of each observation augments the diagonal of the matrix \mathbf{K} . The covariance para-

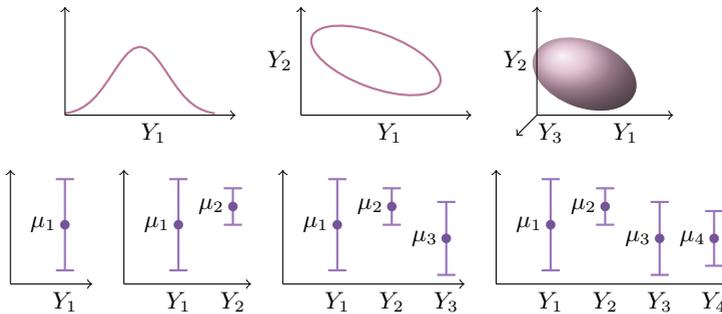


Figure 2.2: Intuition for Gaussian process as a generalisation of multivariate distribution to infinite dimensions.

meter is one way of expressing prior information – the wider the covariance, the smoother the function. Another way is to start with a particular function as bias in the first place and evidence only supplements this bias. Note that Gaussian processes are equivalent to probabilistically interpreted kernel ridge regression (Yu et al., 2005).

Another, sometimes more intuitive, view of Gaussian process is that it generalises multivariate Gaussian distribution to infinite dimensions. A way to visualise this intuition is given in Figure 2.2. The usual way of drawing the Gaussian probability density function is as a bell in 1D, as ellipse in 2D and ellipsoid in 3D. The orientation of the ellipsoid describes the covariance of the multivariate distribution. Instead, enumerating on the x -axis the variables, we could draw densities as a mean and error bars on the y -axis. In such a way, we can easily visualise a lot of densities. Exaggerating this, think of the x -axis being a dense variable index. The curve through all means, the one through all upper and one through the lower error bars yield a typical GP graph. The information that is missing is how the points are related to each other, why the curves have this particular shape? The answer is given by the covariance function: we need to add it to the picture. With it, the equivalence to the ellipsoid visualisation is complete. This view makes more explicit what

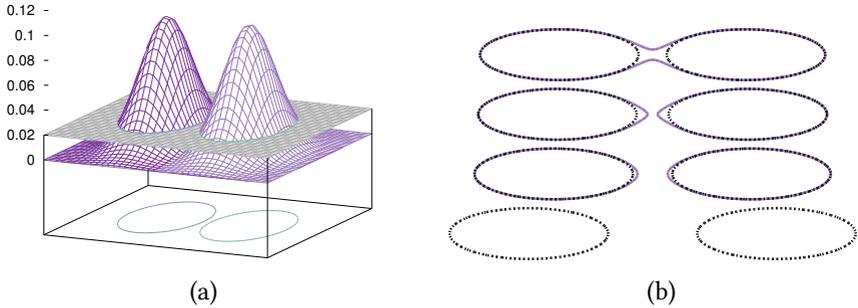


Figure 2.3: Blending implicit surfaces. Left: Twice, next to each other, the 0.2-level set of a 2D squared exponential function. Right: projection on the plane. The individual objects' contours are dotted, the blended contour is solid. While moving to each other the objects start to fuse before their original contours touch.

sampling from a GP means: to sample a point from each 1D distribution. An according to a GP distributed randomly drawn sample is roughly an infinite-variate Gauss distributed vector.

In this work, we use Gaussian processes as an ingredient in a probabilistic geometric object representation which is based on implicit surfaces.

2.2.4 Implicit surface

An *implicit surface* is implicitly described as a level set of a function: $S = \{\mathbf{x} : f(\mathbf{x}) = c\}$. Using implicit surfaces to represent shapes started as a method to visualize 3D models of molecules in the early 80s' computer graphics research (Blinn, 1982). The constant c is often taken positive and the function is mostly zero and rises only to form the object, like in Figure 2.3a. Then, blending, for instance, can be done by adding the functions which describe the two implicit surfaces. Observe in Figure 2.3b that the shapes connect before their original boundaries touch – due to the superposition of the functions near the objects exceeding the watermark. This effect corresponds to the way molecules

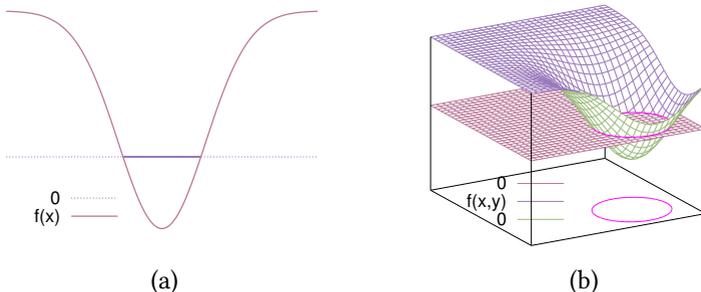


Figure 2.4: Implicit surface examples. An 1D object is a segment delimited by its surface (endpoints). A 2D object is subset of the plane delimited by a contour.

interact and is intended in this case, but could be undesired for other applications and then should be compensated for. Related to our shape estimation approaches we discuss later, e.g. by Williams and Fitzgibbon (2007); Steinke et al. (2005), use $c = 0$ and the positive-inside-object definition. Lots of shape deformations can be described analytically on functions rather than on the shape contours (Steinke et al., 2005), which makes implicit surface appealing in many contexts.

In this work, we use the zero-level set, i.e. the shape is the set of all points which have a value of zero. The function that describes a particular d -dimensional object is of the form:

$$f : \mathbb{R}^d \mapsto \begin{cases} < 0, & \text{inside} \\ > 0, & \text{outside} \\ 0, & \text{on the surface.} \end{cases} \quad (2.10)$$

Figure 2.4 shows examples of our implicit surfaces and the corresponding functions. In a one-dimensional world, the objects are segments of the line and are determined by their endpoints – this is their surface (Figure 2.4a). Two-dimensional objects (Figure 2.4b) live on a plane and are delimited by

a contour. This contour is where the function crosses the zero plane. Note that a function describes a single object, but there are a lot of functions which describe the same object. The sign inversion in Formula 2.10 compared to the usual definition has to do with our interpretation of f as the potential given rise by the object (Chapter 3).

Chapter 3

Gaussian process implicit surface potential

In this chapter we first review various ways to represent the shape of an object and point out their respective advantages and comment on their suitability to different application domains. Then we formulate requirements in the context of object estimation and grasping under uncertainty and eventually propose a representation: a machine learning tool to handle uncertain inputs for approximating a function which implicitly defines a shape, this function being interpreted as potential field generated by the shape.

3.1 Representations

We use the term representation to describe the way we systematise the information about a phenomenon: the miscellany of sensor readings originating from the phenomenon becomes a representation, we believe, only after it is put in a context – in a meaningful frame – and what matters is extracted and reasonably recombined. In the course of studying the mind cognitive sciences have varying viewing angles and see different levels of detail which

determine the very different representations they use. Some computational approaches to intelligence describe the world in terms of symbols, facts, rules – this is the traditional, symbolic AI research (Brooks, 1990). Their representations capture abstract properties of the scenes, they are less concerned with individual objects. Other efforts in understanding cognitive processes speak of affordances (Gibson, 1977) as the perceived entity which carry information about an object (Sinapov and Stoytchev, 2007). Connectionists have different approach to computationally modeling the mind: they see cognitive processes as emerging from networks of neurons (Elman, 2005). The structure and activations of a neural network can be viewed as representation of the information relevant for the cognitive task. Other neuroscientists approach the mind by looking into the very machinery: they measure activities in brain regions or single neurons and refer to the observed patterns as representations of cognitive states or tasks (Murata et al., 1997; Glover, 2004). On the end of man-made ones, predictive state representations (Singh et al., 2004) see the state as the belief about all possible futures. Investigating hierarchies of actions can come up with aggregated actions which are more suitable to reason with than the lowest level control signals are, e.g. options (Precup et al., 1998). Clearly, all these examples address different problems and contribute to different aspects of the principles of intelligence.

For someone with background in programming and software engineering it is easy to think of a representation as the data structure used by a computational algorithm. Knowledgeable computer scientists argue convincingly that programming is mostly about designing the data structures. From the proper data structure the algorithms emerge naturally. (Pike (1987) introducing his implementation of `sam`: ‘The presentation centers on the data structures, because that is how the program was designed and because the algorithms are easy to provide, given the right data structures.’ And again Pike (1989), more vigorously: ‘Rule 5. Data dominates. If you’ve chosen the right data structures and organized things well, the algorithms will almost always be self-

evident. Data structures, not algorithms, are central to programming.’) This applies also to representations in the sense that the organisation of the information suggests very much the actions that the system performs with it. Thus, it makes more sense to regard representation as the whole concept of information organisation and way it is used. In terms of a classical RL setting, for instance, this means that the agent’s internal state and action together make up the representation, i.e. the relevant world information and the relevant ways to change the world.

The motivating RL example from Section 1.5.1 insists to care more about representations. The problem can get even more complex: highly autonomous agents acting in changing environments most likely will find themselves solving very heterogeneous tasks. Properly dealing with these will likely demand agents to adapt or even bootstrap (invent) state and action representations themselves. So, where do s and a come from? Apart from hand-crafting them for specific problem or copying them from nature, recent representation learning or symbol learning efforts (Jetchev et al., 2013) come up with a scheme to find a s which is optimal for fulfilling a given task (encoded in the reward). Loosely, our contribution can be motivated from this perspective as investigating the relations between s and a and hand-crafting ones. We contribute a representation (think s) which has a simple interface to the sensor input and at the same time is well suited to design reflex-level or affordance-level primitives (think a). For a we do not go so far as to bootstrap a hierarchy from experience; we modestly handcraft several heuristics for manipulation tasks which use the designed s and turn out to resemble grasp stability criteria like force closure. Our s is not formally shown to be optimal for a given task. Its virtue is that it plays well with the a , it is able to capture grasp relevant geometrical features and incorporate priors and multiple cues.

In the context of interaction with the physical world, the manipulable entities which form the world are the physical objects. Thus, describing contexts, tasks, etc. is best done by reasoning about objects, i.e. using object represent-

ations. Despite the above argument about the scope of representations, for clarity's sake, we first consider object representations as a way to organise information about objects, comment on their properties and outline differences. Then in the next chapters we look at the methods which use this particular choice of relevant data and its organisation.

3.2 Related work: geometric shape representations for grasping

We do not understand completely the mechanisms of dexterous grasping. There are plenty of factors which are important for a successful grasp. They give rise to various heuristics and representations.

3.2.1 Fitting and composing primitives

The shape determines to large extent what configuration of the fingers can stably hold it – especially for rigid objects. One way to think about the shape is to see it as belonging to some ‘regular’, well understood class and find the particular parameters for that shape. The most obvious candidates for this are abstract shapes which the human learned to produce effectively in the history, like sphere, cube, cylinder. Although such pure forms do not exist in nature, they seem to approximate a large number of important natural objects. A more powerful set of forms, including arbitrary good approximation of the above mentioned, can be covered by superquadrics (Barr, 1981). They extend quadratic surfaces through spherical products of curves to larger class with well understood structure. The parameters of the superquadrics determine the roundness/squareness along each dimension, leading to variety of flavours of ellipsoids, tori, hyperboloids. Figure 3.1 shows some non-trivial examples. Note that all shapes in the lower row have a hole through which one can see along the z -axis. Operations like union and difference of volumes allow for

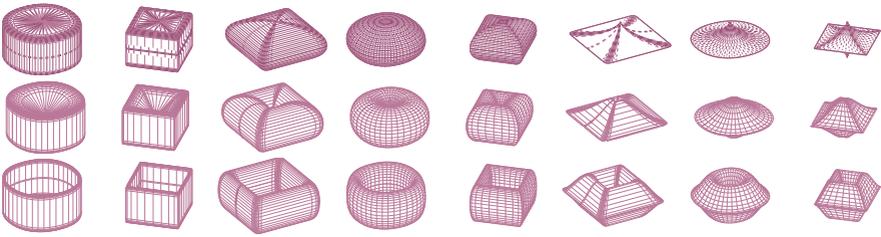


Figure 3.1: Examples of superquadrics. Thinking of superquadrics as modified tori, these plots result from varying three parameters: the roundness of the two circles of a torus separately and the radius of the hole of the torus.

flexible design of additional shapes. The main advantage of superquadrics is their compactness – just a few form parameters, scale and orientation – as well as the available in closed form and easily computable surface normals, tangents, and higher moments.

With this variety a large number of objects can be fit reasonably well, but of course, by far not all. Furthermore, they put a bound on the precision of the representation – one cannot give up some compactness for the sake of precision. To achieve this, a general pattern is to glue primitive shapes together to a model which resembles the original shape. Note the difference between modelling the surface (as with a polygon mesh) and the volume of an object (e.g. with voxels). For deformable objects and for applications which need to reason about deformations the volumetric representations might be much more preferable, since they can model the interaction between the inner elements of the object. For instance, superquadrics as primitives can be joined together to approximate complex shapes, as do Zha et al. (1998). They subdivide poorly fit parts and adjust a new superquadric to the subparts recursively, successively decreasing the residual error.

Instead adaptively fitting primitives, one can go for a single primitive shape, say, a cube. This would give rise to a 3D occupancy grid which density can be varied to achieve certain quality of approximation and the corresponding computational effort. At first sight, giving up the flexibility of super-

quadrics may not seem attractive, but the regularity of a cube opens a very interesting possibility for storing and accessing the collection of primitives. Rather than enumerating linearly the cubes (also referred to as voxels, in analogy to pixels in planar images), they can be placed in an octree – a tree, each node of which is either terminal or has exactly 8 children. Each child corresponds in 3D naturally to $1/8$ of the volume its parent takes. The octree encoding was proposed by Meagher (1982) together with efficient algorithms for manipulation of so represented objects, e.g. orthogonal transformations, calculating interference between objects, calculating visibility of surfaces, center of mass, etc. The tree structure makes the addressing of a region very efficient by, on each node, partitioning the space in two halves along each axis. Furthermore, the hierarchical organisation preserves the local structure – nodes representing neighbouring volumes are stored in neighbouring branches of the tree. Octrees allow to trade computational efforts off against precision: a node either describes the occupancy of its volume sufficiently well – as occupied or free, and is then a terminal node – or it is marked partially occupied and subdivided into its eight children recursively.

3.2.2 Polyhedrons and patches

The above kind of representations are interesting for applications which by design deal with volumetric data in the first place, e.g. fMRI scans. However, our senses – touch, sight, hearing – deliver information from the surface of an object, i.e. surface data. The information about the inside is usually secondary, inferred. A polygon mesh is a way to discretise the contour of an object, as opposed to voxels which do so with the volume. The smooth surface is approximated by piecewise linear patches, forming a watertight mesh. It can be used for visualisation with planar primitives, for rendering texture, for intersection and collision tests. Again, the more resources available, the better fit can be achieved. An existing triangle mesh can be refined by repeatedly splitting an

edge in the middle and thus building two new faces until the maximal edge length of the mesh goes below a threshold. Figure 3.2 shows the approximation of the Stanford bunny improving with the increasing number of nodes (and thus faces) of a triangle mesh. Compared to a point cloud, a mesh offers richer information by stating how the points are connected. It induces also a piecewise constant tangent and normal approximations: the normal and tangent do not change along a face; at a node the average of the adjacent faces can be used. If the normal is not part of the data, a face defines the normal axis, i.e. the normal vector up to a sign. Resolving this inside-outside ambiguity might be hard in some circumstances; achieving agreement on that question among all faces is relatively straightforward, though. Meshes can be used to extract more features about an object, for instance Pokorný et al. (2013) compute topological properties like the existence of holes.

A different flavour of polyhedral representations – convex and with heterogeneous faces – is used by Caselli et al. (1996). They are produced from tactile data for object recognition purpose by building separating planes. The authors note that such representations are central to haptic sensing because of the very nature of haptic data: it is inherently sparse and 3D.

Another example of polyhedral representation is the one used by Joshi and Sanderson (1993). They want to match registered contacts to existing object models rather than build a model. Thus, they are not so strict about convexity – at the cost that the description does not fully determine an object; it rather serves to discriminate. By using a minimal representation size criterion they solve jointly (i) the correspondence of model to sensor feature problem and (ii) finding the most appropriate pose of the model. Finally they use an interesting way to generate candidate poses from partial matches.

Other discrete surface approximations use loose planar patches, surfels (Andersen et al., 2010). The surfels do not connect rigidly, their relation to each other is modeled in this case by a Markov random field which carries the surface prior across multiple surfels. Since they are not so firmly constrained

by their neighbours, sometimes the perceived local approximation of the surface is better than a mesh for the same computational effort. Their non-connectedness is a drawback for applications like intersection tests, precise distance queries and some physical interactions. However, for visualisation purpose the smoother nature and computational advantage can be appealing.

3.2.3 Visual and sub-shape representations

Geometric and related to shape and to visual processing is also the representation used by Calli et al. (2011): it reduces the shape to a 2D projection, the contour of an object as seen by an in-hand camera. They approximate the contour by low harmonics and obtain a smooth curve which they can reason with about convexity and extract interesting points from. Note that this representation is strongly linked to the used sensor type and how it is attached to the body: the view changes smoothly when moving the viewing point. This is something one exploits in visual servoing approaches; Calli et al. use it to come up with more advantageous approach direction when grasping.

As to grasping, shape motivated representations do not necessarily need to have the notion of shape as holistic geometric object. This is evident from the following example. Even deprived from our sense of touch, we would use redundant visual information and likely successfully grasp. This suggests that mere visual information could in many cases be rich enough. Saxena et al. (2008) use direct visual cues for their grasping algorithm. They refer to neuropsychological evidence (Goodale et al., 1991) that recognising the form and location of an object is dissociated from the ability to grasp it. This suggests a shortcut which avoids the mental compilation of perceived features of the object into a rich representation of identity, shape, position and orientation, and directly come up with a grasp from more primitive features: edge and texture filters of a patch and its neighbours and take care of different scales. From these features Saxena et al. predict whether a patch contains a grasp-

ing point. The learning algorithm takes synthetically generated scenes and the labeled patches and computes the maximum likelihood parameters for the prediction. In this case, the features and the parameters actually constitute the representation.

Another example for pre-shape (sub-shape) representation for grasping motivated by developmental psychology are the early cognitive vision descriptors (ECVD) (Pugeault, 2008). They are used by Kroemer et al. (2010) to represent whole scenes – the target object and obstacles – and carry the information needed to learn to imitate demonstrated grasps and associate them to scene configurations.

3.2.4 Implicit surfaces

Discretisation does not necessarily make things more simple. Handling the surface of an object as a single entity can be an elegant and efficient choice for some applications. Using implicit surfaces (we introduced them in Section 2.2.4) to represent shapes gained popularity in visualisation of 3D models of molecules in the early 80s (Blinn, 1982). There are a variety of analytical algorithms for deformation and blending (Bloomenthal, 1995). Also in the Machine Learning community, implicit surface shape representations have been investigated, e.g. by Steinke et al. (2005). Depending on the class of functions used, they can provide efficient analytical computation of normals, tangents, and other surface features. Since they only implicitly contain the surface, some other applications require more efforts. Drawing samples from an arbitrary surface – this is essential for visualisation or meshing – requires dense discretisation of the space and probing. With the proper assumptions about the function and surface, more efficient approaches can be used. Here we use implicit surfaces because they allow us to translate shape assumptions into function assumptions and reason about them in our approximation and grasping methods.



Figure 3.2: Discrete shape representations.

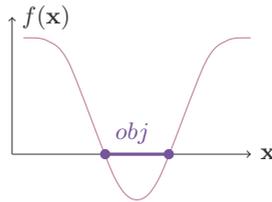


Figure 3.3: Implicit surface.

3.2.5 Summary

Geometric representations can be seen as volumetric and as surface-centered. They include polyhedral surface approximations, simple and complex primitives fitted to surface points, composition of primitives, implicit surfaces, but also representations based on purely visual features.

Calli et al. (2011), Pugeault (2008) and Saxena et al. (2008) are tied to vision and let down various other modalities. The reviewed polyhedral and other surface representations are not well suited to deal with uncertain information. They expose the surface explicitly and thus are suitable to test grasp stability criteria (discussed in Chapter 5) or visualise, however, do not shed enough light on the choice of contact points and the grasping trajectory. Implicit surfaces as introduced for computer graphics (Blinn, 1982) suffer the same issues with uncertainty but wrapping them around with a learning method helps, as Steinke et al. (2005) do. With a probabilistic method like GP they are capable of conveying uncertainty (Williams and Fitzgibbon, 2007). As discussed in Chapter 4, both above references use a hack to learn from essential sensory

information – surface normals (Walder et al. (2006) cure this in the SVM world, while we use the very natural way of GP to handle normals). All three latter references are focused merely on shape approximation, rather than sensory-motor grasping and manipulation which the present work addresses by the potential interpretation of the implicit function (related potential approach lacking the uncertainty exposure is discussed in Chapter 5 and 6).

3.3 Requirements

Object manipulation involves two directions of information processing: sensory feedback and motor control. The above mentioned shape approximations and cues do not fully account for this. A good representation has to (i) provide means for incorporating all available information: prior shape knowledge and heterogeneous uncertain sensor feedback from different modalities need to be fused in a coherent way; (ii) the resulting shape belief needs to be in a form convenient for the synthesis of a reach-and-grasp motion. These requirements on representation are motivated also by research about the human sensory and motor system from the fields of psychology and neurobiology. Intraub (2010), for instance, suggests that pure unimodal sensory representations alone cannot explain boundary extension in human scene perception. In terms of cognitive neuroscience/neurophysiology the requirements are justified by findings that *premotor* areas in the cortex are equipped to process sensor information and (*sensory*) posterior parietal areas play *motor* control role (Gallese, 2005). Thus, neuron complexes exist that (i) jointly process multi-modal sensor information (tactile and visual), (ii) are specific to objects and locations in space, and are in addition (iii) coupled to the motor system (Gallese, 2005) – in more robotic terms, a sensory-motor object representation with its own frame of reference.

Concerning the shape estimation part, we aim at integrating different sensor inputs, for instance, the methods have to be able to deal with haptic,

visual and laser feedback. Visual and laser sensors provide uncertain 3D coordinates of points on the surface of possibly far objects, with laser typically being more precise than 3D vision (based on stereo triangulation of key points) (a survey on stereo vision algorithms has been compiled by (Scharstein and Szeliski, 2002)). Haptic sensing can be more precise than vision, but is useful only with objects which are in reach. In addition to the position of a surface point, haptic feedback provides also an estimate of the local tangent plane of the shape at the point of contact. The above three examples illustrate the need to have a representation that allows to fuse the heterogeneous multi-modal sensor information in a shape estimate.

Commonly, the information about objects comes at different levels of confidence. Fusing it requires to have means to deal with sensor uncertainty in the representation. Machine learning provides a variety of instruments to handle noisy input. Example applications to shape approximation algorithms include the Support Vector Machine (SVM) approach by Steinke et al. (2005); in a similar setting, Williams and Fitzgibbon (2007) employ a Gaussian Process (GP). A GP (Rasmussen and Williams, 2006) takes inputs, possibly with different noise distributions, and produces estimates with error bars. The above works prove the suitability of established ML tools for object estimation – a sane estimate is necessary for the successful interaction with the environment – but deal with uncertainty merely in passive manner. Dealing actively with uncertainty requires to integrate sensing and motion and have a representation which supports the interactive acquisition of information following the Bayesian rule – constantly updating posterior belief – and making it accessible for motion generation.

To the two domains which motivate GPISP – estimation and grasping – we dedicate the following two chapters. We explain in detail in Section 4.2 how sensor information (from vision, laser or tactile) can be translated to a Gaussian process estimate consistently fusing the modalities. And the motion generation is discussed in Section 5.3.4.

3.4 Glueing GP, IS and P to GPISP

The following concepts add up to GPISP: *Gaussian process*, a non-linear ML formalism, estimates from sensor data a function which implicitly defines the surface of an object as its zero-level set, an *implicit surface*. The estimated function is so designed that it can be interpreted as a *potential field* generated by the object – monotonously increasing with the distance to the surface. With this interpretation, the value and direction of the potential field can be used to navigate an end effector towards the surface and specify its relative orientation to the object.

3.4.1 Gaussian Process

No matter what particular way we choose to describe the shape, we face the following problem: we need to predict the shape of an object only from limited evidence. The evidence is in form of noisy observations of only some points from the shape and no direct evidence is available for the rest. The task is to consistently (i) predict the shape where no evidence exists and (ii) correct/align the sensor measurements where they exist. This sounds like the classical problem of regression, i.e. find a model which explains the measurements and can be used to approximate beyond them. Every method which is regularised or has some prior on the space of possible models already deals with uncertainty by assuming the observations can be relaxed. Here we want to go beyond that and explicitly know how reliable are the predictions at different points. Any method which provides this information should be good to go with. However, some might be more convenient to use:

While both *Gaussian process* and *Support vector regression* (SVR) are popular with machine learners for estimating nonlinear functions, the former provides at arbitrary point \mathbf{x} together with an estimate $f^*(\mathbf{x})$ the variance of the estimate, $\mathbb{V}[\mathbf{x}]$ as part of the vanilla definition. This is the case because it in the first place is thought as a probabilistic method which predicts a distribu-

tion rather than a value. We need to note here that for SVR exist methods to give an estimate of the variance, too, using bootstrapping or by a probabilistic interpretation of the SVM kernel, as done by Sollich (2000).

The mean of the Gaussian process is its best guess when it comes to predict the value at some point. Formally, it is the maximum a posteriori (MAP) estimate of the GP conditioned on the observations. In the light of available data points the method can be differently confident at different positions. Near an observation the variance would be typically low compared to points in regions without observations. We take the variance as a quantification of the uncertainty about the estimate.

The view on possible means of predictive distributions $P(y^* | \mathbf{x}^*, \mathbf{x}_{1:N})$ as a map $\mu(\mathbf{x}^*)$ from any \mathbf{x}^* to the corresponding estimate, one can interpret GP as extension of Gaussian distribution for functions, and can write $f \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. Thus, a GP is a probabilistic estimate of a function from evidence at individual points.

3.4.2 Implicit surface

Now assume the function to be estimated describes an object; then, by estimating the function from evidence, we estimate an object. *Implicit surfaces* (introduced in Chapter 2) are described by a level set of a function. If one considers the 0-level-set, an object is given by

$$f : \mathbb{R}^d \mapsto \begin{cases} < 0, & \text{in} \\ > 0, & \text{out} \\ 0, & \text{on} \end{cases} \quad (3.1)$$

This is the class of functions we approximate using a GP. The representation as 3D implicit surface abstracts non-geometrical features which either do not influence grasping, e.g. color, or go beyond the approach we take in this work,

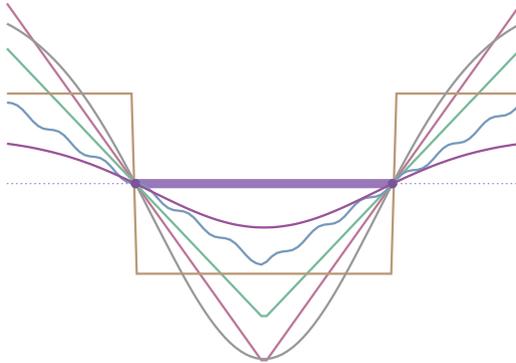


Figure 3.4: Many functions (slim lines) representing implicitly the same 1D object (bold line) by its surface (end points of the bold line).

e.g. friction, material, relations to other objects.

Note that any function which crosses the zero hyperplane exactly at the surface points defines implicitly that same surface. In other words, there are multiple (in fact infinitely many) functions which represent the same object. Figure 3.4 illustrates this. For instance,

- a staircase function, $f : f(x_{out}) = 2, f(x_{in}) = -2, f(x_s) = 0$, taking a constant positive value outside, and constant negative value inside an object is loosely speaking an extreme case of having a non-smooth function;
- another example is the modulus function shifted to cross the zero line at the appropriate points: it is smooth up to points inside the object. The function value has a nice property of being proportional to the distance to the object, which the first example lacks.

The choice of the exact class of functions we estimate by a GP is determined by the further application of the object representation: we are interested in generating motion for grasping. Important features for producing smooth grasping trajectories are the distance to the object, and smoothness of the function so

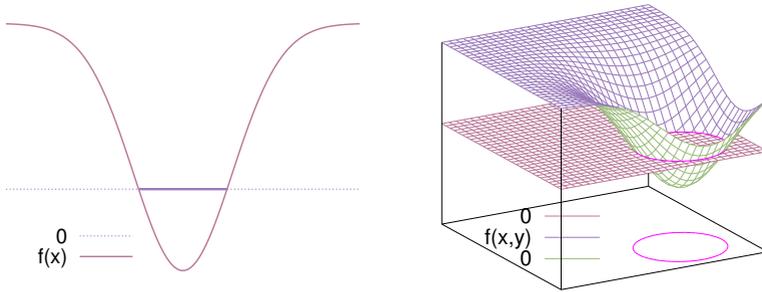


Figure 3.5: Objects represented by an implicit surface function. Left: 1D object on the zero line; right: 2D object described by $f(x, y)$: the ellipse in the plane is the surface.

we can use it to modulate the motion features. The metaphor of a particle moving in a potential field, attracted by its goal position and repelled by obstacles has been often used in mobile navigation. Among all implicit surface functions, we opt for a family which resembles a potential field, in particular, in addition to Formula 3.1 we restrict the functions to (i) have certain positive bias, $\mu > 0$, in the absence of data – to reflect the lack of knowledge about existing object there (positive outside object); (ii) smoothly and gradually decrease to zero on the surface of the object. Both requirements we can accommodate in the GP hyper parameters: smoothness of the estimate is determined by the covariance width, and μ is the bias which needs to be added/subtracted upon evaluation/training to the value of an otherwise zero-centered GP.

Figure 3.5 gives two examples of objects represented by the 0-level of a function which monotonously rises with increasing distance to the surface, bounded by some positive bias μ_0 . Note that in this class of functions there are infinitely many which result in the same shape, too. For a fixed μ_0 , they differ in how steep they descend to zero. The steepness can be controlled by the proper choice of parameters of the GP, in particular the width of the covariance. In spite of this variety all of them share the properties which allow to interpret them as potentials.

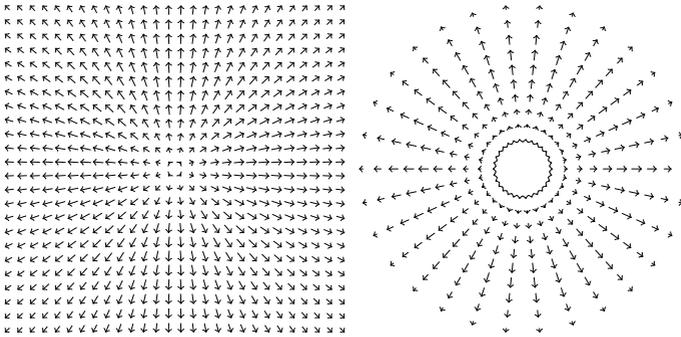


Figure 3.6: Vector field associated to a planar circular implicit surface as in Figure 3.5, right.

3.4.3 Potential

The estimate and its negative gradient $-\nabla f(\mathbf{x})$ can be thought of as a field, for instance similar to the gravitational potential. The analogy to gravitation plays a crucial role in the design of the reach and grasp motion controller based on the GPISP. Figure 3.6 shows the vector field induced by the two-dimensional object from Figure 3.5. The GP mean function can be thought of as an approximation for the distance to the object in its vicinity. (Further away, the estimate flattens out infinitely to μ_0 and the approximation is very crude. Later we will see that from controller's perspective this is just right because near the object it needs the precision while away it is enough to know that there is still way to go until one reaches the object.) At each point \mathbf{x} the gradient of the GP mean points away from the surface of the object, equivalently, the negative gradient points towards the surface – this is more explicitly illustrated on Figure 3.6, right, where the gradient is drawn at iso-lines. As in gravitation, in the absence of other effects, a particle put under the impact of the field would follow the gradient to locations with lower potential, eventually reaching zero potential on the surface. The magnitude of the gradient has a single peak between a local minimum (inside the object) and local maximum (far from surface). The course of the magnitude on this segment depends on

the GP hyperparameters and the observed data. It also can be used to modulate some motion features, but we employ mostly only the direction. This has the advantage of being not too sensitive to varying parameters.

3.4.4 Requirements revisited

Now we can again look at which characteristics are important for a representation to qualify as suited for grasping and multi-modal sensing under uncertainty and ask ourselves how we make sure they are present in GPISP.

For dealing with uncertainty we have the variance of the GP on our side: it reports for any point how confident the mean prediction is, how strongly the prediction is supported by available data.

Multi-modal sensing means to consistently fuse sensor readings from different modalities so that the estimate benefits from all available data. Different sensors imply different noise distributions and raw data representation. With the GP we assume Gaussian noise, but we can vary the noise variance from data point to data point. In Section 4.3, when we elaborate on estimation, we provide also the data interface to our representation: surface points and normals. Then multi-modal sensing amounts to preprocessing raw data to points and normals and attaching the proper noise variances.

For motion generation we interpret the estimated function as potential and use it in grasp controllers naturally to attract and orient actuators. Furthermore, to make a controller aware of uncertainty, we use the whole distribution over shapes – i.e. the GP variance around the MAP implicit surface.

An arbitrarily good continuous shape approximation is possible, however on the cost of more data. The prior on surface smoothness (the complexity of

the function class) determines the trade off between ability to generalise from few data points and the precision of the estimate.

Previously available knowledge can be incorporated in form of priors or bias: (i) the covariance function imposes a desired level of smoothness; (ii) instead of constant bias μ_0 , we can estimate on top of a $\mu_0(\mathbf{x})$, e.g. an implicit surface representation of a sphere, and thus with less observations attain more precise estimate.

3.5 Parameters

The conceptual intersection of GP, implicit surface and potential is a matter of identifying hyperparameters for the GP for the task at hand, hardware capabilities and environment.

3.5.1 External and internal parameters

In Machine learning, one usually would seek to infer automatically the parameters of the model from data so that the model accommodates the evidence best. However, the resulting too complex models often witness over-fitting and regularisation is applied to avoid them. When there is no clue about the true nature of the data, the only way to find optimal regularisation and parameters is to try out different combinations and cross-validate or marginalise in other ways the influence of the current training sample. Though, we are not clueless in that matter: the embodiment can advise the system which models are worth looking at. The hints are in some cases directly transferable to the free parameters, i.e. some of the hyperparameters can be thought of as resulting from the own body, environment and task specification.

3.5.1.1 Covariance width

Technically, the width of the squared exponential covariance function σ_w^2 is a hyperparameter of the GP. It determines how strong the influence of an observation decays with growing distance. As a result, larger σ_w^2 make the estimate smoother, and small ones allow for more jumps. (Using RBF kernels corresponds to penalisation of the derivatives of the function in terms of Tikhonov and Arsenin (1979) regularisation (Smola et al., 1998).) This translates to the smoothness of the zero-level set, i.e. to the shape of the object, so that σ_w^2 controls the detail size of the estimate. Another point of view on σ_w^2 is as the parameter used to trade accuracy off for compactness of representation. Large detail size means that a single observation already influences large neighbourhood, i.e. other observations in this neighbourhood are redundant and can be omitted. But if the prototype shape has fine structure, the large covariance width forces the estimate to roughly average the near observations (fine details give rise to larger variability in near observations, so they are not redundant), and the estimate becomes imprecise. On the contrary, with small σ_w^2 the GP needs a lot of observations even for boring large similar areas.

The above can be considered translation of the statistical learning arguments of model complexity in terms of implicitly represented object shapes and relates the data and the outcome. The other perspective is rooted in this thesis' commitment to a wholistic view on sensing and manipulation, a view which puts them in each other's favour and considers aspects of the coupled task of manipulation and sensing: then, what needs to be sensed for is what is relevant to the embodiment, and a quite good example of this relation is the detail size of an object. For the Schunk hand, which has 2cm wide fingertips, sensor arrays with several millimeters spatial resolution, there is hardly a meaningful task sequence which includes picking a needle from the table top, neither moving a huge stone. Therefore, it needs not to sense for such kind of objects, or at least not on that scale. This results directly in a prior over σ_w^2 ,

and the more specific the task becomes, the more narrower the prior, possibly to the extent of collapsing the prior to a single value which works for most expected tasks.

3.5.1.2 Observation noise

The observation noise, too, can be tuned so that the data is explained best by the model. But in a real setting it makes more sense to see the observation noise as data rather than as hyperparameter. Observation data comes from sensors which operate in an environment. Taking into account the environment and the intrinsic parameters of the sensors, one can estimate the sensor noise for the particular data point. Here we assume that the noise distribution is Gaussian centered around the true value, and its variance represents it sufficiently. Generally, a data point in terms of sensing consists technically of multiple data points: (i) surface point and (ii) surface normal which itself consists of three partial derivative data points. Therefore, for best performance, attention should be paid to find the proper noise variance for each sort of observation.

In fact, we are somewhat cheating when subsuming the imprecision of a sensor into the response noise. For most sensors it is rather the point that is uncertain, not the reading. Girard et al. (2003) look deeper into the problem of how input noise translates into posterior variance and we refer to them for details.

3.5.1.3 Bias and prior variance

As we already argued, one can see a GP as Gaussian distribution over functions so that $f \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. Usually we uniformly bias the GP in GPISP by a positive constant to reflect the fact that in absence of data it is not aware of the location and shape of the object. Without data, $\mu(\mathbf{x}) = \mu_0$ and with coming evidence the posterior mean evolves. The covariance of the distribution is

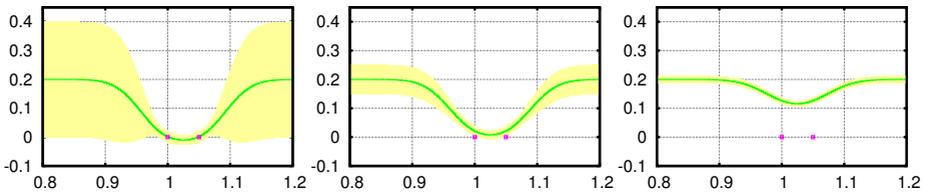


Figure 3.7: Choosing the prior variance too small trusts too strongly the initial bias and can prevent the data to speak.

given by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_p^2 \exp\left(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^T (\mathbf{x}_i - \mathbf{x}_j)}{2\sigma_w^2}\right).$$

From Section 3.5.1.1 we already know the effect of the exponential decay. In addition, the leading factor σ_p^2 in the covariance is the prior variance of the GP, i.e. the variance of the function without data. It quantifies how far one trusts the initial mean. For instance, if it is set too narrow, the observations will not be allowed to pull the mean down and effectively the evidence will be ignored. The prior mean and variance depend on each other in the sense that the variance must not restrict the posterior mean to go below zero. In case the initial bias is chosen non-stationary in order to reflect prior knowledge about the shape, e.g. $\{\mathbf{x} : \mu_0(\mathbf{x}) = 0\}$ is a cylinder, this has the effect that observations are still allowed to contribute to the object shape. In contrast, if the cylinder evolved from sensor experience, the areas around observations would typically have low variance, and additional observation cannot change a lot. The plots in Figure 3.7 illustrate this effect. Most left, the variance is the quadrat of the bias and allows the mean to comfortably be bent and cross the zero at the observed points. On the other two plots, the GPs are forced to more strictly adhere to the 0.2 bias. Effectively, they do not trust the observations enough to reach the zero. They do not result in an implicit surface.

3.5.1.4 Derivative observations

The representation is incomplete without saying how information about the normals of the surface can be used. We mentioned in Section 2.2.3 that GP can be conditioned on derivative observations. In these terms, the normal is in fact the array of partial derivatives at the surface point. To incorporate them in the covariance vector \mathbf{k} and the Gram matrix \mathbf{K} we need the derivative of the covariance function, as noted by Rasmussen and Williams (2006, p.191). Using $\mathbf{x}_i, \mathbf{x}_j$ as the observed points; d_j as the index of the component in which the derivative $\frac{\partial f(\mathbf{x}_j)}{\partial x_{d_j}}$ is observed, δ_{ij} being Kronecker's delta and $k_G(\cdot, \cdot)$ the RBF covariance, we get:

$$k\left(f(\mathbf{x}_i), \frac{\partial f(\mathbf{x}_j)}{\partial x_{d_j}}\right) = \frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial x_{d_j}} = \frac{(\mathbf{x}_i - \mathbf{x}_j)k_G(\mathbf{x}_i, \mathbf{x}_j)}{\sigma_w^2},$$

and

$$\begin{aligned} k\left(\frac{\partial f(\mathbf{x}_i)}{\partial x_{d_i}}, \frac{\partial f(\mathbf{x}_j)}{\partial x_{d_j}}\right) &= \frac{\partial^2 k(\mathbf{x}_i, \mathbf{x}_j)}{\partial x_{d_i} \partial x_{d_j}} \\ &= \frac{(\sigma_w^2 \delta_{ij} - (\mathbf{x}_{id_i} - \mathbf{x}_{jd_i})(\mathbf{x}_{id_j} - \mathbf{x}_{jd_j}))k_G(\mathbf{x}_i, \mathbf{x}_j)}{\sigma_w^2}. \end{aligned}$$

In Appendix A.2 we derive all covariances needed for the features we construct with GPISP. These include evaluating the GP mean, its gradient and Hessian, as well as the variance and its gradient.

3.5.2 The resulting implicit shape prior

Our Gaussian process prior implies directly a prior over shapes. This prior depends on the choice of hyper parameters for the covariance function and bias, as considered above. In the following, we illustrate the resulting shapes in a 2D world. Since the definition of an implicit surface is invariant to scaling of function values, we can – without loss of generality – fix the GP bias to

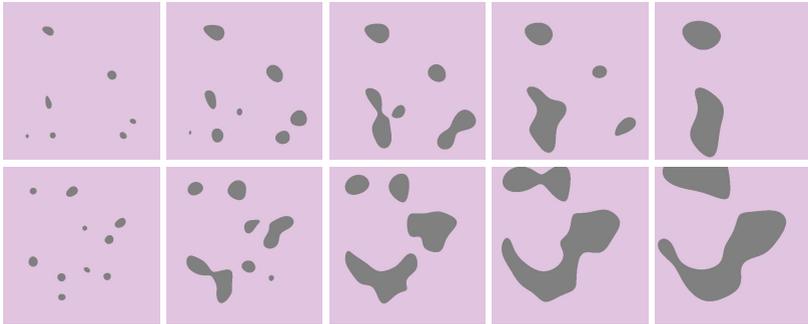


Figure 3.8: Random samples from GP with 2D domain. Sampled functions are produced with varying covariance width, σ_w^2 , rises from left to right in both series. Functions are cut through the 0-level set and positive values are light colored, negative – dark.

$\mu = 1$. The resulting shape prior is then influenced by the covariance width σ_w^2 .

Figure 3.8 illustrates in 2D the shape priors (and also the implicit prior of how shapes are distributed in space) depending on σ_w^2 . It shows a series of randomly sampled functions from a two-dimensional GP. They are produced by sampling random observations and plotting the resulting predictive mean. Colors code positive (light) and negative (dark) values of the mean. The contours therefore, mark the surface of the objects. Note that the images in each row share the observations and parameters, apart from the covariance width which increases from left to right. One should imagine the function as a surface, cut at the zero-level, i.e. dark regions say ‘under water’, light regions saying ‘on the ground’.

- In statistical learning terms, low σ_w^2 correspond to complex models, making the method generalise very carefully, in a small neighbourhood, forcing it to consider individual observations virtually independent.
- Large width implies simpler models which are harder to influence by individual observations. With growing σ_w^2 , the notion of neighbour-

hood grows, allowing observations to influence larger areas, resulting in more areas being categorised as part of an object. The blobs become smoother and larger. For the highest width, the upper row shows that the left-part observations influence goes that far that they even inhibit the right part because the model does not manage to bend strong enough to accommodate all (and probably ‘overshoots’).

Back to our argument on embodiment, here we see that for manipulation it makes sense the resulting blobs to be roughly the same size as the actuator. And the notion of ‘near’ (or ‘neighbourhood’) intuitively correctly grows with larger objects. To make it clearer, imagine in each plot one of the blobs as an actuator or as a mobile robot. Then the distance relevant to collision detection or to safe navigation to the surface of other objects roughly matches the size of the blobs.

For the sake of simplicity, we assumed here the same noise distribution for all observations, σ_n^2 (can be thought to reflect the precision of the used sensors). Furthermore, as commented by Ohtake et al. (2005), sensors may have different precision along different axes. In our case this translates to a GP with multivariate noise in the input dimensions. Here we simplify to isotropic noise, too.

Note that in this thesis we consider squared exponential covariance only. Possibly, other covariances constitute different shape priors, e.g. ones which imply non-smooth objects.

3.6 Summary

An implicit surface estimated by Gaussian process brings a lot of advantages compared to other state of the art choices for object representation for grasping. With GPISP we reason about a continuous function rather than about 3D coordinates and manipulating this function changes the implicit surface.

Conditioning a GP on shape information makes possible to translate the uncertainty of the sensory data to uncertainty of a shape, producing (roughly) a probability distribution over shapes. The mean of the Gaussian process, i.e. the MAP estimate of the function, interpreted as a potential can provide alternative motion features to a grasp controller. The whole predictive distribution, the variance as measure of uncertainty in particular, conveys the sensor imprecision to the motor level and enables uncertainty-aware control laws. Between the sensory and motor systems, GPISP does not sit in the way, it mediates, establishes common ground and promotes their integration.

The potential function being an imperfect estimate of the distance to the surface is an example of property which can be interpreted both as bug and feature depending on the application. So does the trade-off introduced by the covariance width we mentioned above – between data amount and accuracy, too. Apart from that, whatever can be considered limits of GPs applies to GPISP, too. Concerns about the computational performance of GPs (covariance vectors and Gramian need to be updated when evidence changes) usually get practical answers along the lines of lazy evaluation (i.e. take into account only locally relevant data) or substitute of multiple data by an artificial new data point summing their information. Both introduce some approximation and thus work possibly on the cost of estimation error. In our experience we cannot report a considerable delay – most of it was related to precise visualisation, which we do not consider critical. The numbers of data points in our scenarios for grasping unknown objects based on tactile feedback would hardly exceed 1000 which can be handled by our implementation well w.r.t. to obtaining motion features.

Chapter 4

Shape estimation

In Section 3.3 we pose the requirement on sane object representation to easily incorporate sensor data from heterogeneous sources to estimate a shape. In this section we elaborate how to do this in terms of GPISP after we review some existing alternatives for shape estimation.

4.1 Related work: shape estimation approaches

Building a model from observations needs to resemble the original object according to a representation and to be most informative for the purpose of the model. The estimation, thus, highly depends on the chosen representation. It determines the interface for the observed data and the preprocessing which needs to be done on raw sensor readings. In Section 3.2, we introduced different ways to represent a shape or otherwise relevant information for grasping. Here we discuss the different interfaces suggested by the representations, the ways the particular models come into being from available data.

4.1.1 From surface points to superquadrics

The minimum data that is available for reconstruction of shapes is a set of 3D surface points. Assume the object needs to be represented by a single primitive. The aim is to find the primitive which accounts best for this data. The natural objective is to minimise the amount of evidence which is not well accounted for. How can this be formalised? Zha et al. (1998) assume the data come from dense range imaging, i.e. it is available on the whole surface of the object. Their superquadrics are parametrised by 6 DoF for position and orientation of the object frame and 7 shape parameters for roundness and tapering along the axes of the superquadric. The inside-outside function of a superquadric defines an implicit surface at which the value of the function is zero. This function, evaluated at a data point, tells how far from the superquadric surface the point is, i.e. it is an error measure. Accumulating the errors for all available data points gives an objective function which is to be minimised w.r.t. the 13 parameters in order to find the best fitting superquadric. When using other primitives, a suitable distance function needs to be queried for the data points but the general pattern is the same. In the above work, the authors note that a single superquadric does not approximate well objects with parts and propose an algorithm for fitting superquadrics to subparts. It recursively splits the object in parts until the data is covered sufficiently well. The splitting is done by a dividing plane which depends on the distribution of the residuals (distances from the currently fit superquadric) and splits the object mostly at concave areas. The resulting collection of superquadrics achieves much better precision than a single superquadric can.

4.1.2 Volumetric occupancy map

A recent approach using octree representation has been developed by Wurm et al. (2010) in the context of mapping environments for mobile robots. In contrast to the originally proposed trees by Meagher (1982) which store bin-

ary occupancy, here the emphasis is on probabilistic occupancy information and more efficient memory usage. The data comes from range sensors: each light beam delivers the information that voxels on its way are empty and the one that reflects it is occupied. Each leaf of the tree holds a number which expresses the likelihood for that volume to be occupied. The number can vary in the light of encountered new measurements – this allows for fusing multiple measurements for the same region. The likelihood can vary between two thresholds which mean confidence in being or being not occupied. Once all children of a node agree and are confident they can be removed, since the parent node already carries the information. If contradictory information for a subtree arrives, the respective branch needs to be recreated. This allows to keep the memory usage low by pruning redundant subtrees. The probabilistic updates allow to explicitly model regions for which no evidence is available, rather than just declaring them free, as is the case with binary coding.

4.1.3 Polyhedron walls from tactile points and normals

The polyhedral representations employed by Caselli et al. (1996) rely on the convexity of the objects. They are designed for tactile data, in particular they have the notion of normal at the contact point. An *upper bound* model is built using tangent planes at the contact points. Due to the convexity, it is guaranteed to have the object on one side of the tangent only; thus, the plane can be adopted as a face of the polyhedron. In other words, the estimate is the space closed between all such planes. The *lower bound* model ignores the normals and employs again the convexity to note that the convex hull of the contact points must be contained in the true object. These both models are used for tactile object recognition: given a previously known object, the discrepancy between it and each of these two models generates a similarity index which allows to rank object hypotheses. For verifying the hypotheses, a very basic notion of uncertainty is used: it is aware of noise in that it introduces

thresholds when rendering hypotheses valid or not.

For their work on object exploration for acquiring grasp hypotheses Bierbaum and Rambow (2009) build a model of the object from tactile feedback. The contact points are used to build a polygon mesh, using the power crisp method of Amenta et al. (2001). It takes a set of points and constructs a watertight surface from it. The surface is the boundary between outer and inner spheres which are derived from the Delaunay triangulation and its dual Voronoi diagram of the points. This polygonal model can be used to extract shape features relevant for grasp hypotheses. One of the features Bierbaum and Rambow are interested in are parallel surfaces. For discovering parallelism, however, they found the measured end effector orientation as estimation for the surface normal to be more reliable than the estimates given from the faces of the mesh. Note that the power crisp algorithm does not use the normals to determine the orientation. Effectively, the power crisp representation is augmented with the more precise normal estimate.

4.1.4 Fusion of heterogeneous datasets

Object surface estimation is very similar to terrain modelling from sensor data. This is investigated by Vasudevan et al. (2011) in a complex multi-sensor, multi-modal setting. The representation of terrains as a graph of a two-dimensional function is very intuitive: the function value is the elevation at a given position on the plane. Such a function is what Vasudevan et al. learn to predict from sensor data: a Gaussian process is conditioned on the measured points. This is already a smooth function representation of a shape which comes nearer to what we use. However, elevation does not make a lot of sense for an object, unless we see it in polar coordinates and make a lot of assumptions about its convexity, origin of the coordinate system, etc.

4.1.5 Approximating implicit surfaces

Implicit surface representations can cure the deficiencies of elevation functions for our needs. Methods to acquire implicit surfaces from data include the Curless and Levoy's one (1996) for integration of multiple range images and Multi-level partition of unity implicits by Ohtake et al. (2005). The latter use an octree based adaptive subdivision of the space dependent on the level of detail. In each piece, they approximate a distance function accounting for the local shape type (e.g. edge or smooth). The partition of unity approach is then used to weight the contribution of each local approximations and blend them to a smooth global distance approximation. Its zero-level set approximates the surface. In their work Steinke et al. (2005) consider object reconstruction with implicit surfaces and morphing between implicit surfaces. They cast both as machine learning problems. For the former, which is relevant for us here, they employ a variation of support vector regression to learn the signed distance from the surface as a function in 3D. The surface is effectively the zero-level set of the learned function. The input data set is in form of surface points and normals but each point feeds the svm machinery with two observations: a positive one somewhat outside the surface in the direction of the normal and a negative one in the opposite direction. Such a 'pseudo' observation must be discarded if it turns out to be nearer to other observed points on the surface, in order to not overestimate the distance in bended regions. For the morphing, an svm learns a transformation from several given correspondences between key points, while transforming the whole volume, i.e. roughly preserving the distance to the surface.

Gaussian processes got introduced to implicit surface modelling by Williams and Fitzgibbon (2007). The setting is similar to the svm approach above, the definition of the function is inverted, though, and it is not interpreted as distance to the surface. They condition the gp estimate on positive observations inside the object, negative outside and zero-observations on the surface.

The authors develop thin plate covariance and emphasize that the prior it introduces allows to preserve desirable function structure in the absence of data, rather than restoring to zero mean like vanilla RBF covariance does.

In another approach to surface modelling Walder et al. (2006) minimize an objective function which is very similar to the SVM induced one. The surface points are, again, zero value observations, but the normals are directly taken into account rather than constructing interior and exterior pseudo observations. The authors use the same implicit surface definition as Steinke et al. (2005) – positive function values outside – and note that the derivative of the function on the surface is the normal to the surface. They augment the objective function with a term which penalises the deviation between observed and estimated gradient, which allows to directly include such observations. (Then, the authors take this occasion to generalise the representer theorem to linear operators: identity and derivatives in the case.) This is largely similar to how the GP machinery, which we employ here, deals with observed derivatives.

4.1.6 Visual representations

The above approaches directly model the surface of an object from 3D points. The nature of tactile and range sensors is that they probe points on the surface so that they deliver exactly the needed data. For visual sensors, however, the stream of images needs first to be processed to yield triangulated data points. Other representations do not focus on the shape. In Section 3.2 we mentioned the ones used by Calli et al. (2011) and by Saxena et al. (2008) for direct discovery of grasping points from vision and the one used by Kroemer et al. (2010) for actively learning a grasp value function. They both extract more abstract features from images and employ machine learning methods to discover combinations which are relevant to the task.

4.1.7 Summary

The state of the art in estimating comparable to our geometric object representations include polyhedron estimation from points and normals (Caselli et al., 1996) – estimating surfaces uncertain not due to sensor noise but due to insufficiently constraining input. Vasudevan et al. (2011) use elevation which is essentially 2.5D and barely suitable for manipulation, but employ very sophisticated machine learning methods for fusing sensor sources. The octomaps by Wurm et al. (2010) are probabilistically updated volumetric occupancy maps. Williams and Fitzgibbon (2007) introduce a probabilistic method to implicit surface estimation. Both Steinke et al. (2005) and Williams and Fitzgibbon (2007) use a inside-outside-virtual-observation hack to incorporate observed normals. On the contrary, Walder et al. (2006) have a principled way to learn from normals in their svm approach, but lack the probabilistic interpretation. All three Williams and Fitzgibbon (2007); Steinke et al. (2005); Wurm et al. (2010) do not interpret the resulting function as distance / potential or in a way directly usable for manipulation trajectories. In summary: neither is probabilistically interpretable *and* interpretable as potential/distance *and* dealing with a natural information like normals the right way – all these are required for a model to link sensor and motor systems.

Note that some of the shape representation approaches can benefit from normals attached to the surface points. Tactile sensing combined with the kinematic model can provide them. Structured light sensors and stereo vision, for instance, do not. However, a rough estimate of the normals can be obtained from the local topology of the neighbouring points and from the viewing direction.

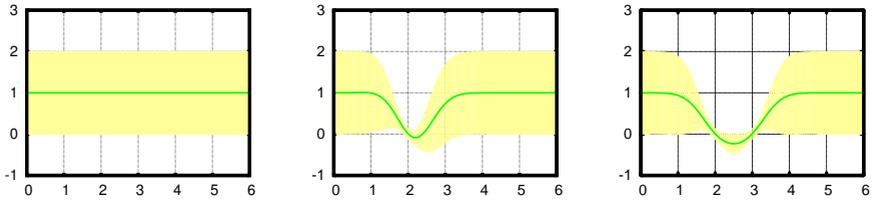


Figure 4.1: Learning an implicit surface representation for an object in $[2, 3]$. Initially $\mu(\mathbf{x}) = 1$. Belief after one observation of surface point and normal (left) and after two observations (right). Note that the variance of the posterior is a quantification of how precise and trustful the estimate is, given the present data and prior knowledge.

4.2 Shape estimation with GPISP

How can we condition a Gaussian process on sensor information such that the respective ISP represents an estimate of the object shape? Before seeing any evidence, the GP is initially biased to some positive value to express the lack of knowledge about objects in the space, e.g. $\mu(\mathbf{x}) = 1$. An observed surface point \mathbf{x}_s conditions the GP estimate to cross the zero plane: $f(\mathbf{x}_s) = 0$. In addition, the surface normal at the observation point can be incorporated in the GP as the gradient of the function, $\nabla f(\mathbf{x}) = \mathbf{n}_s$. Adding observations incrementally improves the GP estimate of the function, and its zero-level set resembles more closely the original object surface. Consider a minimalist 1D example for illustration: an 1D object is a segment and its surface consists of the two points which delimit the segment. This is plotted on Figure 4.1. The GP prior has bias $\mu = 1$, and two observations at $x = 2$ and $x = 3$ condition the GP posterior. First we observe $f(2) = 0$. We must ‘look’ (light sensor) or ‘approach’ (tactile) from left in order to make this observation, i.e. the surface inner side is on the right. From this we can conclude that the gradient points to the left, i.e. $\nabla f(x) = -1$. Now the belief describes a segment in approximately $[2, 2.3]$. Consistently, it shows a lot of confidence at the observed point but not much at the other zero-point. This comes about as result of extrapolation due

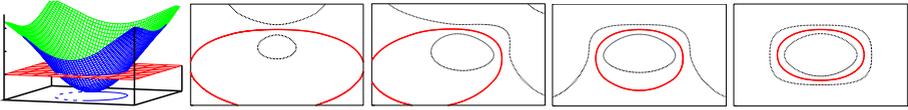


Figure 4.2: Learning an implicit surface representation for an object in the plane. The estimated function is a surface as in the first plot. Next plots show the projection on the plane. The thick red line is where the mean crosses zero, thin lines are the zero-levels of the ± 1 standard deviation surfaces.

to the prior imposing particular degree of smoothness. Then we observe the other side of the object, $f(3) = 0$. The corresponding ‘surface normal’ here is in the opposite direction, $\nabla f(x) = 1$. The estimate is now complete.

Analogously, Figure 4.2 shows the evolution of the belief in a 2D example. The true object is a disc, i.e. its surface is a circle. The graph of the estimated function is a funnel which crosses the zero plane. The projection of this contour in the plane is the estimate of the object’s surface. This is what the four plots depict. In the previous figure, the shaded area denotes the confidence belt of ± 1 standard deviation. Here the thin lines are the zero-levels of the same upper and lower confidence borders. A sensor observes the north, west, east and south point on the surface of the object, improving the estimate with decreasing uncertainty. Note that, unlike the previous example, there are infinitely many points on the surface so that the estimate could always improve by getting new point. However, by choosing proper prior, we would often not need many observation since near points would be estimated well from available evidence.

In the following we stress again what information do the observations carry.

4.3 Data (points and normals)

Evidence comes to a GP as function values and derivative observations. In the context of implicit surfaces these are surface points and surface normals.

The representation is a function which goes through zero at the object surface. This needs to be ensured when feeding the data in the GP. Section 4.1 showed a way to estimate by giving off-surface points. Technically, it is required in that case in order to teach the SVM what is inside and outside. With GP, though, one can condition on the derivatives of the function, i.e. ascertain the gradient – it carries the surface orientation. Furthermore, from the viewpoint of creating sensor-near interactive representation, one can argue that the sensor data come into being exactly on the surface. This is obvious for tactile feedback and laser scans. But stereo vision and structured light sensors as well read features and patterns from the surface. Same applies to infrared cameras, which just operate on different spectrum of electromagnetic waves. Other measurements, like the magnitude of a field or temperature, do not directly give clue on the shape of an object and are therefore not relevant to the shape estimation. If additional prior information is used to infer the surface position from these readings, again the surface gives rise to the data. It is not surprising that the input for a GPISP are points from the surface of an object and the normals to the surface at that points.

4.3.1 Surface point observation

In the simplest case the sensors return coordinates of points which are believed to be on the object surface (e.g. laser points, stereo triangulated points or contact points). In GPISP terms this data implies that the value of the potential at these points is zero. Therefore, every point in the data set implies a value-zero-observation. The positively biased GP mean is pulled down by the observed zeros and seeks to bring the evidence in agreement with the priors. The hypothesis that agree best with these constraints becomes the posterior mean.

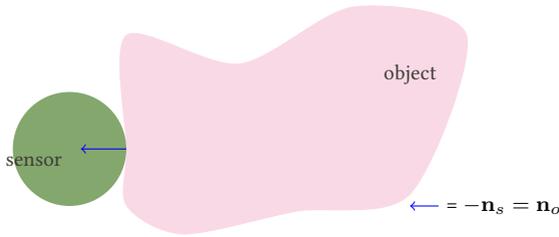


Figure 4.3: Example of a tactile sensor touching an object. Relationship between the normals of the tangent planes of the object sensor at the contact point. $-\mathbf{n}_s(\mathbf{x}) = \mathbf{n}_o(\mathbf{x})$ (depicted as \leftarrow)

4.3.2 Precision

Searching for the best hypothesis, the degrees of freedom (loosely spoken) to which the GP may violate priors and observed constraints is given by the priors' variance and the variance of the sensor noise distribution. In other words, the latter variance is the inverse precision of the sensor. It is part of the data point, each observation carries the precision, the confidence of recording it and thus instructs the GP how much to trust it when adjusting.

4.3.3 Surface normal observation

On registered contact, tactile sensors do not only give information on the location of a 3D surface point but also on the normal vector $\mathbf{n}_o(\mathbf{x})$ of the local surface tangent, as illustrated in Figure 4.3

Other sensors, e.g. laser, stereo camera, structured light, optical flow, can deliver normals, too. Knowing the position of the sensor one can derive information about the normal (with more uncertainty, though) from the direction of the visual contact or the laser ray. If multiple neighbouring points are available, a better guess for the normal can be inferred as an approximation from the point's neighbours: the sum of the normals to the adjacent faces.

By definition of implicit surface, the normal vector of the tangent plane needs to be equal to the gradient direction of the implicit surface potential, i.e.

$\nabla f(\mathbf{x}) = \mathbf{n}_o(\mathbf{x})$. The gradient is the vector of partial derivatives, thus, each vector component is a separate derivative observation, carrying its precision.

4.4 Fusion of multi-modal data

Every piece of information coming from the robot's sensors can contribute to improve an object estimate. This implies that different modalities need to be fused for the benefit of accuracy.

The ability to fuse different modalities amounts to provide a procedure for each sensor to incorporate its readings into a representation. The input for this procedure may be the raw sensor feedback or may demand some pre-processing. When we argue that GPISP is capable of multi-modal data, we emphasize two points: (i) the preprocessing is reasonably simple and (ii) we combine the information in principled way.

4.4.1 Preprocessing

Speaking in system design terms, the interface of our method are surface points and normals. Per sensor type, we need to show the way in which its readings are transformed into the suitable points and normals.

Point clouds are very common in recent days; this is facilitated by the development and wide spread of the Point Cloud library (PCL) (Rusu and Cousins, 2011). In a point cloud, the points are explicitly available by design, but not necessarily the surface normals. They can be inferred, however, by averaging normals and/or knowing the sensor position at the time of delivering a point. Thus, any sensor yielding a point cloud is easy to interface with GPISP.

In the case of tactile feedback, we use the awareness of the robot about its own body: forward kinematics determines upon detected contact the position of the texel that fires. The normal to the body surface at the contact point is an estimate of the normal of the object surface. More elaborated interpretation

of the pressure distribution on the tactile sensor can shed even more light on the surface locally.

4.4.2 Combining sources

Conditioning a GP on an observation means to provide a value (of the function or its derivative) and variance. The latter stands for the confidence of the provided information. This variance can differ for every observation. For convenience, we use in our experiments single variance for all value observations and single variance for all derivative observation. This makes sense in the context of a single sensor. But the ability to have heterogeneous variance on observation basis is one of the strengths of the GP which makes it so appealing as approximation tool in our representation. When the observations come from different sources, the accuracy of both value and normal will be different. The natural solution is to have a pair of variances – for value and derivative – attached to each source.

In a more involved scenario, the precision of a sensors may vary in different states – this would imply non-stationary variance models. A straightforward example is a camera which encounters different light conditions in different parts of the environment.

The precision of a normal may even depend on the direction. Then with appropriately chosen frame of reference, the single partial derivatives – the components of the normal vector – can be observed with different variance.

4.5 Performance

The performance of GPISP is largely determined by standard considerations about GP performance. For a given set of observations the GP delivers posterior predictive distribution. Typically, the mean of the distribution will be used in

subsequent applications.

$$\bar{f}_*(\mathbf{x}^*) = \mathbf{k}^{*T} (K + \sigma_n^2 \mathbf{I})^{-1} \mathbf{y} \quad (4.1)$$

The Gram matrix augmented with the observation noise on the diagonal $\mathbf{G} = K + \sigma_n^2 \mathbf{I}$ is constant for a fixed set of observations. Given this, prediction of a point from the mean involves multiplication of a vector of covariances (\mathbf{k}_*), vector of observations (\mathbf{y}) and the inverted Gram matrix (s. Formula 4.1). The vector \mathbf{k}^* contains the covariances of the query point with the observations. The covariance computes an inner product (in the input space) and some arithmetic operations, i.e. can be considered $O(1)$ w.r.t. number of observations n , making the construction of \mathbf{k}^* linear. The rest of the line does not depend on the query point, so it can be computed once for a set of observations, yielding a vector. Having done this, the multiplication complexity of $O(n)$ dominates the computation.

For fixed observations, the computation of $\mathbf{G}^{-1} \mathbf{y}$ takes time of $O(n^3)$ for the inversion and $O(n^2)$ for the multiplication. (To be more precise, inversion can be done in less than $O(n^3)$ but the constants start to play big role.) Note that these are sequential complexities and considerable speedup can be achieved on parallel architectures, e.g. using GPUs.

The importance of the inversion starts to matter when observations come incrementally. Then with each new observation the Gramian needs to be inverted again. To mitigate this one can use, e.g. an incremental algorithm which builds an inverse matrix from the available inverse of the submatrix on a lower cost (using Woodbury and Sherman-Morrison identities (Hager, 1989)).

Another technique which can be employed to cure performance deficit due to growing amount of observations is lazy evaluation. Above we followed the strategy to pre-compute the query point independent product $\mathbf{G}^{-1} \mathbf{y}$ once the whole data is available. For large n , especially for dense observations and comparatively small covariance width, the influence of far (as determined by

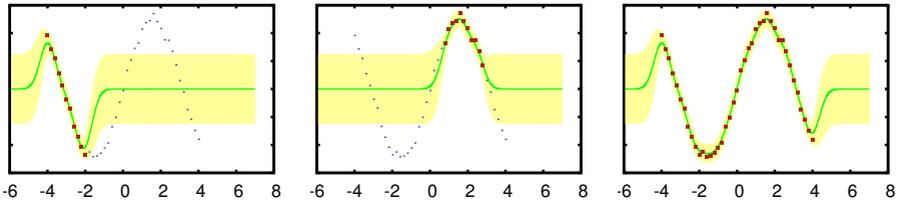


Figure 4.4: GP plot to show lazy learning, on-demand evaluation using local neighbourhood of the query point. The available data is shown as blue dots on all plots. Most left and middle plot use the highlighted data only: in $[-4, -2]$ and in $[1, 3]$ respectively. Most right GP uses all data.

covariance width) observations on the prediction is negligible. It will likely turn out that doing the whole procedure on the fly for reduced set of near observation is more worth than pre-computing fixed results for all data. Lazy evaluation corresponds to a GP which pretends to know only a subset of relevant (near) observation. Figure 4.4 illustrates the principle with a toy GP. The resulting estimate is accurate around the used data, but degrades away from it. (The GP mean in absence of data is zero, and the estimate degrades to it.) The new complexity calculation needs to include (i) finding the nearest m observations (or all neighbours in a fixed neighbourhood $\|\mathbf{x}^* - \mathbf{x}\| < \epsilon$) of the query point and (ii) doing the inversion and multiplication for small set of observations. Finding nearest points in big datasets can be expensive too. It depends mainly on the way the data is organised. An efficient data structure for spatially interpretable data is the octree, which partitions recursively the space resulting in near points laying on near leaves, thus easily addressable together.

The time for querying several points in an online application is negligible. This is the case for our grasping controller, where we only need a few motion features derived from the GPISF, including derivatives of the mean and variance. However, if large portions of the mean need to be queried, complexity becomes apparent (e.g. visualisation); even more if this is coupled with high

rate of arriving observations.

4.6 Visualisation

Visualisation of the model is already a demanding application which requires to estimate unseen regions, often at a fine resolution, i.e. lots of estimates need to be computed. The primary design goal for GPISP is not visualisation. Unlike the interface for sensory data, the relevant information for visualisation needs to be extracted non-trivially. In the first place, the *implicit* surface is defined by a function which is defined on the whole space without *explicit* pointer to the location of the object. The only thing one can do is to probe the function at many places: where the function is negative, the space is occupied.

Visualisation often amounts to polygonisation of the surface because polygons are well suited for being drawn and rendered in 3D graphics. Probably the most prominent technique to polygonise an implicit surface is the Marching cubes algorithm (MC) (Lorensen and Cline, 1987; Chernyaev, 1995; Lewiner et al., 2003). For each voxel in a discretised volume, it probes the function value at its vertices and if they include both positive and negative values, the surface crosses this voxel. Now depending on the configuration of the positive and negative vertices the algorithm decides how exactly the part of the surface goes. This can be made arbitrarily precise – the smaller the voxels, the finer the resolution. Marching cubes works well for arbitrary surfaces. However, this comes at high computational cost, which we can reduce by introducing some assumptions rooted in our way of growing surfaces. Although the voxels relevant for the surface are the ones laying on it, MC probes the whole volume containing the surface. This is the only way to make sure that it does not overlook a small piece of surface. Instead, if one assumes that the function is smooth and all pieces are connected, starting at a surface voxel, only a belt of voxels around the true surface must be visited. For finding the starting voxel one either needs an oracle to show a point on the surface, or systematically

probe voxels until one hits the surface (like normal Marching cubes would do anyway), and then continue along the surface only. Properties of the function can be exploited, e.g. in our case the monotonicity, so that starting from a random point, a walk along the gradient will hit the surface. The smoothness assumption is quite natural, but what if one needs to visualise intermediate steps of an estimation procedure, where a few blobs are still isolated (Figure 4.6 shows an example in the next section)? Then, the connectedness assumption does not hold, and a complete model can only be produced if it is guaranteed that the procedure starts at least once for each separate surface. For GPISP, there are very good hints for starting points: the observations. While there is no guarantee, typically the priors and noise distributions are chosen so that the observation points are very near to the surface. At most a few steps along the negative gradient (for positive f ; along the gradient for negative f) yield a starting point for polygonisation.

Polygonisation builds a complete model and even if visiting only surface nodes this can cost a lot. Then one can reduce the resolution, but this leads to sharp edges and appearance suffers. Scales are an alternative to polygonisation for visualisation. A scale is a small flat surface centered at a point on the object surface, its normal aligned with the object surface normal. Aside from the question how to find the points on the object surface, individual scales are more independent (not necessarily seen as being neighbour of other scale) and have the property that locally they remain true to the original contours. This is handy if one wants to bound the time needed for visualisation by fixing, say, 500 scales to be shown. Then one can sample rays in the 3D space and via the field find points on the surface to place scales at. Figure 4.5 shows a low resolution crude visualisation of sphere with polygon mesh and a corresponding scales sphere. To each vertex corresponds a scale at the same position on the sphere – both are not optimally distributed on the surface. At small resolutions, scales can be more appealing for visualisation of smooth objects, since they allow the human perception to complete the missing transition between

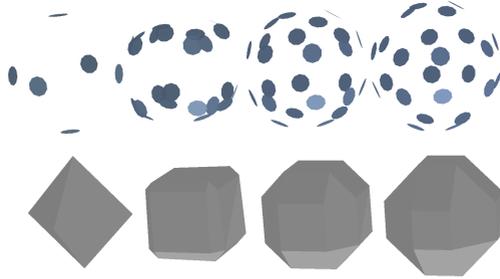


Figure 4.5: A sphere visualised with polygons and with scales at small resolutions. For fair comparison the scales coincide with the mesh vertices.

scales. They do not force the perception to assume false contours. This advantage gets even more apparent with view point changes, and the relative positions of the scales is better perceived.

4.7 Noise distributions

There is a point in which we were not very accurate in the interpretation. We state that we translate sensor precisions to variance of the noise distribution of a value observation. This is not strictly correct. Actually, the sensor registers surface but is unsure about the location, i.e. the noise is around \mathbf{x} , not around $f(\mathbf{x}) = 0$. Furthermore, depending on the choice of coordinate frame, it might be non isotropic. Similar arguments hold for the observed normals. The one source of uncertainty is again the position, the other is the process of estimation of the normal direction itself. The true noise distributions might be much more complex – asymmetric, with fat tails, anisotropic. Here we approximate them with a Gaussian for the sake of simplicity.

For the evaluation of the estimation procedure, we leave out sensor fusion problematics and focus on showing that the representation performs reasonably inside its domain. This is the topic of the experiment described in the following section.



Figure 4.6: Random object (top left); belief improvement with new observations (black dots).

4.8 Experiment: estimation of synthetic surfaces

We carry out following experiment as basic validation of the estimation scheme. It can be thought of as using synthetic tactile data for a randomly generated object. First we create a GP with certain prior distribution and covariance. From this initial distribution we sample ‘observations’ which we condition the GP on. Note that these are not observations in terms of GPISP, i.e. not zero-value and normals, just arbitrary function observations. We obtain an implicit surface by sampling a function from the GP posterior. This is all about producing a synthetic object which serves as source of observations and is to be estimated in the following. Now we randomly select points on the surface of that object and use them as input to a GPISP – simulated observations of surface points and normals. The mean of the latter GP’s posterior constitutes the object estimate. Refer to Figure 4.6 for snapshots from this procedure. The first image shows the random object which is to be estimated. Starting with one observation, second image, the belief about the object improves progressively with new evidence. Black dots mark observed points on the surface.

We repeat the experiment with different prior variances, covariance widths and sensor noise levels. Note that these parameters are the same for the two GPs in a single experiment. In other words, for estimation we employ a second GP which is armed with very good priors for interpreting the evidence. This makes its task easier than what it is exposed to in the wild reality. Aware of this fact, we interpret the results of the experiment merely as a validity check for the conjecture that the strength of GPISP is the easy and principled way of incorporating priors resulting from the embodiment, which enables effective

estimation from few observations only.

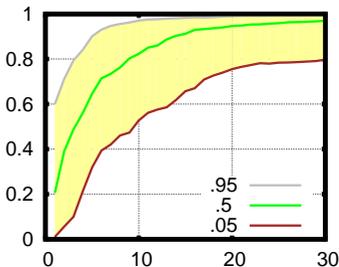


Figure 4.7: The median and upper and lower 5% quantile for the precision of the shape estimation as function of the number of (random) observations provided.

Figure 4.7 shows a plot of the accuracy of the estimation when using different number of observations. The quantiles for particular observation count (on the x -axis) are computed from 300 randomly generated objects of different sizes. The value on the y axis is the similarity of true and estimated object:

$$\frac{v_c - v_o}{v_t}. \tag{4.2}$$

This is obtained by measuring the common volume of estimated and true object, subtracting the falsely overestimated volume (inside the estimated, but not in the true). This absolute value is normalized by the volume of the true object to obtain comparable values independent of the scale. It is impressive that with as few as 10 observations more than half of the trials score above 80%. Furthermore, with 30 observation the estimate is practically identical to the original object. Translating this to tactile experience if we assume a human hand delivers five observations when sensing for an object, 10 observations mean only two sense attempts. The proper priors seem to enable GPISF to estimate effectively even with randomly collected observations.

4.9 Summary

How sensor information goes into an object model is determined by the chosen representation. Since GPISP represents shape, it is fed by shape data – *points* from the surface and *normals* to the surface at these points (where the normals are vectors of partial derivatives). In addition, *precision* is associated to each such observation to reflect the confidence of the measurement. In these terms, fusing *multi-modal data* amounts to obtaining the points and normals and supplying the proper uncertainty measure. Lazy evaluation of the model can be used to alleviate the computational complexity due to large data sets – this is suitable for the computation of arbitrary features from the GPISP estimate. In a special application – visualisation –, GPISP specific shortcuts can be made to ease the sampling of surface points compared to standard discretisation algorithms. An experiment in simulation proves the sanity of the estimation method illustrating the combination of correct prior and few evidence from the shape.

Surface data is the focus of GPISP. This can bring about disadvantages. Unlike volumetric representations, e.g. voxel trees, it cannot naturally accommodate some implicit data contained in light reflection based sensors: GPISP is not efficient at learning from the way of the beam without additional hacks. This is related also to the lost information from tactile sensing: the volume of the rest of the kinematic structure carrying the tactile sensor could very doubtlessly witness free space, however this is not trivial to integrate.

Chapter 5

Grasp synthesis

The development of a grasp controller operating on a GPISP model is the topic of this chapter. An object representation for grasping needs to facilitate generation of grasp motion. Here we illustrate GPISP's suitability for motion generation for manipulation. The controller is formulated in terms of the potential field of GPISP. First we comment existing approaches to grasping and to collision avoidance, then we formulate multiple tasks which, combined and weighted properly into a single objective function, lead to a feasible grasp. We show two approaches – a reactive one and one optimising a whole trajectory prior to execution – which operate on the same, for the most part, objective functions.

5.1 Related work: Approaches to grasping

Beyond shape, what constitutes a good grasp or successful manipulation? A geometric approach focuses on interaction with rigid objects or rigid parts and thus alone cannot master all situations. Some problems require higher reasoning. For instance, think of grasping a piece of flaky cake: its inner forces are smaller than needed to resist gravity. The way to lift it would be from

below, lifting its supporting plate or using spatula. This skill involves planning or retrieving a sequence of primitives, among others, rigid manipulation ones for intermediate steps.

Some non-shape properties can be dealt with by the parameters of a grasp primitive: stop conditions, initialisation of the controller, etc. For instance, if an object offers multiple grasp affordances, and particular one is more preferable in particular situation, a higher level controller needs to decide and probably restrict the primitive to certain approach direction. Fragile objects, for another example, need to be grasped delicately, which can be achieved by lower thresholds on tactile sensors as condition to stop closing the fingers.

Abstracting from primitives, taking them for granted and creating complex behaviours has traditionally been research question of classical AI. There are powerful logical representations which AI systems use to represent the world with facts and infer a sequence of actions.

The above reasonings are out of our scope. This chapter aims to provide low level manipulation primitives. It follows a geometric approach to grasping, i.e. in the scope of the controller are situations in which merely the shape of the object has influence on the grasp. Other features of the object, like mass inhomogeneity, deformability, variable surface friction, etc. are assumed unimportant or dealt with on another system level.

Traditionally, for the sake of simplicity, the generation of grasp motion has been viewed as two problems: finding (near-)final *hand posture* (to close the fingers from), and *reaching*, i.e. moving the hand to that posture.

5.1.1 Posture

There exist a plenty of approaches to find a grasp posture. In the following we mostly revisit the related work discussed in previous chapters to review the grasp techniques suggested by their particular representations. The representations inevitably dictate the organisation of the subsections: polyhed-

ral approaches test stability measures, visual approaches are more heuristic. Compliant and smart alternative hardware complete the review.

5.1.1.1 Grasping with stability measures

Common pattern is to first find the contact points on the surface and then find a posture which can grasp at that points. There are sound measures which can assess the stability of a hand-object-system, e.g. form-closure and force-closure (Bicchi, 1995). (For more examples, Suárez et al. (2006) review various quality measures for grasping.) Roughly, a grasp is form-closure if the frictionless contacts do not allow the object to move in any direction; and is force-closure if any external wrench can be compensated for by the grasp forces (here friction is taken into account). In addition to these binary criteria, partial force-closure can be defined as resistance to a *set of wrenches* – which captures the particularities of a manipulation task. (By analogy there is partial form-closure.) On the one hand, these notions and their partial variants can be used as performance measure in algorithm comparison. On the other hand, they can be used to rank grasps according to the amount of disturbance they allow. Thus, closure arguments can turn the search for suitable contact points into an optimisation problem: evaluating the measures on different configurations one can find the most suitable one. This can be used with rigid objects for which the system has a precise object model before the grasp. The results are as accurate as the modeling of the contacts, the object and the actuators.

Another geometric argument goes under the name caging. Instead of fixing an object, it aims at restricting it within a volume (Rimon and Burdick, 1998; Diankov et al., 2008). While stable grasps allow to see the object as part of the kinematic chain, caging can be used as either pre-grasp from which to stably grasp (Rodriguez et al., 2012) or for pick and place tasks for which the precise pose of the object is not critical. (There are quite a lot everyday examples: this is often the case when picking goods, sorting fruits, etc.; and enough examples of fragile objects where it is not applicable to grab or put

things in arbitrary postures.)

Closure measures answer important questions, they are not the ultimate solution, though. One aspect is illustrated by the following quote from Bicchi (1995):

In many cases, even though form closure may not be verified, contact constraints partially restrain the motions of the object. Actually, these cases are most relevant to workpiece fixturing and dexterous manipulation.

This is, for example because a stable grasp needs to be reachable, too, and this leads the author to formalising the concepts of accessibility and detachability and partial closures mentioned above. Furthermore, specifying the right wrenches for a particular task might not be trivial. In addition, it is not viable to evaluate infinitely many combinations of points until a feasible grasp is found; moreover, the optimisation landscape is not necessarily benign and finding structure in it to guide the optimisation is not trivial. As intermediate step a heuristic search for good grasps can be employed to prune most of the search space and select only few candidates. Alternatively, modelling the task with wrenches and subsequent optimisation in that space can be avoided by employing other methods, search heuristics, features, and working on other representations.

A more recent work dealing with force-closure describes how to efficiently compute good grasps depending on how the objects shall be manipulated: Haschke et al. (2005) assume tasks specified in terms of wrench space vectors, cones and polytopes. These constraints are much more fine grain and thus more relevant to real scenarios than the binary force-closure. For that cases, they introduce task specific measures to assess the stability and applicability of a grasp/manipulation w.r.t. a task.

These grasp stability measures are concerned with the result of the whole process of grasping – the final posture – and often assume point contacts and clear object environment. However, other constraints may render many stable

grasps infeasible: for instance, kinematic constraints of the hand as opposed to idealised points, and obstacles around the object. An example for using such metric, and a step in compensating for additional constraints is done by Berenson et al. (2007) who store a set of precomputed force-closure grasps for each previously known object and evaluate their *score* according to robot kinematics and obstacles at the time a grasp needs to be executed. The stored grasps are produced by sampling a regular hypergrid in the parameter space of a grasp – which includes orientation, approach direction, hand configuration – thus maintaining control over the number of grasps to be evaluated. The score function sums the force-closure quality, a heuristic for kinematic feasibility, and the clutter along the approach direction. Eventually, the trajectory from the current to the final posture is planned by growing bidirectional RRTs. (We will discuss RRTs later in Section 5.1.2 when reviewing collision avoidance approaches.)

For obtaining grasp hypotheses Goldfeder et al. (2007) employ another representation: decomposition of an object into convex superellipsoids organised in a tree structure. (In a similar setting earlier Miller et al. (2003) simplify the grasp computation based on abstracting objects into collections of more basic shape primitives: cylinder, box, sphere, cone). They use an algorithm for splitting and fitting superquadrics to the raw surface point cloud, similar to the one by Zha et al. (1998) we mentioned earlier in Section 4.1. This procedure results in a decomposition tree on which leaves reside the final primitive superquadrics while the parent nodes hold the intermediate ones. The authors argue that using the inner nodes for the candidate grasp generation is beneficial because it forces the algorithm to also consider larger regions, rather than the final parts which may be too finely decomposed for the hand. Then for each node (leaf or inner), they sample points regularly on the corresponding superquadric surface to put the hand on. The hand then with opened fingers approaches the point along the surface normal and the fingers close upon contact. Effectively, this procedure is the heuristic used to reasonably limit the

number of grasps for which to check stability. As a next step, a grasp simulator (Miller and Allen, 2004) is used to rank the so constructed grasps in a physical simulation with the proper friction constants, kinematics, materials.

An alternative to invention is to let an expert, a human, demonstrate to a robot how to grasp. In the context of Programming by demonstration, Tegin et al. (2009) recognise demonstrated human grasps, map them to the robot kinematic model and build thus an experience database which can be searched for the proper grasp when the same object is encountered by the robot. They augment this general procedure by generating lots of grasps of the same type for each demonstrated one and labeling them according to their stability. The labels are result of offline simulation of the generated grasps on the object model. At runtime then, the best labeled one can be chosen which is reported to lead to robustness w.r.t. deviations in the recognised by the robot object pose.

5.1.1.2 Grasping with visual cues

In predicting suitable grasps from camera images, Saxena et al. (2008) speak of a grasping point as a region of the object (its 2D appearance on the image) which can be pinch-grasped. The extraction of basic visual features and subsequent classification of image patches eventually yields a grasping point which defines a corresponding hand posture. Then, by using a RRT a collision-free trajectory is planned to that posture. Here, the algorithm counts on the predictive power of the ML methods: neither physical simulation, nor grasp stability measures are used to check the feasibility of a grasp; and for performance assessment directly the execution on a real hardware is measured.

A vision based grasping strategy is employed also by Montesano and Lopes (2012) for actively learning the grasp value function over images. Since the emphasis is on the latter, the strategy is reasonably simple: a grasping point is chosen from the image, the hand moves to it and closes the fingers like by reflex. The learned mapping from image points to grasp success eventually

reflects the way the grasping is performed.

Employing together active vision and visual servoing Calli et al. (2011) grasp at points with maximal curvature using an eye-in-hand setting. They first obtain a contour model of the object, then find the concave parts with high curvature – the candidate grasp points – and finally evaluate them in terms of force-closure and how well they fit the kinematics of the robot: this yields the best grasp points. Now, a visual servoing controller is used to first find a view of the object which maximises the curvature at the selected grasp points and then drive the hand to the grasp.

5.1.1.3 Matching the shape

Strictly taken, the computation of final configuration yields usually a posture from which the fingers only start to close and get in contact with the surface. On this problem zoom Steffen et al. (2007) with their work on a policy which reacts to tactile events during grasping and selects from the experience database the most appropriate target posture according to the current configuration. In contrast to this, compliant hardware implicitly contributes such adaptive behaviour by the intelligence of the material or construction, e.g. Deimel and Brock (2013) which we mention later.

Some approaches to grasping question the hand architecture in the first place and arrive at alternative designs which provide very interesting insights about sometimes neglected aspects of the nature of grasping. For instance, the coffee gripper developed in Brown et al. (2010) is a flexible pouch full of coffee powder attached to a vacuum pump. In its normal state, the pouch is freely deformable and can wrap around any shape. Once the air is pumped out, the powder particles stick together, make the pouch stiff and it remains in the taken shape. If it is stiffened after it has wrapped around an object, it mostly can hold it stably, can deal with fragile objects and small objects.

Another example which retains the notion of fingers but argues against complex configuration spaces is Mason et al. (2011). Their hand has a flat

palm, three simple metal fingers and a single controllable degree of freedom. When the fingers close, they force one or multiple objects which are in reach into a stable position. The addressed problem is bin picking and dealing with multiple grasped items, as well as correcting the orientation of a grasped item. The approach leads to non-obvious insights, e.g. that a low friction hand leads to smaller number of stable end poses of the grasped object.

In a related way Eppner et al. (2012) argue by a thought experiment that a human deprived from most dexterity and feedback of his hand by wearing a mitten still can successfully grasp most common objects. The authors examine the importance of compliance of object and hand. They use eye-in-hand system which moves at constant distance to the target object and monitors its contour. The way the contour changes with viewing angle and its eccentricity reveal one of three basic forms: spherical, cylindrical and box. These forms can be seen as compliance modes which teach the hand the appropriate pre-shape as to maximise the compliance with the object. One could interpret this approach as working on object represented as a single primitive but motivated from different viewing angle: a primitive is not used to approximate the shape, it is the mere fact that the object is classified as this type of primitive that matters. Continuing the mitten argument, the soft hand developed in Deimel and Brock (2013) has silicon rubber fingers which are thus passively highly compliant. The material encloses an air camera inside each finger so that it can be actuated pneumatically in a simple way. As a result, the hand can match various shapes and successfully grasps objects placed reasonably in its range.

In the current subsection we discussed the search and assessment of a final grasp posture which some authors have alternative names for, depending on their research motivation: grasp hypothesis, affordance, etc. In the following Section 5.1.2 we review traditional *planning* approaches under the topic of collision-free navigation to the selected posture and allude to alternative formulation using probabilistic inference.

5.1.2 Collision avoidance, path finding and planning

In Section 2.1.3, we already explained why collisions need to be avoided when moving a robot. One can argue that moving collision-free from current to goal position is the very substance of motion generation – for a mobile robot on the ground and for a kinematic structure likewise. For the latter case one can formulate the problem in the configuration space such that obstacles need to be translated to usually higher-dimensional space of joint angles and appear as solid areas there. Then the search for articulated collision free movement amounts to finding path for a point. However, this does not scale well with the complexity of the world and the system. Thus, often it is preferable to plan in an easier, abstract domain and translate back through the known kinematics.

Every realistic problem contains obstacles. Following a straight line from start to goal, as simple PID control on the position deviation does, likely fails. The information about obstacles needs to be considered in the control. The options are either to augment the feedback control with a method to walk around obstacles, if encountered; or to simulate moves in advance, before moving, as to find a valid way to the goal.

5.1.2.1 Obstacle avoidance on demand

In a world which is partitioned in ‘free’ and ‘occupied’ areas, an example for dealing with obstacles is the BUG algorithm (Lumelsky and Stepanov, 1986). It tries to go a straight line to the goal and when it encounters an object, follows the contour until it again reaches the straight line at a point nearer to the goal (s. Figure 5.1 for illustration). Although seemingly very simple, it is notable for the guarantee to successfully avoid obstacles in any configuration. It is applicable even in a feedback setting, i.e. the obstacles need not to be known prior to starting. The resulting trajectories look naïve, though, because the BUG algorithm is not concerned with qualities other than a valid path.

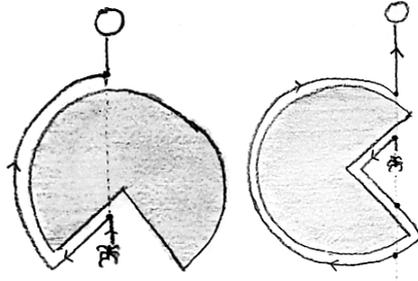


Figure 5.1: The bug algorithm

5.1.2.2 Obstacle avoidance by sampling and planning

Other approaches require to know the obstacles beforehand. They employ sampling techniques and trade some guarantees off for more explorative behaviour. Probabilistic road maps (PRM) sample points from the free space and use a local planner to connect neighbouring ones (Kavraki et al., 1996). With an adequate sampling technique, this produces a graph which covers the relevant free space and any graph search algorithm can be used to find a path between nodes. On the downside, this approach needs time to build the graph and needs to know the obstacles prior to starting, but the roadmaps can be reused as long as the environment does not change. They explore the free space better and the graphs allow to not only find a valid path, but e.g. the shortest one, too. Theoretical results (Svestka and Overmars, 1995) give probabilistic guarantees: arbitrarily high probability to find a path for given endpoints and sampling parameters can be achieved by sampling long enough (exponential in space dimensionality). An expensive procedure is the local planner which ensures that there is a collision-free path between graph nodes so that they can be connected. In the simplest case, it is a higher resolution collision check on the straight line. But for more complex systems there is no guarantee that the one node is reachable from the other with the available degrees of freedom (e.g. holonomic systems).

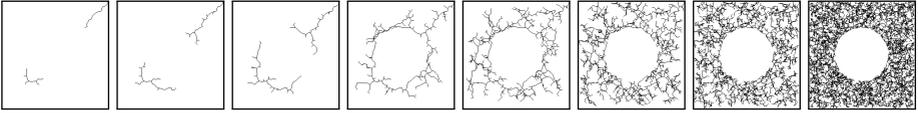


Figure 5.2: Two RRTs growing from start and goal and exploring a planar world with a disc obstacle.

Rapidly exploring random trees (RRT) introduced by LaValle (1998) address the latter problem. Rooted at the start position they, again, sample in the free space but the sample is taken merely as a direction to grow to. From the nearest node in the graph, a local planner goes few steps in this direction and the reached point is added to the tree. The local planner can be just a check for collisions, but typically would be aware of the capabilities of the system. This ensures that the edges in RRT represent feasible paths. Like PRM, RRT cover the free space well, probabilistically guarantee convergence and are easy to implement. Road maps and RRT, though, are agnostic to which nodes are the particular start and end points – they fill the space in all directions equally. This additional information can be used to accelerate the search as an extension of RRT does. With certain probability, as reference direction is taken the goal point. Another technique to make path search more goal-oriented is to concurrently grow two trees from the start and goal point until they connect (Kuffner Jr and LaValle, 2000). Figure 5.2 shows an example of two trees growing to each other and exploring a 2D work space with simple circular obstacle in the middle.

5.1.2.3 Grid search and optimality

Other notable algorithms for finding shortest paths are the derivatives of Dijkstra’s graph search, like A^* (Hart et al., 1968), widely used in 2D discrete domains where each cell is either free or occupied, e.g. mobile robot navigation. In the simplest case it would effectively breadth-first-search, labeling neighbouring cells of the goal to have a distance of 1 to it. Their neighbouring cells

in turn are labeled with a distance of 2 to the goal, their neighbours with 3, etc. Like a wavefront, the distance labels propagate until they reach the start cell. The found path follows in each step the least label of all neighbours. These algorithms guarantee to find a path if one exists. A^* employs a heuristic distance function to speed up search by early pruning unpromising paths (Dechter and Pearl, 1985) and thus avoid unnecessary exploration of graph branches. Depending on the heuristic, it may miss the shortest path, but one can derive guarantees on how much longer it is (Likhachev et al., 2003). The discretisation such planners require is too expensive in higher dimensions which makes them unpopular for problems in configuration space. In this review, so far only this approach explicitly addresses optimality looking for the *shortest* path. This can be naturally extended to *cheapest* in terms of costs, other than distance, assigned to graph edges. The non-colliding is thus a hard constraint under which one seeks for a path with minimal cost. Still, this does not concern the optimality of the trajectory in terms of its intrinsic properties, e.g. smoothness. Such qualities, however, gain importance when it comes to execute the trajectory on real systems. To address them, one either needs to post-process the found path or go for an approach which can accommodate such constraints in the first place.

5.1.2.4 Potential fields for navigation

The binary partitioning in free and occupied space possibly oversimplifies the problem. This gets more apparent when dealing with increasingly dynamical agents and when one works with continuous trajectories. Then a continuous measure of how critical a state is with respect to collisions works better than a binary one. The simplest realisation of such relaxation of boundaries is the sum of the distances to all obstacles.

A prominent continuous collision avoidance approach represents the space as potential field which is high in the obstacles and decreases with the distance to them. Consider second potential field, minimal at the target and increas-

ing with the distance to it, e.g. funnel-shaped. The superposition of the fields would impact a particle to stay at safe distance to the obstacles while moving to the goal. In contrast to the path finding and planning algorithms mentioned, this one is greedy and does not guarantee reaching the goal, i.e. it can get trapped in local minima of the function if used in feedback manner. By using a more sophisticated optimisation technique, e.g. adding stochasticity, one can obtain probabilistic guarantees. Later on, we use the elegant metaphor of a particle under the influence of a potential field, reversing the object potentials for the objects we want to interact with. In more detail: artificial potential fields were introduced first for collision avoidance in planar navigation problems by Khatib (1986). He assigns to each obstacle a repulsive potential field and an attractive one to the goal configurations. Figure 5.3 illustrates a simple planar workspace model. The superposition of repulsive and attractive potentials is not immune to local minima. This can be seen in simple situations; it is sometimes referred to as the banana problem. There are various ways to specify the potential and how large it gets near the object, how fast it decays with increasing distance, what shape the iso-potential lines take around the object. Some of the approaches address the local minima, e.g. Khosla and Volpe (1988) state that with properly chosen curvatures of the repulsive and attractive potentials it is possible to compensate for local minima of arbitrary scale. As potentials the authors propose to use functions which implicitly describe superquadrics: then, the power of superquadrics can be used to approximate large variety of shapes.

Artificial potentials are used in the grasping part of their work on tactile exploration for acquiring grasps hypotheses by Bierbaum et al. (2008). For controlling the hand they employ the same idea of particles attached to the fingers. First they sense for the object by including and deleting potential sources on the surface and thus obtain a point cloud. To come up with a grasp affordance, they extract features from the point cloud, in particular they look for pairs of large parallel faces which positions fit the hand's capabilities. (Par-

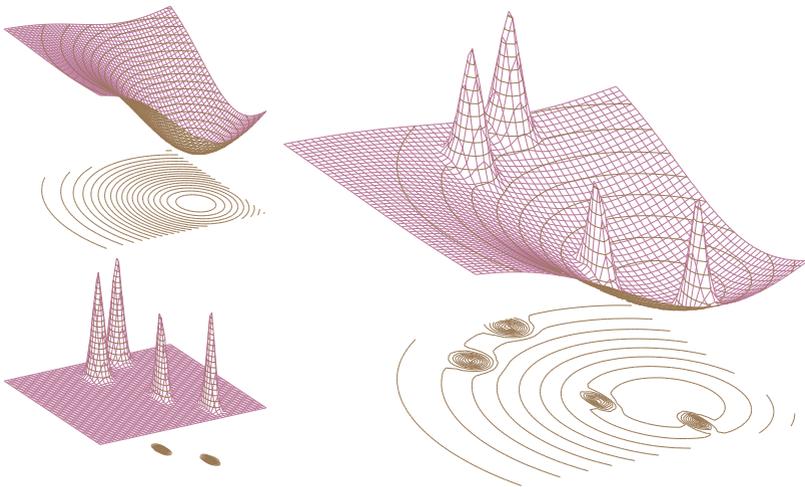


Figure 5.3: Superposition of repulsive and attractive potentials in a planar workspace.

allel faces are commonly employed feature since (i) objects often have box-like parts, (ii) simple jaw grippers are quite common and (iii) most other hardware can go into a configuration in which two fingers oppose.) From that affordance the grasp points are obtained to which the fingers navigate – again, using a potential field generated by adding and removing attractive and repulsive potentials on the surface. The resulting representation is a mixture of a point cloud – for extracting the grasp features, and interactively obtained dynamic potential field – for hand control. This approach comes closest to our way of navigating a hand for grasping and exploration. Our heuristics for feasible grasps or query points are different, however, because they operate on different representation: we use for grasping the very same representation we use for sensing. Furthermore, the potential field is different: ours appears as generated by the whole surface, rather than from point sources. Later in this chapter we explain our heuristics and navigation in detail.

5.1.2.5 Mixing trajectory objectives

In a more general view, avoiding collision is yet another task to be accounted for when acting according to some objectives. This is the way we go for the synthesis of grasp motion. We formulate, in addition to the functional tasks regarding grasping, additional objectives which ensure the body remains at safe distance to the obstacles. These objectives can equally well be used in trajectory planning as well as in closed-loop motion.

The combination of different types of tasks is one of the topics discussed by Zarubin et al. (2012). They are motivated by the need to plan a trajectory which accounts for tasks formulated in different spaces. The setting is optimisation of complex reach trajectories by using inference: Approximate inference control (AICO) introduced by Toussaint (2009), which we also employ later on. The discrete steps of a trajectory are considered as a graphical model where the goals of the different tasks are assumed observed (there might be goals for the last but also for intermediate steps). This effectively conditions the graphical model on the goals, the inference propagates them consistently and computes a locally optimal trajectory. While the penalisation of large movements and joint limits are intrinsically formulated in configuration space, Zarubin et al. (2012) introduce two other topological spaces to robot control: the writhe matrix and interaction mesh. For two kinematic chains, the numbers in their corresponding writhe matrix expresses how much a segment of the one chain is wrapped around a segment of the other. Thus, a particular matrix can be used to describe the relative pose of the two chains – wrapping around an object or winding two strains – and is invariant to global orientation, to configuration space changes, etc. The other invariant, interaction mesh, specifies in a graph the relations between relevant points of two objects and can be used as alternative to collision avoidance in euclidean space. In particular, it is useful with moving objects and targets to express a target situation independent of the particular euclidean positions of the bodies. For each task, by

defining an differentiable map from configuration space to the task's space, essentially one augments the graphical model by a dependent variable in each step, corresponding to that task. This allows to plan in the more convenient auxiliary spaces and the authors demonstrate this on a complex task for which the standard sampling algorithms fail: unwinding of a winded around a pile chain and reaching subsequently for a target. In the other demonstration, a robot hand reaches through a moving hole – where the in-hole configuration is described in terms of interaction mesh and writhe.

The notion of winding numbers has been used also by Pokorny et al. (2013). In their approach, starting from a point cloud, they triangulate it and use the resulting mesh to extract topological information, in particular about holes in the object. The robot hand's detailed structure is abstracted, instead it is modeled as strings which span the opposing fingers. The winding numbers and Gauss linking of the hand strings w.r.t. the closed curve of the object hole are used to find hand postures which wrap around the hole. The search for such postures is done by defining an objective in terms of winding numbers which corresponds to the situation that the fingers wind around the hole contour. For this objective the Jacobian w.r.t. the configuration of the hand can be computed. Then starting from multiple sampled positions around the object, using inverse kinematic control in the topological space in small steps, the posture of the hand converges to a local minimum of the objective. This yields grasp hypotheses for objects with holes.

5.1.2.6 Imitation of trajectories

Imitation learning – letting an expert demonstrate how to do things and trying to learn skills – is another approach applied successfully in manipulation tasks. Addressing the grasping and reaching as one skill, Jetchev and Toussaint (2011) develop a method to retrieve the relevant low dimensional manifold out of a large feature space. They represent a demonstrated trajectory by, for each step, a vector containing large number of possibly relevant fea-

tures. Instead of directly learning a policy, they estimate a cost landscape which agrees with the demonstrated trajectories in the sense that they result in minimal costs. To facilitate this, the observed demonstrations as well as fictive suboptimal ones are included in a training set. An elaborated loss function ensures the consistency with the demonstrated trajectories and their gradients and additionally enforces sparsity. The optimisation in a neural network model setting yields a compact sub-representation: the retrieved task space. The cost is a function of these features only and discards the irrelevant ones. Using the so learned cost landscape, a planner can generalise across various unseen situations and its performance degrades more gracefully than other imitation learning methods which do not take the additional step of cost estimation.

5.1.2.7 Optimal timing

A time discrete trajectory optimisation framework like AICO implies the assumptions that the discretisation and time horizon are given. However, they are often not trivial to obtain. While a simple reaching task, for instance, suggests that the time to reach an object scales linearly with the distance, this is certainly not that clear in a cluttered environment. Rawlik et al. (2010) address this by, in terms of AICO's graphical model, introducing a per-step latent variables which reflect the time discretisation; then marginalising controls and, in an EM way, optimising iteratively for time parameters and for optimal trajectories.

Related to marginalising out control is an optimisation trick we use later which is also employed in Zarubin et al. (2012): compute a posterior over the final state of the trajectory while ignoring other than task and control costs. It is used as suggestion for the optimal final pose and gives rise to a backward message propagated in AICO which leads to faster convergence.

5.1.3 Summary

While wrench space and closure notions are strong arguments when coupled with reliable precise models, they quite isolate the final pose from the whole process of grasping and fall short of linking well to uncertainties. Furthermore they often require the objects to be represented as rigid bodies.

The elegant approaches around caging, e.g. exploiting object topology (Zarubin et al., 2012; Pokorny et al., 2013), need a way to fixate the object, once restricted in space, to a proper grasp (e.g. by using compliant hands, or fine grain planing). Otherwise the posture of the grasped object is barely predictable (which is the case with the simple gripper by Mason et al. (2011), too).

Compliant hardware (Deimel and Brock, 2013), accounts for large variability on the small scale by employing intelligent materials. However they take reasonably fortunate placement of the object for granted and ignore the reach phase.

The reviewed visual methods by Saxena et al. (2008); Calli et al. (2011) are strongly tied to this one modality only.

Planning trajectories with binary collision states (free/occupied) miss the variability in the notions of what is risky in highly interactive a task like grasping. Furthermore, traditional (graph/grid) planning addresses path optimality, while robot systems demand more complex notions of optimality.

While grasping GPISP does not cure everything, it has a holistic view on the grasp process: both in addressing the whole trajectory and in fusing all grasp relevant tasks. Furthermore, it naturally extends to uncertainty, as discussed in the next chapter. Once again, for the most part, the underlying representations are to blame for what the grasping approaches facilitate or lack.

The last paragraphs of Section 5.1.2 deal with settings which coherently combine different tasks. In this framework, staying collision-free is one such task among all relevant for grasping. Its particular realisation can vary. In the following section, we reason about how different parts of the world and the

kinematic chain suggest different time dependent constraints, i.e. we propose a particular realisation of the task.

5.2 Collision avoidance for manipulation trajectories

The continuous measure for collisions allows to modulate trajectories depending on current collision state. Single number as collision state might be enough for mobile platforms or simple kinematic; it is too simple, though, for most manipulators. It misses the point that in a manipulation task different parts of the kinematic chain have different influence on achieving the goal. In a typical scenario, only certain body parts are expected to manipulate the object, e.g. for a grasp trajectory this is usually the robotic hand at the end of the kinematic chain. If it misses the object, this is a failure. On the opposite, if other body parts hit the object, this is a failure as well. A single distance for measuring collision state cannot capture the diverse meaning of proximities between different parts and objects. If one tries to impose a restriction on the trajectory of the hand, it may directly drive the arm in wrong direction, e.g. in the last steps of a trajectory the fingers need to contact the object while the rest of the robot body stays collision-free. Thus, manipulation needs more transparent view on the collision state and finer cranks to tune.

In addition, for a single body part during the execution of a trajectory, the importance of staying collision-free or maintaining a safe distance to obstacles can vary in different phases of the trajectory. For instance, when grasping an object, most of the time one would consider the target object an obstacle because, likely, the fingers need to contact the object only in the end phase (and only the fingers). Thinking in terms of optimisation of a N step trajectory, this results in a collision importance profile for each body part. The importance value goes into the objective function and ensures the corresponding collision constraint is taken care of. These are far too many parameters to be feasible to deal with: a value per body part and time step. An approach to reduce

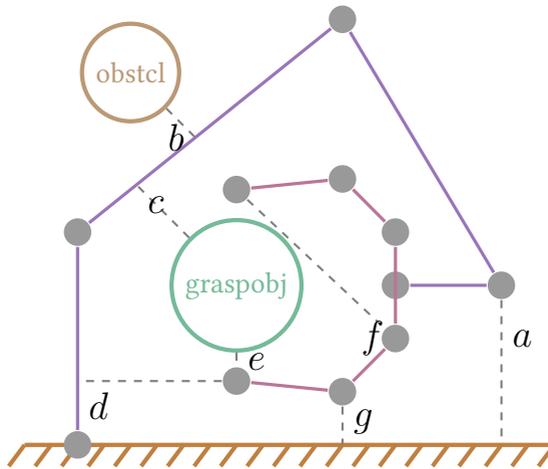


Figure 5.4: Sketch of the pairs of object which may collide during grasping trajectory, clustered by function.

the complexity is to group the bodies by purpose with respect to the task at hand. Figure 5.4 shows such clustering specific for the task of grasping an object in cluttered environment: it distinguishes arm parts, hand parts, grasp object, other objects (obstacles), table/base. With such abstraction, collision constraints need to be specified among the groups only rather than for each pair of bodies separately. This, on one hand, gives flexible enough control on collisions during execution, on the other hand reduces the tunable parameters. In the following, first we look separately into several cases which can differ in general grasp trajectories, but then it turns out that for our needs most of them can be unified to more compact ones.

a) Collisions between the static environment and the robot arm are constantly undesirable throughout the whole trajectory. Coming back to the stiffness argument from above, in terms of safety, this is a collision case which needs to be avoided. Usually joints at the beginning of the kinematic tree are stiffer than the ones at the end, e.g. so is the human arm.

- b) Collisions between movable objects and arm might be less critical for the robot's safety, but can cause harm to the objects. The effect depends highly on the context. Generally, the robot avoids unintended interaction.
- c) With respect to the arm a grasp object is an obstacle – it does not require special treatment. To name an exception, a reactive controller possibly can tolerate collision; still it could be difficult to recover since not only the control is harmed, but the target changes unpredictably its pose as well.
- d) Collisions between hand and arm need to be avoided for hardware's safety.
- e) Collisions between hand and grasp object. This is the first place we encounter time dependent importance: while most of the time collisions need to be avoided, the near the trajectory goes to its end, the more the safe distance becomes smaller and the hand shall eventually contact the object.
- f) Self collision of the hand parts needs to be avoided during the whole trajectory. The parametrisation is quite different, though, from the above cases: the hand is usually more dexterous than the arm (its parts reside near the leaves of the kinematic tree), and operate on closer distance. In effect, the safe distance decreases with execution time, to allow the fingers to get the optimal positions; in the same time, in order to compensate for the fragility of the situation, the importance of not colliding (staying at any rate at safe distance) can be risen.
- g) Collision of hand parts with the static environment are equally important throughout the execution. The arguments for the arm parts apply here.

Note that some groups have all about the same profile, so we can combine them and their relationships, and arrive at five grasp specific collision profiles for trajectories:

- Among arm parts,
- arm \leftrightarrow world + target,
- hand \leftrightarrow arm + world,
- hand \leftrightarrow target,
- among hand parts.

Each of these poses a constraint, a task to be taken care of when grasping. The task is to keep minimal the sum of all pairwise distances between the groups (or inside a group). Thus, each task penalises the bodies of a group going too near to the bodies of the other group (respectively inside the group). The collision handling introduces a few new parameters which give more detailed control on the safety aspects of a grasping trajectory. The parameters are the importances for the above groups not to collide per group and trajectory step. This continuous measure for the risk of collision can be used as a motion feature for optimisation or feedback control.

5.3 Potential field grasping

In the following we present a geometric approach to grasping which operates on a GPISP representation of an object. It generates fluent trajectories without strictly separating reach from grasp motions. This is achieved by introducing several task space objectives and controlling their relative importance based on the GPISP potential interpretation. In other words, the objectives are active throughout the whole trajectory; only their associated importances vary in different stages of the trajectory.

5.3.1 Potential properties

When we estimate a function in Chapter 4, beyond the implicit surface property, we take special care that it takes particular form. Function values shall

monotonously increase with the distance to the surface. This facilitates the interpretation of the gradient as a potential field induced by the object. The function is very close to the initial μ_0 almost everywhere, only in the vicinity of the object it starts falling more steeply to reach the surface. The derivative observations reproduce the direction of the normal. We force an observed normal to be unit length, i.e. the three partial derivatives' squared sum is one. This causes the function to resemble identity close to the surface (for properly chosen μ_0). Once more, we list the properties we use for designing the controller:

- The gradient points always in the direction of higher potentials. Thus, following the negative gradient we near the object by going to lower potentials. The magnitude of the gradient varies, but it is preferable to use the direction only.
- The function value at given point hints at the distance to the object. In the vicinity, it is quite accurate. The farther it is, the worse the estimate. In fact it is bound by μ_0 . Using this property for navigation is reminiscent of a finding by Körding and Wolpert (2004) that the cost at larger distance saturates, while being quadratic near the goal. The potential value can be used to modulate motion features during the trajectory, decoupling the notion of trajectory phase from time. For manipulation tasks, the time spent moving is less informative than a quantity which is rooted at the target, the end, of the task.
- The slope beginning to get steep can be seen as hint that the object is near. The notion of *near*, changes with varying covariance width. It can be either coupled to and suggested by the embodiment or can be set as parameter of the controller to force it to be more careful or more agile.

5.3.2 Heuristics for feasible grasps

Using the MAP estimate of the Gaussian distribution over functions interpreted as potential field we can define task space features which describe feasibility heuristics for grasping, e.g.

- move wrist towards the object,
- orient wrist with the inner side to the object,
- bring fingertips to the surface,
- orient each finger to align with the surface locally, thus maximizing the contact area.

These features, weighted according to their importance, define a criterion for a feasible grasp. The optimum of the so designed function provides a good guess for how to grasp an object. The importance of the features changes during the execution of the trajectory – the potential value is used to modulate them.

5.3.3 General approach

The implicit potential provides the information we outlined in Section 5.3.1 to guide a reach and grasp motion. Combining them we can define desired infinitesimal motions for the hand, fingers, and finger orientations. Roughly, the controller can be summarized as follows:

- Far from the object ($f(\mathbf{x}) \approx \mu_0$) the robot should direct the hand towards the object, move the hand towards the object, and open the hand.
- In the vicinity of the object ($0 < f(\mathbf{x}) < \mu_0$) the robot should move the fingers onto the surface and align the fingers with the surface tangent.
- At the surface ($f(\mathbf{x}) \approx 0$) the robot should close the fingers until pressure sensors indicate the desired contact.

Using $f(\mathbf{x})$ to blend between these task settings leads to a smooth grasp motion without ad hoc separation in strictly isolated phases. This strategy is realised by defining respective *task variables* (sometimes called end effector variables or (differentiable) motion features, introduced in Section 2.2.2) for the hand and fingers and computing the motion as the optimum of a squared cost minimization problem – a straight-forward extension of operational space control for multiple (regularized) task variables. In the following, this optimization problem is described by defining in detail the task variables: how they contribute to an objective function which results in a controller.

5.3.4 Task variables for reach and grasp

The controller we develop is motivated by the Schunk arm with the 3-finger humanoid Schunk SDH hand available at our lab. The methods are not specially designed for a three finger hand or particular kinematic structure, though.

We introduce one task variable for wrist orientation, one for wrist position, a tv for tip orientations and a tv for tip position. In addition, other tvs implement collision and joint limits avoidance. A tv for the skin sensory response is used to eventually control the pressure on the object.

In the following, $f(\mathbf{x})$ is the mean of the GPISP; the forward kinematic function that computes the position of a body part a is denoted $\phi_a^x(\mathbf{q})$; the kinematic function that computes the z -axis of the body frame in the world frame (the orientation of a) is denoted $\phi_a^z(\mathbf{q})$. If typeset in bold face, lower case letters denote vectors, capitals denote matrices. The bold index of a vector, as in \mathbf{x}_i , reads *the i -th vector in the series $\{\mathbf{x}_1, \mathbf{x}_2, \dots\}$* , while \mathbf{x}_i denotes the i -th component of vector \mathbf{x} . Task variables' indexes (mnemonics for their semantic) are not bold face, i.e. y_o denotes the (scalar) 'opposing-fingers' tv; $\mathbf{y}_s^* = (\mathbf{0})$ reads *the target of the (vector-valued) 'fingers-on-surface' tv is the vector of zeros*.

A tv suggests to specify four items:

- An *update rule* to compute the current value. This is the very definition of the task feature.
- The update rule is required to be differentiable. Having the *derivatives* of all features allows to compute the gradient of the whole optimisation problem which can be followed to local optima.
- A *target value* (or reference). Note that a single target value does not imply single optimal point. For instance, requiring the inner product of two 3D vectors to vanish implies that they are orthogonal and the set of solutions takes a whole 2D manifold, a plane.
- *Precision*, how strict is the target to be taken. This is the inverse regularisation, i.e. the inverse of how far is the constraint allowed to be violated.

5.3.4.1 Task variables with GPISP

We proceed by defining the features and the corresponding targets which implement the grasp heuristics.

- We use the idea of navigating to an object driven by the potential field it induces. With smaller values of $f(\mathbf{x})$, the speed of approaching the object should be reduced, eventually becoming zero at the surface where $f(\mathbf{x}) = 0$. The right notion of ‘near’ ensures that the slowing down does not happen too early to be ineffective, neither too carelessly late. For this stands the hand-position task variable (read the following row as: domain, update rule, target):

$$y_h \in \mathbb{R}, \quad y_h = f(\phi_h^x(\mathbf{q})), \quad y_h^* := 0. \tag{5.1}$$

- In order to allow the fingers to grasp an object, the wrist needs to approach the object with the front side oriented to it. We noted in earlier

sections that the negative gradient points to the object. Here we require the wrist normal to align with $-\nabla f(\mathbf{x})$. After taking care of that in the beginning, the relevance of the argument decreases with lower potential values, i.e. when approaching the object, violating this in favour of better finger configuration is favourable. Meet the hand-axis task variable:

$$y_p \in \mathbb{R}, \quad y_p = \langle \phi_h^z(\mathbf{q}), \nabla f(\phi_h^x(\mathbf{q})) \rangle, \quad y_p^* := -1.$$

- In order to grasp, the fingers need to touch the surface of the object. This can be expressed in terms of GPISP as the requirement that the potential value at the finger contact points is zero. This is implemented as the task variable (with constant target)

$$\mathbf{y}_s \in \mathbb{R}^3, \quad \mathbf{y}_s = (f(\phi_i^x(\mathbf{q})))_{i \in \text{tips}}, \quad \mathbf{y}_s^* := (\mathbf{0})_{1:3}.$$

Here i indicates any of the finger tips. This is equivalent to a tv per fingertip. Since they share common precisions during time, it makes sense to have a single entity instead.

- When the *finger tips point to their common centroid*, we expect to encounter stable grasp (This can be seen as analogous to form closure argument). In a three-fingers setting the length of the vector sum of the normals of the finger tips lives in $[0, 3]$, with zero reached if the fingers oppose perfectly. In other words, zero sum of normals is necessary condition for opposing fingers. The opposing-fingers task variable is

$$y_o \in \mathbb{R}, \quad y_o = \left\| \sum_{i \in \text{tips}} \phi_i^z(\mathbf{q}) \right\|_2, \quad y_o^* := 0.$$

- *Large contact area* is achieved if the tangent of the shape is aligned with the finger tangent at the contact point, i.e. the two normals cancel.

Since the normals of the surface coincide with the negative gradient on the surface, the condition insists on the fingertip's normal and the field gradient at the fingertip to be aligned. The importance of the orientation relative to the surface rapidly rises the more the surface approaches, thus here the precision is negatively proportional to the potential. This introduces the task variable

$$\mathbf{y}_f \in \mathbb{R}^3, \quad \mathbf{y}_f = (\langle \nabla f(\phi_i^x(\mathbf{q})), \phi_i^z(\mathbf{q}) \rangle)_{i \in \text{tips}}, \quad \mathbf{y}_f^* := (-\mathbf{1})_{1:3}. \quad (5.2)$$

The Jacobian of this tv is derived in Appendix A.3.

Note that the above tvs are defined in terms of the potential. All scalar features as well as all components of vector features are sums or products of f and ∇f . They depend on \mathbf{q} only through various ϕ functions. The derivatives of the forward kinematics functions are retrievable at any time, since every ϕ is a chain of homogeneous transforms. Thus, it suffices to give the derivatives of the features with respect to f and ∇f and the derivatives of f and ∇f with respect to the position and orientation arguments ϕ^x and ϕ^z . The first part is easily done as derivatives of sum of products. For the second, we have the option to numerically approximate or find a closed-form expressions for the particular potentials. The latter requires one more differentiation beyond the already available gradient: the Hessian matrix of second derivatives, $\nabla \nabla f$, is involved. Appendix A provides it for the GPISP mean and for quadratically increasing potentials induced by few analytic shapes: sphere, cylinder, box.

5.3.4.2 Avoidance of collisions and joint limits

Besides the features which use the GPISP mean there are task variables to avoid collisions and joint limits which are independent of the GPISP. The fact that potentials are not employed here for what they were originally used for – the collision-free navigation – may appear somewhat confusing and demands

explanation. Mobile platforms operating in 2D as well as obstacles in such applications mostly have a symmetrical regular shapes. The links of a robot arm become increasingly complex and it is likewise hard to come up with a potential which describes the true shape of the body parts and is efficiently computable. An approximation by analytic objects may become too crude on the scale a hand operates (think of the shape of the fingers). On the other hand, very precise mesh models of the parts are available from the hardware design process anyway. Thus, we implement collisions as a TV defined on proximities between objects. Below certain threshold, distances start to introduce quadratically increasing costs and in such way penalise the pair of objects getting too near. Similarly, the costs for moving near joint limits is defined directly in configuration space.

5.3.5 Objective function

The precision of a TV expresses its relative importance compared to the other TVs. For instance, for the orientation of the palm the precision decreases with decreasing potential: $\rho_{y_p} := c_p f(\phi_p^x(q))$ with some constant c_p . An intuition for the precision from optimisation viewpoint is that its inverse can be interpreted as a measure of how far the corresponding constraint can be relaxed in order to obtain better value.

The accumulated loss introduced by not satisfying the constraints poses an optimisation problem within the robot configuration space. The loss of a TV is given by the discrepancy between current and target value weighted by the precision, $l_a = \rho_{y_a} |y_a - y_a^*|_2$. The sum of all losses gives the total task costs which we seek to minimise in order to get a good grasp. In addition, each movement introduces movement costs which need to be accounted for. For a feedback controller, optimising both costs yields the optimal direction in configuration space, $\Delta \mathbf{q}$, which is proportional to the control signals that need to be sent to the actuated DoF. In trajectory optimisation, these are the

costs for a single step of the trajectory and contribute to the total trajectory costs which decide on the optimal series of $\Delta \mathbf{q}$ over time.

The above objective function is a convex quadratic program for the optimal $\Delta \mathbf{q}$ since the differentiable features are locally linearized with the Jacobian. For the configuration space we use a metric W which can be thought of as the covariance matrix of a Gaussian distribution for the state transition. The numbers on the diagonal of W imply the agility of the particular joint of the robot. In other words, using $c_w W$ instead of W for some constant $c_w > 1$ makes the degrees of freedom less likely to move between time steps and counteracts increasing task precision ρ .

Note that here for the sake of simplicity the state \mathbf{q} contains the positions only, i.e. all velocity precisions ν_i are zero and we have no velocity targets in the task spaces.

5.3.6 Relation to grasp taxonomies

The above TVs together seek to approximate a sufficient criterion for grasping unknown objects without being unnecessarily restrictive. An illustration of this is that they capture feasibility properties which hold for different parts of a grasp taxonomy. Some of them do not need intervention, some others need to change parameters of the controller. The taxonomy proposed by Feix et al. (2009) organises grasps depending on finger opposition type, virtual fingers, thumb abducted/adducted, power/precision. Defining correspondence between postures of different kinematic chains is not trivial. For instance, researchers engaged in translating human experience for robotic hands concern themselves with such questions. Some properties are nevertheless easy to extract. Consider power grasps as contrasting to precision grasps. The former suggest a contact between the palm and the object; the thumb opposes the other fingers; if the object allows, they in addition all oppose the palm. In contrast, precision grasps exclude the palm from interaction with the object;

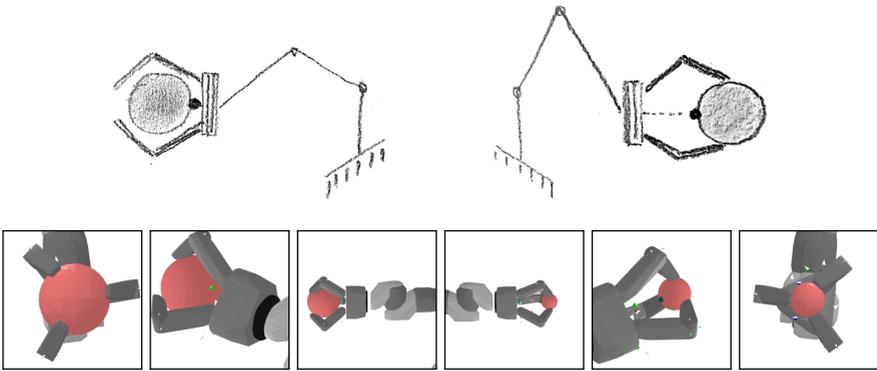


Figure 5.5: Power grasp (left) and precision grasp (right) achieved by proper choice of the grasp center, i.e. the virtual particle attached to the wrist, which is moved by the field.

the contacts are roughly in a plane, rather than what is the case when caging, i.e. friction gets more important for a successful grasp. How does this relate to potential field grasping? The positional TV y_h instructs the hand to go to lower potentials. However, we have not so far specified which point exactly computes ϕ_h^x in Formula 5.1. While the hand is far away this does not matter a lot. When the hand arrives in the vicinity of the object, it *does*: the *grasp center*, ϕ_h^x , decides how the grasp eventually looks like. The grasp center is the ‘particle’ which the surface potential field acts on. The last transformation in the chain ϕ_h^x are the coordinates of the point in the frame of the hand. As grasp center can serve a point on the palm surface – enforcing a power-grasp-like configuration; alternatively, the grasp center can be at about a finger length away from the palm for precision-grasping a small object. This is illustrated in Figure 5.5.

Another distinctive property in the taxonomy is which side of the finger makes the contact. As with the palm, we can choose where to look for contact by appropriately computing the finger tips with ϕ_i^x in Formula 5.2. Note that in terms of functionality a pinch grasp with opposing thumb and index finger

is the same as a sideways pinch grasp between the index and the middle finger (provided the fingers are symmetrical, which is often the case with robots, and not the case with humans).

Unlike the above properties, in terms of GPISP we cannot distinguish thumb abduction from adduction. If the trajectory ends up in adducted thumb configuration, this is not because a TV enforces it directly. There is no task dedicated to it. It is rather the result of the optimisation claiming that the particular configuration leads to lower costs of the overall criterion: larger contact area, better opposing fingers, etc.

Note that grasp taxonomies are usually restricted to poses. As mentioned in the Introduction, grasping is not a single moment. Same ‘end’-grasps can be produced as a result of complex manipulation chains (e.g. first sliding to the edge of a table to make part of the book free and then pinch-grasp, as opposed to directly grasp). Finding a systematisation of grasp trajectories might reveal important additional aspects.

5.3.7 Relation to synergies

It turns out that despite many in principle actuated joints of the human hand our brain – most of the time – effectively decreases the controllable DoF by coupling them. This is easily observable when trying to move independently the last two joints of a finger. These *synergies* can be obtained, for instance, from recorded human grasps by embedding the posture features in a lower-dimensional manifold suggested by PCA or similar techniques. Robotic scientists employ such synergies to design more effective controllers for robotic hands which nevertheless span wide range of useful configurations (Todorov and Ghahramani, 2004). The GPISP features offer another perspective on parametrising synergies – rooted in the interaction, rather than from the viewpoint of the kinematic structure.

Along this line is another property of GPISP as object representation: tra-

ditionally, one would parametrise a grasp by the point at which to grasp and the approach direction, i.e. the pre-grasp orientation and position of the hand. Interestingly, orientation and position can be seen as dual under GPISP. A position of the hand determines unique orientation: aligned with the negative potential gradient. An approach direction suggests typically only a few suitable positions (for a sphere or ellipsoid it suggests exactly one, but for a box it might suggest the whole side of the box and other criteria might be needed to make it unique).

5.3.8 Potential affordance

The notion of affordance comes from the 70's psychology research (Gibson, 1977, 1986). Through developmental psychology and developmental robotics it has found its way into grasping and manipulation research (Stoytchev, 2009; Sinapov and Stoytchev, 2007). This comes natural with the insight that building intelligent machines goes through autonomous learning and interaction with their environment. In an attempt on a short and loose explanation, an affordance is a perceivable property of an object which describes a possibility, opportunity to perform some action with it, something it is qualified to be done with. For instance, all chairs share the affordance of sitting in them, but pile of books or a table afford that as well. In these terms, an affordance relates the capabilities of an agent with the capabilities of an object. Grasp affordances determine how we grasp an object – a cup handle makes itself a good candidate, but the cup can be grasped from the side with a precision grasp, from the top or with a pinch grasp at the wall. This thesis' surface potential attracts and orients fingers and wrist to the surface of an object, thus GPISP can be seen as constructive representation of a grasp affordance. An interesting aspect of that viewpoint is that the models can be built (learned) and refined from sensor data. They evolve with experience – do not exist a priori – just like some motor abilities which evolve in the process of interacting with

the environment.

5.3.9 On feed back and feed forward

A controller is a mapping from world states to actions, i.e. to control signals to the actuated DoF. We slightly depart from this definition and call controller our movement prescriptions – discretised trajectories in configuration space. This is justified because the $\Delta\mathbf{q}$ between steps with vanishing step size is often proportional to the control signals needed to move from one step to the other. Further justification to work with prescriptions is to abstract the particular way the hardware is controlled. The optimisation gives its prescription, the hardware moves, and then a feedback controller implicitly accounts for deviations by appropriately setting the reference for the next step. We stick to saying controller to both an algorithm producing the next step from the current state, and to an optimised discrete trajectory.

This view gets support by how Todorov (2004) uses the terms in his comprehensive review of optimality principles in sensory-motor control. He is concerned with the potential of optimisation approaches to predict and thus explain large class of empirical phenomena by different objective functions. Todorov argues that open-loop optimisation models capture average optimality, i.e. the behaviour averaged over multiple trials or subjects. The choice of objective function varies across total energy consumption, smoothness penalties on accelerations (jerk), variance minimisation and depends on the modeled task. For instance, in settings requiring maximal effort the cost results directly from the task, while in more complex cases more opaque cost is needed to account for the particularities of the system and task (jerk, end point variance). Ignoring feedback, such models are inappropriate for predicting phenomena with larger variability. Feedback becomes even more important with increasing length of trajectories since variability propagates even stronger then.

Prescriptions, i.e. feed forward trajectories optimised prior to execution,

need to be followed eventually by a feedback controller which takes the steps as attractors (the denser the steps, the easier is the job of the follower). This may make the difference between close and open loop appear small. However, the difference should not be underestimated: once available, the open loop trajectory becomes target in itself for a feedback controller. The feedback is only used to stick to this prescription. In closed-loop optimisation, the trajectory is prepared for the possible feedback; it takes it into account during the emerging of the trajectory. As result, we may miss intuitive explanation for the revealed control laws, or as Todorov points out, ‘In less trivial tasks, however, the optimal control scheme will generally be one that we do not yet have a name for. Such flexibility, however hard to grasp, matches the flexibility and resourcefulness apparent in motor behavior.’ And, as he illustrates by a vivid quote by Bernstein (1967), biological systems can then ‘solve a control problem repeatedly rather than repeat its solution’, coping with the stochasticity of the world. Even stronger statement follows: tracking an open-loop trajectory or an averaged closed-loop trajectory is necessarily suboptimal since the individual closed-loop trajectories are always better aligned to the particular noisy instance of environment.

Further observation therein is that in an uncertain world, solutions of the optimisation problem are optimal only when the state estimate is optimal. This is directly observable in the closed-loop case which yields a policy, a mapping from state to controls.

5.4 Potential field feedback controller

Using the tvs for feedback control involves on every feedback step (i) computing their values, (ii) solving the optimisation problem to obtain $\Delta\mathbf{q}$ and (iii) making the step. Cycle by cycle, this procedure is repeated until a termination condition is triggered. The termination can be an outside event, e.g. due to a registered contact at the fingertips with the surface or due to a higher level

controller changing its policy and handing the control to another procedure. But a condition can be derived also from the objective function itself, from the costs level reached in the current step. A typical termination criterion would be linked to the task costs rather than to the total costs because the former are the part which decides how satisfying the current state is w.r.t. the goal. Note that cost thresholds scale with TV precisions and maximal potential values (ρ and μ_0). Thus, the transfer of thresholds between different tasks and situations should be done carefully.

For a suitable step length, the feedback control behaves as a gradient descent optimisation in the configuration space: each step brings the system \mathbf{q} nearer to a local minimum. (Inappropriate step size has the same consequences in both optimisation and control.) This shows why a feedback controller can be trapped in bad local minima, which is the case with all approaches relying on local information only. Alleviating this effect in a feedback setting is a matter of designing an objective function with a benign landscape and/or choosing adequate initialisation.

An advantage of feedback control is that it is highly reactive. If an aspect of the environment which is taken account of in the controller changes, the reaction of the system is immediate and seamless. As an example, if in certain step the grasp object moves, the potential at the end effector changes, then the GPISP TVs correct their gradients, which changes the objective function gradient, and eventually results in a different $\Delta\mathbf{q}$. On the downside of the responsiveness, if for instance an object disappears due to some sensors failure, this might lead to discontinuity – such cases must be taken care of.

Note that in the cost term $\rho \|y - y^*\|_2$ changing ρ is equivalent to changing the distance to the target by $\sqrt{\rho}$. However, since targets are associated with certain semantics, it is preferable to alter precisions.

Completing a grasp. For a GPISP feedback trajectory, there are several options to complete the grasping. For a potential going very near to zero the cost term

for the finger tips potential vanishes and the fingers may never contact the object. To cure this, one can define the finger tips slightly into the fingers or set negative potential target for \mathbf{y}_s^* . More sophisticatedly, if the task costs are low enough but still there is no contact, a new TV may start moving the finger tips along their current normal (which are supposedly aligned with the object surface, as indicated by the low costs). Algorithm 1 shows the feedback control

Algorithm 1 Grasping with ISF.

$y_h^* := 0$	▷ hand potential
$y_p^* := -1$	▷ hand orientation
$\mathbf{y}_s^* := (\mathbf{0})$	▷ fingers' potential
$\mathbf{y}_f^* := (-\mathbf{1})$	▷ fingers' orientation
$y_o^* := 0$	▷ fingers oppose
loop	
$y_h = f(\phi_h^x(\mathbf{q}))$	▷ current potential of grasp center
$y_p = \langle \phi_p^z(\mathbf{q}), \nabla f(\phi_k^x(\mathbf{q})) \rangle$	▷ current orientation
$\mathbf{c} = (\mathbf{0})$	▷ centroid: sum initialisation
for $k \in \text{fingers}$ do	
$\mathbf{y}_{fk} = \langle \nabla f(\phi_k^x(\mathbf{q})), \phi_k^z(\mathbf{q}) \rangle$	▷ finger tip orientation
$y_{sk} = f(\phi_k^x(\mathbf{q}))$	▷ finger tip potential
$\mathbf{c} = \mathbf{c} + \phi_k^z(\mathbf{q})$	
end for	
$y_o = \mathbf{c} _2$	
set $\rho_h, \rho_p, \rho_o \propto f(\phi_h^x(\mathbf{q}))$	▷ precision decreasing near surface
set $\rho_f, \rho_s \propto (1 - f(\phi_k^x(\mathbf{q})))$	▷ precision increasing near surface
$\mathbf{q}' := \underset{\mathbf{q}}{\operatorname{argmin}} \sum_{y \in TV_s} \rho_y y - y^* _2 + \mathbf{q}^T W \mathbf{q}$	
$\Delta \mathbf{q} := s(\mathbf{q}' - \mathbf{q}) / \mathbf{q}' - \mathbf{q} _2$	
$\mathbf{q} := \mathbf{q} + \Delta \mathbf{q}$	▷ move
end loop	

loop of the reach and grasp controller. Typically, the trajectories generated by the algorithm start with a motion towards the object, simultaneously putting the wrist to point with the finger side to the object. When the fingers arrive

near the object, the potential falls rapidly and the relative importance of finger orientation rises with the consequence that the fingers open. Then the fingers continue to approach the object guided by the potential field.

5.5 Potential field trajectory optimisation

In the case of feedback control, the optimiser has access to the world features on every time step and computes the control signal accordingly. This section uses the heuristics and optimality criteria from Section 5.3.3 in a trajectory optimisation setting, giving a procedure which yields a controller for the whole trajectory. The result is a path in configuration points, i.e. a series of robot configurations $\{\mathbf{q}_i\}_{0:N}$. It is optimal with respect to the designed objective function which, as in Section 5.3.5, needs to account for (i) performing sufficiently well some task and (ii) doing this with minimal effort. There exist at least the following two ways which conceptually differ in the point whether one specifies GPISP criteria for the whole trajectory or only for the grasp posture.

5.5.1 Reach and grasp from potential

Like with feedback, some precisions and targets may vary dependent on the potential at the end effectors. In the reactive controller, the precisions and the targets of the next step depend on the potentials in the current step, e.g. $\rho_h^{(i+1)} \propto f(\phi_h^x(\mathbf{q}^{(i)}))$. In the feedback case, the i -th state is available when calculating the next one. Here such construction is somewhat ill-posed since the optimisation is over the space spanned by $\{\mathbf{q}_i\}_{0:N} \in \mathbb{R}^{N \times |\mathbf{q}|}$. One can come up with an optimisation algorithm which is aware of the structure of the problem. This is related to having the proper priors over trajectories – a practical choice would be some problem specific linear state transition with Gaussian noise (LQG). (Effectively, this is a Gaussian process prior over trajectories. This

is beyond the scope of this section. The merit from strict intermediate goals is questionable as argued in the following paragraph.)

Previous sections pointed out some limitations of feedback control. This way of designing optimisation function effectively has a strong bias towards possible deficiencies of the feedback trajectory. Instead, one can benefit from having global knowledge and produce a trajectory which does not suffer from the limitations. The intuition behind this is that the most part of the trajectory is better governed by efficiency. The more the aim nears, the more important to strictly obey the prescriptions of the grasp posture criterion. This idea is pursued with the alternative approach in the next section.

5.5.2 Reach from movement costs, grasp from potential

In a snapshot of the GPISP TVs' precisions and targets in the last part of a successful feedback trajectory, the parameters would describe a posture that is good in terms of the grasp criterion (think of it as a quality measure). If an oracle could provide this final posture, the optimal trajectory would be matter of collision-free navigation from the start posture. This is usually the scheme optimised trajectories are produced in literature (s. Section 5.1.1). The final posture is result of optimisation with respect to a grasp criterion. The criterion is often derived from wrench space analysis for the simulated robot and the current object model. Here the criterion is heuristic and is contained in the task costs of the GPISP grasp objective function.

Technically, the benefit of posing this optimisation problem compared to the previous section is the elimination of weird local minima which may arise due to complex geometry of the potential field or due to non-optimal parameter choice.

The oracle providing the final posture is not aware whether the configuration is reachable form the start. It would certainly take account of the kinematic chain and thus produce a feasible posture, but cannot guarantee

that there exists a path in configuration space between start and final posture. This question can only be resolved once a complete trajectory is available, i.e. when the optimisation is more or less done.

It is certainly unfeasible to take into account all aspects, e.g. finding a collision-free trajectory, at this stage because it is too expensive to test collision for every prospective trajectory. But it makes sense to make a ‘light weight’ feasibility checks. One can use the assumption that consequent steps are correlated: the configuration in step $i + 1$ is restricted to some neighbourhood of the i -th step’s configuration. This allows to estimate where the robot could possibly be after a number of steps. Thus, one can ensure that the final posture is reachable in terms of the structure and dynamics of the system. This is related to marginalising time while integrating costs as found in the work by Zarubin et al. (2012) in the context of AICO.

It is too expensive to compare and choose the best of all possible robot configurations with fingers touching the surface. Instead, one can define a partial optimisation problem as starting from an intermediate position which is likely to lead to successful grasp and go the last steps towards a local optimum. Such promising configurations feature the hand close to the object, oriented towards the surface and with fingers wide open. Since this optimisation is cheap compared to the whole process of trajectory optimisation, one can afford to test different intermediate positions in order to find better local optimum. We restrict the global orientation of the wrist to several values, which results in different poses. This translates to different approaching directions, e.g. top grasp or side grasp. Summarised, the above boils down to two cheap optimisations which split the problem into easier subproblems: in a first step, find an optimal intermediate configuration which is restricted to some approach direction, the hand is near the surface and properly oriented to the surface with open fingers. In a second step, starting from the intermediate position find an optimal grasp. Doing these two steps for different approach directions results in several local optima to select the best one from. Figure 5.6

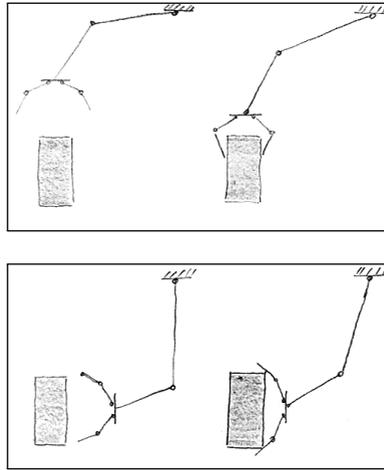


Figure 5.6: The two stage optimisation process: First, care most about approach direction and find hand posture with wide open fingers; second, start from there and optimise for the complete grasp criterion; then, compare the final task costs of individual candidates. The upper row corresponds to approach direction from above, while the lower row corresponds to approach direction from the right. In this sketch, the upper candidate grasp is more promising, since it appears more stable.

shows a sketch of intermediate optimisation results with different approach direction and the resulting final postures. Note that this is merely an optimisation trick. Its result is similar to having reach and grasp separated in the first place. However, here the emphasis is on being aware of the relatedness of reach and grasp, considering them together. Only for the sake of coping with a hostile optimisation landscape some tasks are assigned less importance.

Once a final posture is available, how to move collision-free from the current position to it? We employ approximate inference control – a linear optimisation method that propagates beliefs in the graphical model of a discrete trajectory.

5.5.3 Optimisation heuristics

The optimisation problem searches for a point $\{\mathbf{q}_i\}_{0:N} \in \mathbb{R}^{N \times |\mathbf{q}|}$ with least costs in terms of the objective function defined by the tvs and the movement costs. The result depends on the parametrisation of the objective functions and on the optimisation algorithm. It would likely be close to a local optimum.

Section 5.3 gives the interpretation and semantic of the parameters of the tvs (which define the grasp criterion) and movement costs (responsible, roughly, for the energy efficiency). Summarising the particular heuristics used for the optimisation: we split the problem in smaller optimisations and argue that this avoids high-dimensional nasty cost landscapes with bad local minima. Furthermore, we let available information about start and end postures impact the intermediate trajectory:

1. Find final posture:
 - (a) Use approach direction as a sampling parameter. Find intermediate promising position for this approach direction. This is done by optimising only the approach direction and position of the grasp center ($\mathbf{x} \in \mathbb{R}^6$, $y_h \in \mathbb{R}$, $y_p \in \mathbb{R}$) in the GPISP field. Ignore all other variables. Once arrived at the optimum, open the fingers. (One can interpret this posture as a pre-grasp.)
 - (b) Taking the above posture as initialisation, optimise according to the full grasp criterion. This yields a candidate final grasping posture.
 - (c) Choose the candidate with lowest task costs.
2. Use the final posture and the start posture in the initialisation of the optimiser, i.e. as \mathbf{q}_0 and \mathbf{q}_N in $\{\mathbf{q}\}_{0:N}$. This is a hint, bias for the approximate inference control employed for solving the large optimisation problem. For AICO the controller is a graphical model and messages are

passed until convergence to a local minimum. The forward message propagates the system dynamics. The backward message propagates the bias towards the final posture. Since it is a local method, it depends on a good initialisation.

3. The movements in the final stage of such trajectory are trickier than at the start. Usually more joints are moved considerably (i.e. larger values in the Jacobian), the task demands more precision, the environment is more complex. Optimising the reverse trajectory allows the algorithm to faster descend to feasible costs; i.e. the trajectory ‘starts’ at the final posture and ‘ends’ in the current one.

5.6 Validation

The current chapter is about demonstrating that the GPISP representation is naturally suitable for manipulation tasks, in particular for grasping. The conceptually simple TVs which operate on GPISP features describe a grasp criterion and contribute to an objective function which can be used to produce grasp trajectories. Thus, the validation boils down to the question whether the grasp criterion is feasible. This is the case when there is a parametrisation of the TVs which produces valid grasps in different situations and with different objects. The feasibility of the criterion is independent of the precise way the models have been acquired. Thus, we rule out the influence of object estimation and test the approach on analytic shapes. Furthermore, we somewhat simplify the scenario by assuming that the environment is not cluttered to such a degree that a global method is necessarily needed – we focus on the feedback approach (prone to sticking in local minima like the banana problem, etc.) and do not include obstacles other than a table.

For the evaluation of the controller a robot grasps real objects in the setup shown on Figure 5.7. The robot hardware comprises the 7-DOF Schunk light-



Figure 5.7: Experimental setup: Schunk 7-DoF arm, Schunk 7-DoF hand, Bumblebee stereo camera, coloured objects with various shapes.

weight arm (LWA), 7-DoF Schunk dexterous hand (SDH) with 6 tactile arrays at the fingers, and a Bumblebee camera. Stereo triangulation is used to localize the objects and provide surface points. The test objects are a box, a cylindrical can and a ball. They are presented – one at a time – at different places on a table in front of the robot. Every object is uniformly coloured to ease their localisation. The visual perception starts with a simple hue threshold segmentation which delivers a contour of a shape. The position and the size of the object can be inferred from the form (ball, cylinder, box) which optimally fits the contour and from the depth provided by the stereo camera. This gives rise to a parametric implicit surface potential corresponding to this object. (In contrast to GPISP it is not approximated from data, i.e. bypasses the GP.)

To objects placed at different locations on the table correspond different grasp trajectories resulting from the gradient descent in configuration space minimising gradually the cost using the τ_V Jacobians. The arm starts with fast movement towards the object and orients the wrist accordingly. Nearing the object it slows down, the fingers open and align with the shape. This can be seen when comparing the ball and the cylinder. The fingers follow the symmetric shape of the ball, whereas for cylinder they stay aligned with the vertical wall (cf. Figure 5.8). Eventually, the fingers get in contact with the

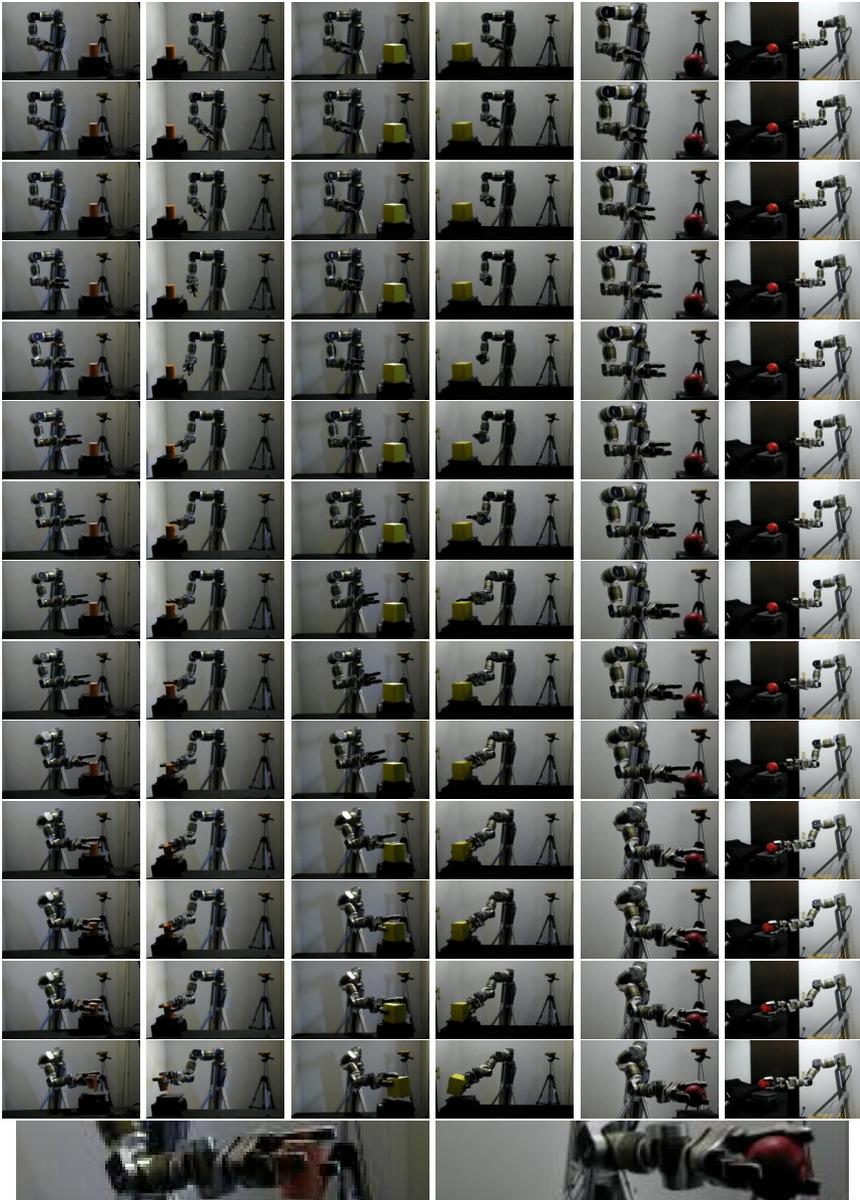


Figure 5.8: Demonstration on the real robot grasping a cylinder, a box and a ball.

surface and the skin pressure rises. When it goes beyond given threshold, the controller stops and the robot lifts the object as an indication for successful grasp.

In some cases, the grasp does not succeed in the real execution. There are two possible explanations for that:

- The controller does not manage to produce a feasible grasp according to its model, it fails already in simulation, i.e. it is not correct in itself, fails to capture the features of the model well.
- The grasp looks fine in simulation, but the model differs from the real object to a degree that the controller is not able to tolerate.

Given the simple setup without obstacles, the controller was able to produce a fine grasp in its perceived world, and failures from the former type are rarely encountered. The grasp criterion was able to enforce an optimum which eventually orients the fingers and hand in the right way. Furthermore, modulating the intermediate precisions according to the potential field leads to a smooth reach trajectory, naturally introducing the final grasp.

The experienced failures to grasp were from the latter kind: the error in the size and estimated position were beyond the controller's ability to generalize. An overestimation of the size or bad position estimate would likely lead to a collision with the side of a finger because the fingers start to open too late or are not opened enough. (One can take measures to counteract this by opening the fingers earlier and wider but this makes the movement look less natural, more inefficient and only shifts the responsibility to a final tactile driven phase.) We can conclude that the controller employs well the grasping relevant information encoded in the GPISP to produce a correct grasp for a given model but suffers from model inaccuracies. This is a conceptual problem rooted in the usual (insufficient) interaction between the system and the environment. A solution would in the first place design a controller which does not blindly trust the model or/and require hardware/system capable of gathering

more feedback during the grasp. This direction we head in the next chapter by providing means to ‘know what I know’, i.e. a measure of the quality of the model, ways to obtain it and to use it to improve control.

The controllers used in the following chapter are largely derived from this one. Thus, the criteria suggested here contribute to benefits of the methods developed later.

The most interesting real world problems suggest conditions which are too complex to be reliably predicted. In such environment, pure feed forward control is not appropriate. The trajectory which is planned in a simulated world (or retrieved from a collection) is then played on the real hardware. Since one cannot rule out unforeseen events, they should be dealt with at execution time: the system needs to react to deviations from the simulated world for variety of reasons including safety, security and precision.

5.7 Summary

A GPISP model is well suited to generate grasp motion. It provides novel task spaces which allow to specify grasp objectives conveniently. The targets in these spaces give rise to an objective function, a heuristic grasp criterion. Some of its parameters map directly to distinctive features in grasp taxonomies. It offers a shape-rooted view on motor synergies and can be seen as constructive representation of a learnable grasp affordance. In both open-loop and closed-loop context, it can be used together with other objectives to synthesise a grasp trajectory. To the end of moving collision-free, the large number of proximities between robot parts and world objects have to be monitored: an objective orthogonal to the grasp criterion. However, for the purpose of manipulation trajectories, the bodies and parts can be organised in a few groups which give the flexible cranks essential for interaction at a low conceptual price.

5.7.1 Discussion

We stress in the following some implications of this chapter, clarify the scope of our approach to grasping and point out limitations.

5.7.1.1 Hierarchies, planning

Generating sequences of motion primitives is as much planning as the sequences of controls of a motion primitive. With the right representation, all these can be done using the same methods, as illustrated by Toussaint et al. (2010). They use probabilistic inference, i.e. inferring the steps to reach from the current state the imaginary state of the fulfilled goal. In this thesis, we work on the level of primitives in the geometric world. We are tempted to think that tasks do not require higher level planning, that a feedback sensor-action-loop is – more often than assumed – enough to handle the task. Even so, this is merely a quantitative shift and there will be enough complex tasks impossible to address without hierarchical control: hierarchical sensory-motor representations have the potential to make complex behaviour tractable.

5.7.1.2 Learning

The presented approach to grasping does not so far involve learning at any level. Neither does it benefit from a teacher, transfer across different embodiments, nor does it extract features from experience. The controller is assumed fixed and correct. Thus, to the end of obtaining better grasps, we head at improving/learning the object model. The previous chapter offers the instruments to put information into the model, the next chapter discusses how to seek and obtain information. However, a desirable feature would be to integrate learning and improve the controller itself: by learning the right parameters from own experience or by transferring them from known optimal demonstrations.

5.7.1.3 Relation to closure arguments

Without being equally formal, GPISP grasping is related to form- and force-closure. The task of the fingers to oppose as much as possible is a heuristic to restrict the object all between them. The task to align the fingertips' surface with the object's surface at the contact points is directly related to maximising the friction cones and thus maximising the external wrench the grasp can resist. They are, however, heuristic and soft constraints (actually weighted objectives) in the more complex grasp objective function.

5.7.1.4 Control dependent collision avoidance

A rarely considered aspect of collision-free movement is linked to control noise. Todorov (2004) cites the Fitt's law, which relates movement duration to movement amplitude and target width – the more accurately the target should be reached, the longer it takes. He points out that the law is predicted by different feedback control models and can be explained by control dependent motor noise – faster movements are less accurate. Collision avoidance ideally should be aware of this fact. It can be incorporated into the TV framework by making the precision of the collision TVs depend on the current motion rate.

Chapter 6

Uncertainty-aware control

We perceive the world through sensors. Sensors are noisy, i.e. we cannot be certain about their readings. Yet we use these readings to build models of the world and to interact with it. Thus, these models and our actions are uncertain. But to what degree? If we knew the answer, can this help to improve our actions?

The first question above is a question of how to view and transport the uncertainty – what representation to use. It is often assumed that all the uncertain measurements are available at once. This is rarely the case. In an artificial data set, possibly yes, but not for real manipulation tasks. So another question arises: where and how do the data come from?

The synthesis of motion for particular task cannot be perfectly dependable and a controller cannot be completely correct due to sensory and motor noise: the system will necessarily sometimes fail to fulfill a task. If the task has a meaning beyond merely being a benchmark for evaluating the controller, it needs to be *completed*, not only *tried*. Thus, the most natural reaction to a failure is to start another try. A valid policy is to try again and again the same, relying on the stochasticity in the control to arrive at a successful point. This is not the smartest way since it lacks any analysis of what goes wrong. As an

example for another policy, a human could repeat the motion slower and more carefully. This can be seen as employing the same controller more precisely, assuming that the failure is due to imprecision in control. For another example, if multiple ways for solving the same task are available, a failure might be a hint that an alternative approach is more suitable. In such case, the next try might employ a different controller. All three examples above are aware of the failure. Formally, they reduce the status of a manipulation task to a binary success variable. However, there is more to learn from a failure than this: in order to come up with more elaborate control policies, one can reason about the imprecision of the underlying model or about the connections between failures. This interaction with the world is where the information can be extracted from! The implied interplay of both sensor and motor systems delivers successively the data points!

In a grasping scenario, the interaction is due to the system trying to grasp until success. To begin with, the spectrum between success and failure is much richer than zero and one. A grasp can fail in many different ways: the fingers might grasp in the void, i.e. fail to establish contact; the established contact points might be not the expected ones (still, this can be a valid grasp); the contacts might be right, but the grasp might fail nevertheless. Successful grasps can also be distinguished in terms of stability (recall grasp quality measures mentioned in Section 5.1.1). In addition, misplaced contacts could move the object and require reconsidering the situation: as an extreme example, it can fall off the table and thus create a completely different setting and constraints.

Continuing this argument in machine learning terms, the experience of an execution of a grasp is a source of information in twofold manner. (i) It is a labeled instance of trajectory; a failure suggests a negative label, while for success a multitude of labels can be derived from a quality measure or can be subsumed to a single positive label. This knowledge is valuable for reasoning about the trajectories, i.e. in the space of motion features where trajectories live. Mostly, this would be helpful with scenarios where the object model is

previously known, with fixed uncertain position. (ii) The second piece of information comes into being with the interaction with the object: if a trajectory – failing or succeeding – gets in touch with the object, the points of contact deliver observations from the surface. With the appropriate representation, this geometric information can be used to refine the object model.

After this short argument towards interaction, iteratively reducing uncertainty, and the inevitability of sensor integration in manipulation, we proceed with a review of work related to uncertainty in shape estimation, exploration and grasping. The rest of the chapter is dedicated to the question we started with – how can uncertainty awareness improve the actions we take. We present an approach which employs GPISP. Here we integrate the estimation and control tools we developed in the previous chapters to provide an interactive scheme for grasping and estimation. It finally shows how the integral view on sensory and motor systems is beneficial for both. The interpretation of GPISP as distribution over shapes and its variance as measure of uncertainty allow to formulate control laws for navigating to a high-confidence or to a low-confidence area of the object surface. Based on these control laws we introduce the concepts of *explore-grasp* and *exploit-grasp* which can be combined in higher level policies.

6.1 Related work: acting under uncertainty and using it

The research related to this chapter addresses ways to deal with different sources of uncertainty and the coupling of sensory and motor systems for grasping.

6.1.1 Probability, uncertainty

Probability theory is commonly used to reason about uncertain aspects of the world. The probability of a proposition being true is taken as a measure of how certain we are about it. Uncertainty can be due to different phenomena. The

world may be uncertain in itself. Then even knowing its governing principles, a noise term is involved in the exact description of the world state for given history; one cannot speak of a unique new physical state, instead there are a lot of states possible with a particular likelihood, e.g.

$$P(s_{t+1} | s_{0:t}, a_{0:t}) \sim \mathcal{N}(f(s_{0:t}, a_{0:t}); \sigma_s).$$

Most reinforcement learning models assume such indeterminism (Sutton and Barto, 1998), interestingly the work of Lang and Toussaint (2010) on learning in stochastic relational worlds even relies on stochasticity: compression to compact representation would not work without this assumption.

The world as perceived by an agent is mediated by sensors which deliver imperfect measurements. Exposed to this procedure, the view on the world gets tainted with more uncertainty. It can be quantified using probabilities in an observation model, how likely is to witness o_t in state s_t , e.g.

$$P(o_t | s_t) \sim \mathcal{N}(g(s_t); \sigma_o).$$

Again RL can be referred to for an example: augmenting a Markov decision process by lossy observation model yields a partially observable MDP (POMDP) (Cassandra et al., 1994). These formalisation models provide a good infrastructure to reason about nature and observation of phenomena: by assuming different distribution or transition functions, one can put different prior knowledge about the system. With these models the Bayes rule (cf. Section 2.1.5) can be used to transport information between observations and states and answer questions about them in various settings. For instance, the collected evidence can render some state or model more likely than other and thus strengthen the belief in a particular state/model (using the action corresponding to the MAP prediction).

The present thesis implicitly assumes the simplest case in terms of computation. The observations (contact points and normals) come from a Gaussian

distribution around the true state. The world is assumed static, apart from the actions taken by the robot. The motion prior is relevant to the trajectory optimisation later in Section 7.1: it is also assumed multivariate Gaussian in configuration space, i.e. the current joint angles are differently exposed to variation.

6.1.2 Under uncertainty

We know now that uncertainty is fooling around. How a model can be made less prone to uncertainty, how to get more from the data, how to get the right data, how does uncertainty help in this endeavour?

Vasudevan et al. (2011) develop a non-stationary neural network covariance and use dependent GPs to accommodate three multi-modal data sources in a single terrain model. They go beyond only varying the noise levels of data points from sensor to sensor. Instead, by calculating auto- and cross-covariances the fused GPs benefit from each other. In addition, the NN covariance turns out to be more suitable for this kind of fusion and the employed method manages to decrease the uncertainty in prediction (measured as the variance of the dependent GPs) as well as the prediction error. This is a demonstration of elaborate methods which profit from redundant data in terms of both prediction and certainty. Our GPISP representation uses different geometric arguments and a more basic GP formulation, argues in the same direction, though.

Regarding grasping, as argued in the beginning of this chapter, a straightforward approach to reduce the uncertainty is to not rely on a single grasping attempt but rather take into account multiple tries. Every encountered disagreement between the interesting features can be used to improve the estimate. By this schema Hsiao et al. (2010) take a POMDP approach for inferring an uncertain object pose by repeated execution of grasping primitives. On each execution they learn from unsuccessful grasps – due to collision or miss – and

update the belief for the next execution. The uncertainty is limited, though, to the 2D-position of a *known* object relative to the robot.

Another approach focuses on precisely *dealing* with uncertainty, rather than reducing it: this shall result in more robust grasping primitives. Christopoulos and Schrater (2009) study the reaction of human subjects to positional uncertainty induced by the environment. This inspires Stulp et al. (2011) to address the uncertainty in the position of a known object. They represent an integrated reach and grasp trajectory as a dynamic motion primitive (DMP). The robot grasps an object at a fixed position several times. By using an elaborate strategy for model free reinforcement learning with binary cost function (denoting grasp success) they optimise the parameters of the DMP to adapt to the situation. The optimisation starts with a trajectory and end point provided by a grasp planner. Then the object is moved to another position and again the DMP is updated in several tries. The positions are sampled from a very flat 2D Gaussian which resembles position change along a single direction on a table top. The DMP target remains at the mode of the Gaussian, while the object position varies during the learning phase. Repeating this procedure, the DMP should become more robust to the uncertain position. The authors report small but significant differences in the learned DMP depending on the orientation of the Gaussian, which suggests – like in the human subjects – adaptation to uncertainty. The robot learns to open its fingers wider, push (collect) possibly nearer objects further and only then close the fingers. Notable is the effect that it learns to exploit physical contact without having contact sensors – in our interpretation, again, a manifestation of the argument towards the inevitability of sensor and motor integration. The position uncertainty is limited to the flat Gaussian, i.e. to a single direction, which allows the robot to exploit this single strategy. For every other kind of noise model new primitives need to be learned. Furthermore, this approach deals with objects with known geometry (needed by the grasp planner to provide the initial DMP). In common with our approach to grasping, the authors view the reach and grasp

motion as a whole. In contrast, in summary, they aim to relax assumptions in grasping research which are orthogonal to the ones this thesis relaxes. For instance, as a plus, they deal with movable objects, and as downside do not address collisions.

Sen et al. (2011) represent objects as collections of probability distributions for geometric features. These features can be visually or otherwise perceivable, e.g. hue, position, point cloud, and are relevant to one or more affordances associated with the object. The actions which correspond to affordances are closed-loop sensory-motor controllers. Two types of actions – SEARCH and TRACK – work on the features and can be in one of the states: converged, non-converged and unavailable reference signal. TRACK preserves the registration of certain stimulus (feature, signal) over time, whereas the SEARCH tries to find one. Searching is realised as consulting obtained from experience spatially related distributions over the features. A distribution reflects the likelihood of finding the feature at a particular place and SEARCH can, for instance, move the robot as to draw such regions into the view field. In order to achieve a goal, the system first needs to put itself and the environment in a state in which the action corresponding to the goal can be executed. This involves planning of (i) actions which reduce the uncertainty of a model (e.g. about the object pose); these actions are proposed based on the *expected decrease of uncertainty*; (ii) actions which change, for instance, position and orientation of an object as to make the goal action executable (e.g. pull a hammer to make its handle reachable before grasping). The authors make a notable observation about uncertainty: even if the object pose is ambiguous, an action can be safely executed if it is agnostic to this type of ambiguity. In other words, uncertainty w.r.t. the task is what matters. Interesting property of the approach is that the representation can mediate between higher level symbolic planning and reasoning and the geometric world. Put in relation to this work, ours can be seen as providing a specific uncertainty-aware closed-loop grasp controller with the corresponding alternative features: we contribute the insight that

uncertainty is a geometric motion feature, relevant also beyond (below, to be precise) high level reasoning.

Not only in grasping context one can search for particular interactions which are more advantageous for specific goals. For instance, for discovering the kinematic structure of articulated objects Katz and Brock (2008) evaluate effects of the robot's own movements relative to the parts of an object and find the types of joints that connect rigid bodies. Essentially, this employs interaction to gain certainty about a kinematic chain; more importantly – even for known structure, many kinematic parameters can be acquired only by interpreting the effects of interaction with the chain (e.g. joint limits).

The effectiveness of human use of tactile feedback for grasping suggests that in biological entities the sensory and motor pathways are tightly connected. It provides inspiration to seek to integrate them in artificial systems expected to possess advanced motor skills. Various ways to do this include integration of tactile feedback in the control loop, visual servoing, grasp success assessment based on tactile events. This reduces the negative impact of uncertainty in the sense that the more reactive a controller, the less it allows deviations to grow unnoticed. For instance, the integration of tactile feedback and grasping experience is discussed by Steffen et al. (2007). Their policy reacts to tactile events during grasping and selects from an experience database the most appropriate target posture according to similarity function based on the current contact configuration. While the contact points are used by Steffen et al. to predict the grasp from experience, we use tactile feedback to refine the world model for the purpose of grasping.

6.1.3 Active learning

We encounter manipulation approaches which act as if they had perfect models – ignore stochasticity and uncertainty and let them go undisturbed into their success rates. Other approaches learn from experience, they take any

incoming data to improve their models which become more confident. Other class of approaches not only deal with uncertainty but base discrete decisions on, for instance, whether they are certain enough to perform an action. This section looks at active learning approaches. They reason about the informativeness of next queries and go and harvest data where it makes most sense. In the context of robot manipulation, this means to put actuation in favour of sensing. The specific questions arise from how to model the cost of moving and manipulation and the gain from the actively changed situation.

Different use cases can benefit from actively selected data. MacKay (1992) investigates this in Bayesian learning framework and identifies the following three cases in the context of regression: (i) For a fixed model class, find the data which is most informative about the parameters \mathbf{w} of the best model. (ii) In the same setting, when one is interested in the model predicting best only at a set of points, not everywhere. (iii) Find data which helps to best discriminate two competing models. As measure of information gain he uses the expected *change in entropy*, ΔS , between the distributions over the parameters before and after observing the new data, $P^N(\mathbf{w})$ and $P^{N+1}(\mathbf{w})$. He shows that an alternative to ΔS , the *cross entropy* (G), although different in interpretation, yields the same in expectation, i.e. $E(G) = E(\Delta S)$. This choice of measure leads to a strategy which favours *reliable* data at points of the model which currently have *large variance*. The maths of this result reveals that the value of the future observation does not go into the computation, which is why one can select the query point without knowing the answer. Instead of the *total information gain*, MacKay (1992) shows for the second case that the *mean marginal information gain* is the suitable measure. The intuition behind the information measure for the third case is that it favours queries at points where the predictions of the two models disagree and their confidence is different. Finally, MacKay (1992) warns about a weakness of information-theoretic approaches to data selection: they assume a correct model – we need to be aware of undesirable results due to violating this assumption in practice.

Another information theoretic way to measure uncertainty – the variance of the model – determines similar objective functions and policy for optimal data points selection. Cohn et al. (1995) introduce the integrated variance

$$IV = \int \sigma_y^2 P(x) dx, \quad (6.1)$$

with $\sigma_y^2 = \sigma_y^2(x)$ the predictive variance at the query point x ; $P(x)$ is the distribution in the input space. They derive σ_y^2 for three statistical models – neural network, mixture of Gaussians and locally weighted regression – and show that for the latter two it can be evaluated exactly and efficiently, in contrast to the NN case. A Monte Carlo approximation of the integral (6.1) is computed by evaluating at a number of points drawn from $P(x)$.

Another, in a sense orthogonal to MacKay’s, setting is global function optimisation. The aim is not only to predict correctly the outcome at a point or set of points, but to find the proper point in the first place – and one needs the model in order to reason about that. Srinivas et al. (2009) are concerned with how much loss one suffers while optimizing a function in the bandit setting. (A multi-armed bandit is a common model of an environment with unknown usually static reward distribution which is to be uncovered and exploited. *Regret* is the loss suffered by not taking the optimal policy.) They bound the *regret* for GP optimization in terms of information gain (on the introduced GP-UCB rule for sequential optimization). Upper confidence bound (UCB) is a data selection approach to trade certainty off for opportunity, exploitation off for exploration. Intuitively, it takes the point at which it could potentially gain the most, i.e. the sum of the MAP estimate and the error bar according to the current model is maximal. Points lose saliency if they do not confirm the high outcome one hoped for. Unvisited points which do not predict the highest outcome become worth a visit. Srinivas et al. notice that sequential exploration, and function optimisation have different goals, and exploration strategies do not translate directly to optimisation: concentrating at maxima happens too

slow. However, they note that the greedy strategy of taking the maximum-variance point is near-optimal for exploration – this is very much related to the control laws we design in this chapter.

An actively learning system evaluates current evidence to select a query point in the space it learns in. Using data in form of demonstrated or experienced grasps allows to select the next point in the space of grasps, the next query is a grasp attempt. In an application of active learning to the domain of robotic grasping Kroemer et al. (2010) define a hierarchical controller – a lower level for executing grasps, and higher for determining the next grasp to be executed. Their decision policy is to maximise the sum of 2 terms: (i) the expected immediate reward (*value*) of a grasp and (ii) the standard deviation at that point (cf. UCB above). This policy ensures convergence to a grasp only when no other grasp is more promising (under the assumption that finding global maxima is feasible).

A way to express the *value* of a grasp is the probability of success as being function over visual image features. Estimating it is again a matter of function optimisation or exploration. This is how the problem is casted by Montesano and Lopes (2012): image positions/features give rise to a feature space and cost landscape which are common across different images and objects. They model each grasp as a specific to a image position Bernoulli distributed experiment with certain parameter ρ , the success rate. The posterior over ρ after a number of such iid experiments is Beta-distributed with two parameters interpreted as $(\alpha, \beta) = (\text{successes}, \text{failures})$. This is how the authors estimate a grasp success distribution (grasp probability) p for a single image point. Assuming smoothness of the grasp probability, one can discount according to a Gaussian similarity the impact of the observations (number of successes and failures) and of the prior p and thus propagate the data to the adjacent areas. This allows the Bernoulli evidence of different points to contribute to each other's posterior. The result is a smooth estimate of the grasp probability for each image point. With the employed Beta distributions,

Montesano and Lopes look at both exploration of the whole feature space and giving preference to points with higher probability of success (which amounts again to function optimisation). The objective functions are *model variance* or *model entropy* for the exploration case, and *expected improvement* as function of probability and variance. A toy experiment shows clearly that in the latter case, the quality of approximation in the preferred regions is better on the cost of worse estimate in the rest of the feature space. The same example illustrates the difference between the variance and entropy optimisation: the entropy focuses on uncertain outcomes – around $p = \frac{1}{2}$ – while variance is equally interested along the whole spectrum. Notable feature of this approach is that by seeing the *value* of a grasp as probability distribution, it can learn from unsuccessful attempts, too.

Two approaches which decide on interaction based on some notion of uncertainty are discussed by Bierbaum and Rambow (2009) and Caselli et al. (1996). They both have in common with us the fixed object assumption, i.e. that interaction does not move the object. The latter work is focused on recognition of known polyhedral objects from tactile information. Since they assume convex objects, they can use two polyhedral representations – a lower and an upper bound. The lower bound polyhedron has the contact points on its vertices, and the upper bound polyhedron has them on the faces. The difference between upper and lower structure can be seen as measure of uncertainty and can provide hints which parts to explore.

Bierbaum and Rambow (2009) explore an object to acquire grasp hypotheses. They fill the volume of the cube containing the object with attracting points which gives rise to a potential field. They employ the same idea of particles attached to the fingers and describe a policy for creating and deleting potential sources and for changing attracting to repulsive potentials depending on registered contacts, thus leading the end effector to unexplored areas. The uncertainty is implicitly contained in the repulsive and attractive potentials in the volume and thus used by the particle motion.

6.1.4 Summary

Steffen et al. (2007) integrate sensory feedback and motor control but focus on the very last phase: closing the fingers from a pre-grasp, complying to the shape. Stulp et al. (2011) learn a grasp trajectory from experience, Hsiao et al. (2010) use multiple interactions to localise an object; both address positional uncertainty of previously known objects. Bierbaum and Rambow (2009) and Caselli et al. (1996) operate on polyhedral models. The latter are restricted to convex objects, represent uncertainty as upper and lower bound polyhedrons which enclose all models consistent with the contacts. The former focus on determining grasp affordances (parallel faces) and use point-potential sources for attracting to unknown (repulsing from known) regions. In contrast, our methods use a single representation for both sensing and grasping and thus links more directly what is relevant for motion and for estimation.

Active learning provides the information-theoretical foundations for autonomously and iteratively collecting valuable world observations. It is especially relevant to robotics due to the nontrivial costs of moving kinematic structures involved in harvesting a data point. It naturally relates these costs, i.e. control, to the data acquisition, i.e. sensing. It has been rarely explored in a geometric setting, though. Kroemer et al. (2010) and Montesano and Lopes (2012), essentially, actively learn grasp success densities over visual features. Both are bound to vision; the latter do not focus on control.

Sen et al. (2011) represent objects as probability distributions over perceivable features. They assign states to motion primitives which makes them discrete and more suitable to plan with. This has the potential to bridge the geometric and the symbolic world. They deal with uncertainty at the higher planning level, while we introduce uncertainty as motion primitive feature.

After reviewing related works which deal with uncertainty, the following sections present an approach which emerges from the GPISP representation.

6.2 Uncertainty-aware grasping

This chapter started with the question whether uncertainty awareness can improve the actions of a system. The current section uses GPISP representation to describe a principled way to convey the uncertainty from the sensors to the motor commands: it extends a controller by a feature which allows to prefer particular grasps – depending on the certainty of the model at the contact points.

6.2.1 From sensor uncertainty to model uncertainty

A coherent way to translate sensor uncertainty to model uncertainty is to transform the sensor imprecision to model variability. This precisely is what GPISP does: map distributions from sensor space to model space. On the one side, the variance of the observation noise distributions encodes the sensor reliability, on the other side, the resulting model is a Gaussian distribution over shapes. Its mean is the MAP estimate: $f \sim \mathcal{GP}(\bar{f}(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. In other words, GPISP approximates a function $f(\mathbf{x})$ whose 0-level set is the surface of the approximated object, $S = \{\mathbf{x} : f(\mathbf{x}) = 0\}$. For an arbitrary point \mathbf{x} the GP gives a posterior distribution conditioned on the observations: together with an estimate $\bar{f}(\mathbf{x})$ of $f(\mathbf{x})$ the model provides a quantification of the uncertainty about this estimate – the variance of the distribution, $\mathbb{V}[\bar{f}(\mathbf{x})]$. The variance is low if the query point is near observation points and increases in unseen regions. It is a function of the parameters of the GP covariance. As argued in Section 3.5, they would be typically set prior to estimation to reflect the embodiment, in particular the precision of the used sensor: the more trustworthy the data, the less variance at observation points and overall.

Now we show how the variance can be computed for the particular GP configuration we use. Following roughly the notation of Rasmussen and Wil-

liams, the variance is given by

$$\mathbb{V}[\bar{f}(\mathbf{x}_*)] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}^T \mathbf{G}^{-1} \mathbf{k}, \tag{6.2}$$

with query point \mathbf{x}_* , estimate $\bar{f}(\cdot)$ and GP covariance function $k(\cdot, \cdot)$. \mathbf{k} is short for the vector of similarities between $f(\mathbf{x}_*)$ and the observed values: $\mathbf{k}_i = k(\mathbf{x}_*, \mathbf{x}_i)$. \mathbf{G} is the Gram matrix containing the covariances between the observations. The used covariance is squared exponential:

$$k_G(a, b) = \sigma_p^2 \exp\left(-\frac{1}{2\sigma_w^2}(\mathbf{a} - \mathbf{b})^T(\mathbf{a} - \mathbf{b})\right). \tag{6.3}$$

In the presence of noise (uncertainty) the Gramian is actually augmented with the diagonal matrix Σ_n which has on its diagonal the variance of the Gaussian noise assumed for the particular observation: $\mathbf{G} = \mathbf{K} + \Sigma_n$. We have two types of observations: function value and function derivatives (gradient). Therefore, $cov(\mathbf{a}, \mathbf{b})$ is in fact different function depending on the arguments' indexes. For instance, if the i -th observation is a derivative, \mathbf{k}_i measures the similarity of $f(\mathbf{x}_*)$ to $\frac{\partial f(\mathbf{x}_i)}{\partial x_l}$ (x_l denotes the component in which the observed derivative is taken). This is not the same as the covariance between $f(\mathbf{x}_*)$ and observed value $f(\mathbf{x}_i)$. Nevertheless, following the result from Rasmus sen and Williams (2006, p.191), for given $k(\cdot, \cdot)$, all entries \mathbf{k}_i can be expressed in terms of $k(\cdot, \cdot)$. Please refer to Appendix A.2.5 for the proper equalities for this and the other mixed covariances between derivatives. With \mathbf{k} and \mathbf{G} all terms are established and one can analytically compute the model variance for the squared exponential covariance case. For example, it can be visualised: Figure 6.1 shows a 3D GPISP object model obtained as an estimation from several virtual contact points; this is the zero-level set of the MAP estimate, the MAP shape; the variance at the surface points is painted so that dark means high variance. This can be seen as the uncertainty map of the model. Note that the figure shows the map for the surface points only, but the variance is avail-

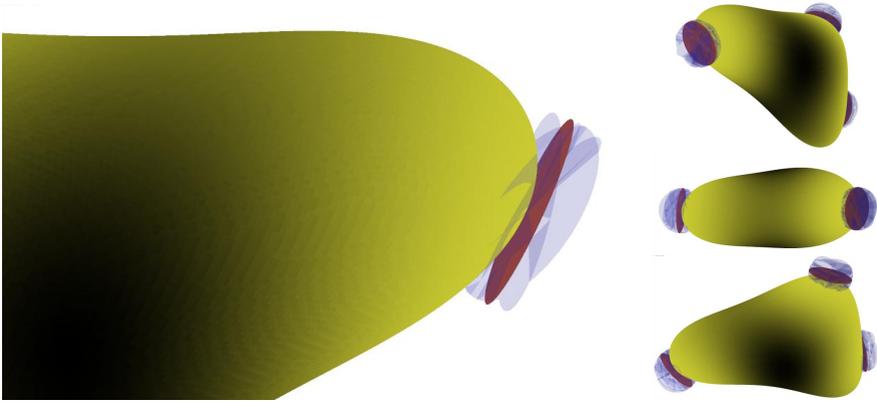


Figure 6.1: GPISF model of a 3D object with color encoding the variance on the surface (the darker the higher). The overlapping transparent discs are sampled from the observation noise distribution. Its mean is the opaque disc – the sensor reading. Consistently GPISF translates this noise into model uncertainty (in the visualisation: displacement into color).

able everywhere. The farther from the surface, the higher the variance – even around observed regions.

6.2.2 Uncertainty-aware grasp control

Now that can be measured how certain the model is in particular region, one can use this to improve the quality of grasps or grasp sequences. The controller proposed in Section 5.3.3 works on the mean of the shape distribution, i.e. on the maximum a posteriori estimate of the shape. It practically ignores one of the strengths of the GP – the error bars – which the present chapter focuses on.

6.2.2.1 Model variance as motion feature

Introducing variance to control enables a range of controllers which consistently define exploration-exploitation levels for grasping: the resulting uncer-

tainty-aware controller implicitly contains the control costs and they can be traded off for variance maximisation (or minimisation).

In GPISF context it is natural to realise such control laws by following the *gradient* of the variance. For a GP with squared exponential covariance we derive it analytically and can use it as a feature, i.e. $\mathbf{y} \in \mathbb{R}^d$, the vector of GP variances at d body points. For instance, consider the motion feature ‘how much do I (mis)trust the model at my fingertips.’ The value of this tv (s. Section 2.2.2) at each time step is simply the vector of variances $\mathbf{y} = \mathbb{V}_{1:d}$ at the changing positions of the fingertips.

Before we proceed, we clarify some terms and notation: \mathbf{q} denotes the vector of joint angles (point in configuration space); the current position of each finger is computed from forward kinematics: $\mathbf{x}_i = \phi_i(\mathbf{q})$. Its Jacobian is assumed known: $\frac{\partial \phi_i(\mathbf{q})}{\partial \mathbf{q}} =: \mathbf{J}_i(\mathbf{q})$. $f = f(\mathbf{x}_i)$ is the potential field value and \mathbb{V}_i is the variance at this point. Again, \mathbf{k} is the vector of covariances between particular point (here ϕ_i) and all GP observations (and $\dot{\mathbf{k}}$ the vector of their derivatives).

Back to \mathbf{y} , the Jacobian of a component of \mathbf{y} is

$$\frac{\partial \mathbf{y}_i}{\partial \mathbf{q}} = \frac{\partial \mathbb{V}_i}{\partial \mathbf{q}} = \frac{\partial \mathbb{V}_i}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \mathbf{q}} = \frac{\partial \mathbb{V}_i}{\partial \mathbf{x}} \frac{\partial \phi_i(\mathbf{q})}{\partial \mathbf{q}} = \frac{\partial \mathbb{V}_i}{\partial \mathbf{x}} \mathbf{J}_i(\mathbf{q}). \quad (6.4)$$

Now, to the end of analytically computing the value, it remains to find the derivative of the GP variance. By definition

$$\frac{\partial \mathbb{V}_i}{\partial \mathbf{x}} = \frac{\partial}{\partial \mathbf{x}} (k(\mathbf{x}_i, \mathbf{x}_i) - \mathbf{k}^T \mathbf{G}^{-1} \mathbf{k}). \quad (6.5)$$

Noting that $k(\cdot, \cdot)$ is stationary in this case and has 0 derivative, only \mathbf{k} varies

with \mathbf{x} . Applying this and the chain rule yields

$$\begin{aligned}\frac{\partial V_i}{\partial \mathbf{x}} &= -\frac{\partial \mathbf{k}^T \mathbf{G}^{-1} \mathbf{k}}{\partial \mathbf{x}} = -\frac{\partial \mathbf{k}^T \mathbf{G}^{-1} \mathbf{k}}{\partial \mathbf{k}} \frac{\partial \mathbf{k}}{\partial \mathbf{x}} \\ &= -2(\mathbf{k}^T \mathbf{G}^{-1}) \frac{\partial \mathbf{k}}{\partial \mathbf{x}} = -2(\mathbf{k}^T \mathbf{G}^{-1}) \dot{\mathbf{k}}.\end{aligned}$$

For squared exponential covariance \mathbf{k} and $\dot{\mathbf{k}}$ are available in closed form (cf. Appendix A.2.1, A.2.5), so finally, for the i -th of the d points

$$\frac{\partial \mathbf{y}_i}{\partial \mathbf{q}} = -2((\mathbf{k}^T \mathbf{G}^{-1}) \dot{\mathbf{k}}) \mathbf{J}_i. \quad (6.6)$$

Thereby all terms are known and the differentiable motion feature is complete. It essentially augments the task space so that one can conveniently express goals in terms of uncertainty: uncertainty control laws.

6.2.2.2 Control laws

The Jacobian of a task space feature is a control law, direct prescription, what has to be done on infinitesimal scale in order to maximise/minimise the feature, i.e. the inverse Jacobian gives locally the $\Delta \mathbf{q}$ which fulfills the task.

The variance gradient would likely point away from the surface. Thus, if a variance task which favours uncertain regions is combined with a manipulation task, their goals contradict. This can be accounted for by dynamic relative weightings or prioritising. Alternatively, one can split the gradient in two orthogonal components, $\frac{\partial V_i}{\partial \mathbf{x}} =: \dot{\mathbf{v}} = \dot{\mathbf{v}}_t + \dot{\mathbf{v}}_n$, one tangential and one normal to the surface. To get rid of the aforementioned trade-off (or better: resolve it explicitly), the Jacobian of \mathbf{y} can use $\dot{\mathbf{v}}_t$ rather than the proper variance gradient, i.e. the RHS of Equation 6.6 is projected on the surface tangent. In optimisation terms, this is a directional constraint on variance optimisation. Using $\dot{\mathbf{v}}_t$ makes the difference from optimising *weighted potential and variance TVs* to optimizing *potential TV and in its null space a variance TV*: the latter finds

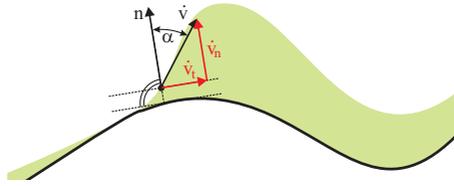


Figure 6.2: Gradient of the GPISF variance split into components normal and tangential to the surface. The solid curve is the implicit surface, $\{\mathbf{x} : f(\mathbf{x}) = 0\}$; variance along the surface is depicted as shaded area; $\dot{\mathbf{v}}$ is the variance gradient and $\dot{\mathbf{v}}_n, \dot{\mathbf{v}}_t$ its components.

the variance optimum on the surface rather than a trade-off optimum possibly far off surface. Figure 6.2 is a sketch of the variance gradient and its components at a low-variance point near surface. Knowing $\mathbf{n} = \nabla f(\mathbf{x})$ and $\dot{\mathbf{v}} = \nabla \nabla[f(\mathbf{x})]$, the directional component along the tangent is computed as $\dot{\mathbf{v}}_t = \dot{\mathbf{v}} - \dot{\mathbf{v}}_n$. The unit normal vector is $\frac{\mathbf{n}}{|\mathbf{n}|}$. From trigonometric arguments, $|\dot{\mathbf{v}}_n| = |\dot{\mathbf{v}}| \cos(\alpha)$. Together,

$$\dot{\mathbf{v}}_n = \frac{\mathbf{n}}{|\mathbf{n}|} |\dot{\mathbf{v}}| \cos(\alpha).$$

Plugging here the inner product $\langle \mathbf{n}, \dot{\mathbf{v}} \rangle = |\mathbf{n}| |\dot{\mathbf{v}}| \cos(\alpha)$ resolved for $|\dot{\mathbf{v}}| \cos(\alpha)$ yields

$$\dot{\mathbf{v}}_t = \dot{\mathbf{v}} - \left(\frac{\langle \mathbf{n}, \dot{\mathbf{v}} \rangle}{|\mathbf{n}|^2} \right) \mathbf{n}.$$

Now this can be substituted for the variance gradient in (6.6),

$$\frac{\partial \mathbf{y}_i}{\partial \mathbf{q}} = \left(\dot{\mathbf{v}} - \left(\frac{\langle \mathbf{n}, \dot{\mathbf{v}} \rangle}{|\mathbf{n}|^2} \right) \mathbf{n} \right) \mathbf{J}_i.$$

The result is a more precise control law which states *maximise variance, keeping constant distance to surface*. This formulation suits the grasping and manipulation context better since it does not interfere with the natural task to establish contact with the object.

As a demonstration of the control law, consider a GPISP model and a particle in its field. The particle has the mission to land at a low variance point on the surface. This translates to two objectives – approach the surface and search for low variance. They can be put in two task variables: the one defined here and the one from Section 5.3.4.1 which follows the GPISP gradient. This is demonstrated in the following section.

6.3 Demonstration: Variance aware particles

Figure 6.3 shows the trajectories of particles which are attracted from the surface and in particular from uncertain (left) and certain (right) areas. The particles start at randomly sampled positions around the object (green diamonds). Each particle's trajectory is made up of simple optimisation steps. A gradient descent optimises the cost function of the two TVs. Regardless of the different start positions the different particles eventually land at few common points on the surface (yellow diamonds) – according to what the target of the uncertainty TV is set to. The particles in this simple demonstration exhibit exactly the behaviour expected from the effectors of a robot manipulator – we seek to integrate into our grasp controller the tendency to establish contact at a low variance (or high variance) surface points. The following section shows how manipulation can benefit from these control laws and formally integrates them into a manipulation paradigm.

6.4 Exploration and exploitation

To explore means generally to actively look for data points in order to improve the world model. A moving camera looking at an object from different sides, aggregating the views to a complete object model is an example. However, this is still passive – it accepts that the information is static, i.e. does not alter the scene to induce additional data. More powerful scenarios loosen this

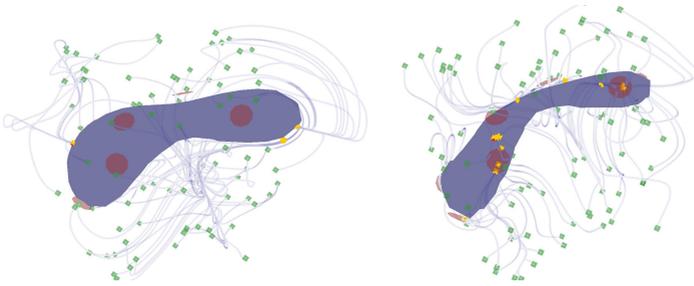


Figure 6.3: Particles following the gradient of a simple objective function which jointly minimises potential and maximises (left) or minimises (right) variance.

assumption: manipulation put in favour of sensing opens the possibility to actively change the scene, obtain data points which are originally unreachable. This can improve the models faster and obtain better predictions. Back to the camera example, an end effector may push an obstacle to the side to avoid occlusion and explore visually a previously unseen part. This example could lead to very powerful exploration strategies planned at a higher cognitive level. Here however, we are rather concerned with exploration and exploitation on a motion primitive level. Later we demonstrate an exploration scheme in the context of grasping with tactile sensors only. Then, to explore means to get in contact with more areas of a surface and create a better model from these observations, while for exploitation one takes an existing model of an object and does one's best to grasp it.

The uncertainty-aware control laws from Section 6.2.2 can be tuned to span the spectrum between pure exploration and pure exploitation and can be used as building blocks for explore-exploit grasp sequences. Examples include *explore-grasp*, where all fingers are set up to high variance; *exploit-grasp*: fingers tuned to minimize variance. One can think as well of grasps which lie in between these extremes: a control law that assigns explore task to one effector and exploit to other. For instance, if parts of an object are sufficiently known, it can be reliably touched with one or two fingers to fixate it (the exploit part).

An additional finger can be used to explore the opposite side. Similarly, as bi-manual manipulation one hand may hold an object and the other explore its unknown parts. Placing one effector at a known region and sending another to an uncertain region amounts technically to different target values for the two particles in the uncertainty TV.

With such primitives, grasping previously unknown objects can be viewed as interactive process, as active learning setting where each next interaction with an object is a trade-off between exploitation of the more certain regions of the estimate and exploration of unseen (or less certain) regions. In order to define a policy for such grasp series, one needs to decide how long and how much to trade off. Means to support such decisions are the locally and globally measured variance. Locally the variance is a measure of the certainty of the shape features. It answers the question how far one can trust the grasp points, i.e. how confident is the model that the candidate contact points for a grasp are indeed suitable. This sort of information drives the control laws from the previous sections. One can term them local, primitive, aware of uncertainty but unaware of the more general context. On the other hand, globally, the variance of the whole model or just along (what is believed to be) the surface can be integrated. This overall variance indicates how much better the model can get, i.e. tries to answer the question whether an explore step would substantially gain information. Such questions serve a higher level decision maker that is aware of the global context and tasks.

Consider for example the trivial policies *always explore* and *always exploit*. The former is a sequence of explore-grasps and is expected to effectively estimate an unknown object. The latter is a sequence of exploit-grasps and shall grasp the object robustly. The stopping criterion for pure exploitation arises rather naturally – go on exploiting until successful grasp. A stopping criterion for pure exploration, on the other hand, can be derived from the very same measure which is used as target: the trend of the accumulated variance along the belief surface. (Section 6.5 discusses these measures). If the decrease in

total variance flattens out as the grasp attempts progress, this could be a sign that the algorithm knows the surface good enough. (More precisely, it has already reached the most of confidence that it could reach.)

Note that an exploit sequence, even though unlikely, can get informationally stuck in repeating the same trajectory. This would be the case if the trajectory is not successful and the information gained from the interaction does not change the model in a way that some other grasp becomes more preferable. One way to make this less likely is to introduce some stochasticity, e.g. a bias towards a random approach direction.

6.5 Model quality measures

With the choice to express model uncertainty by the error bars of the GPISP probabilities aggregated variance naturally emerges as a tool for assessment of model quality, for comparing models and for supporting educated decisions in higher level explore-exploit policies.

The total variance expresses how certain the model is everywhere. One can numerically integrate it by dividing the space in voxels and evaluate the variance once per voxel:

$$\omega_T = \frac{1}{Z} \int \mathbb{V}[v] dv. \quad (6.7)$$

The factor Z makes different models comparable and is related to the number and volume of voxels evaluated. However the quantity ω_T is not specific to any objective which comes to our mind – it is too general to be informative.

What is a GPISP model interesting for? Since it is a shape-focused object model, the surface is of most interest, and the surface is a small subset of the whole distribution. Two flavours of aggregating the uncertainty on a surface follow.

The variance, measured on the surface of the true object is

$$\omega = \frac{1}{|S|} \int_S \mathbb{V}[v] dv. \quad (6.8)$$

It numerically integrates the model variance along the true surface, S , normalised by the volume of the voxels visited. The result is the mean variance encountered along the ground truth zero. This allows to compare in terms of their reliability models which have similar MAP predictions. One can distinguish a trustworthy model which evolves due to comprehensive exploration and has low variance on the true surface from one which turns out to make correct predictions merely by chance. In conjunction with a later experiment, Section 7.1.4 contains examples of how this quantity evolves with acquired new evidence. Note that both ω and ω_T monotonously decrease with new information arriving.

The model variance integrated along the estimated – rather than the true – surface is a measure which does not require knowledge of the ground truth. With estimated surface \hat{S} substituted for S 6.8 yields

$$\omega' = \frac{1}{|\hat{S}|} \int_{\hat{S}} \mathbb{V}[v] dv. \quad (6.9)$$

Unlike ω this measure is accessible to the system and can be used introspectively to decide if an estimate is good enough. Just like \hat{S} is an estimate of S , ω' is at most an estimate for ω , at most as good as the current belief resembles the true surface. While S is stationary, \hat{S} grows and shrinks according to the encountered observations. Although evidence always reduces the total variance (in the whole space or for fixed subset), summing along a changing shape estimate may distort the average. Concretely, ω' is not necessarily monotonous and can not be a consistent approximate of ω . Still, if monitored over time, it can support decisions: e.g. if after an initial decrease ω' appears constant for some time, this might hint that the aggregated variance arrived at its local

minimum and cannot gain much more accuracy from further exploration of the surface.

These measures are very similar to the integrated variance by Cohn et al. (1995). In terms of Equation 6.1, the variance is $\mathbb{V}[v] = \sigma_y^2$. The $P(\mathbf{x})$ used here is an uniform distribution and its support is the set $\{x : \hat{f}(\mathbf{x}) = 0\}$, i.e. $P(\mathbf{x}) = 0$ for $\mathbf{x} \notin S$ (resp. \hat{S}). Consequently, we numerically approximate it, computing the variance at all voxels on the surface (for an appropriate voxel resolution). This can be done efficiently by starting at an observed point, descending to the surface (if it is not there anyway) and going along neighbouring surface voxels until none remains unvisited. Repeating the procedure from all observed points, it will mostly visit even not connected surfaces (this is essentially the same traversing procedure Section 4.6 describes for visualisation). Similarly to the statistical models of Cohn et al., the variance of the Gaussian process is evaluated exactly and efficiently. Unlike their mc approximation of the integral, we compute the sum over a discretized manifold. The intuition behind $P(\mathbf{x})$ is to hint where to look at, based on what points are probable, i.e. it is the metric of the input space in general. In contrast, we set it ourselves to S , knowing what is the important part of the input space in our domain.

(On a related note, the variance integration is reminiscent of what Seitz et al. (2006) term ‘scene space photo consistency measures’ in the context of stereo vision reconstruction algorithms.)

6.6 On the notions of attempt optimality

Information theoretical arguments alone are helpful when candidate points differ only in terms of information content. However, queries may have overall goals beyond information acquisition and may be not equally convenient to obtain. The next sections discuss different notions of optimality which arise in this context, distinguishing *value*, *utility* and *worth*.

6.6.1 Constrained exploration

In the context of function optimisation, on the way to bounding regret, Srinivas et al. (2009) point out that simply greedily picking the data point with largest variance at each step is *guaranteed* to find near-optimal solution for function exploration. Information gain maximization involves computing the change in the volume of the variance of the *whole* model. The above remark is central because the local strategy is much cheaper but still comes with approximation guarantee! Looking closer, this provides an alternative view on our variance TVs: the explore-grasp controller tries to mimic exactly this greedy criterion by navigating the effector to the maximal variance area. It is never exactly the same, since the kinematic constraints imply a rivaling cost term. Thus, *a sequence of pure explore-grasps is an approximation of a near-optimal information-theoretical policy for object exploration*, each primitive trading the movement costs off for gain in information. The usefulness of a grasp (trajectory, configuration, contacts) to the end of purely exploring an object must respect kinematic and energy constraints. Let this mix of costs be the *value* of a grasp.

6.6.2 The grasping bandit: maximising *utility*

The first paragraphs of this chapter state that one would intuitively try to accomplish a manipulation tasks over and over again until they eventually succeed or give up. The whole process matters, not single attempt alone. The last grasp is the most important, but it is only defined after it has happened. This is very similar in abstract sense to searching for the optimum of an unknown function: the input space is the howsoever represented grasp and the response is its quality. Function maximisation is often considered not as goal in itself but as part of a setting where the accumulated gain of evaluating the function is the goal to be maximised. The winning algorithm walks rapidly to the global optimum and then exploits it. This constrains the resources spent in explor-

ing the function by introducing an explore-exploit trade-off. In conjunction to this problem the notion of *regret* of a query expresses opportunity costs: how much one loses from not being optimal. In these terms, the regret of one of a sequence of grasps can be seen as the distance (in the grasps' space) to the manifold of valid grasps. Grasping necessarily involves a controller which operates on an object model. Then a way to factorize the regret would be to ask how good is a grasp w.r.t. the model and how good is the model w.r.t. the true object. Each non-terminal failing attempt provides additional observations which improve the model. On its part – assuming that the controller is correct (works better with better models) – this improves the subsequent grasps. To get more concrete and more formal, view the control as a primitive which maps a model (GPISP conditioned on evidence) and start configuration to an end configuration (contact point or a bunch of contact points or other interaction or feature), $\kappa : (\mathbf{x}, \mathcal{GP}) \rightarrow \mathbf{x}'$; and let the *utility* be a mapping from such end state to a real number, $v : \mathbf{x}' \rightarrow r \in \mathbb{R}$. Then, the regret of the i -th attempt depends through the *utility* on all the involved entities – the model, the initial state, the control:

$$\rho_i = \sum_1^n \max_{\mathbf{x}} v_i(\mathbf{x}) - v_i(\mathbf{x}'_i).$$

The *utility* contains not only information-theoretical arguments for acquiring better model: it has a higher goal, the model should be just as good as to lead to the maximum. Note that at execution time, the system does not have access to the *utility* v . Kroemer et al. (2010) as well as Montesano and Lopes (2012) explicitly learn an approximation from experience, while we learn approximation of the object model. Our *utility* is implicitly contained in the tvs (motion features) which try to determine what is a good grasp: they evaluate the *utility* on the current belief. (One could attempt an analogy to model-based and model-free RL: our model is partly assumed known – the controller – and the rest is learned; in contrast, the other two approaches learn to map

the features directly to the *utility*.) This difference explains why Kroemer et al. can use explicitly a rule like GP-UCB to trade exploration off for exploitation in the space of grasps in their higher level controller. Our controller is flat – the exploitation-exploration is built in the primitive, i.e. it is a local/greedy explore or exploit.

6.6.3 The *worth* of an explore-grasp

While the *value* of a configuration quantifies how much information a robot gains w.r.t. general model improvement and the *utility* is focused at arriving faster at a successful grasp, another measure can be defined to capture how well the model is improved w.r.t. to certain task: the *worth*. In the bandit setting, *utility* is related to optimisation against maximising accumulated reward, i.e. running fast to salient points in input space and exploiting them. With *worth*, we do not regret. What matters is the final exploit, the non-terminal attempts need to merely harvest relevant information.

Learning actively, the agent needs to account for the costs of the query and as argued already they are implicitly contained in the grasp criterion and thus in the *value* and *utility*. A GPISP model is explored by explore-grasps, not by single touch. The *explore*- in explore-grasp gets information, while the *-grasp* ensures the information is relevant for grasping. The *worth* measures how relevant the query (and *querying*) is for the final task. Taking the controller objective function apart reveals that the oppose TV tries to make sure that with a single grasp attempt the robot acquires data points which are on opposite sides of the object; the kinematic constraints ensure furthermore that the acquired points are reachable in a feasible grasp. This is the benefit of tuning a grasp controller to explore: it is by design biased to maximise the *worth* w.r.t. its original task: grasping.

In Section 6.4 we touch on the different flavours of uncertainty-aware grasps, explore-grasp and exploit-grasp being most obvious. Having multiple

fingers allows to even mix targets in the same primitive: one or two finger exploit-grasp at confident areas while another finger explores an unknown region. Analogously in bi-manual manipulation. These mixed flavours can be seen as manifestation of maximizing *worth* w.r.t. grasping (having two good points, explore for third one so that eventually have a good grasp), rather than acting under pure information gain objective or under pressure to exploit.

6.6.4 Information, learning, interpretation

Active learning usually focuses on finding the point which would mostly impact the current model. The change of the integrated total variance (measure of uncertainty) of the model can be taken as a measure for impact (as do Cohn et al. (1995)). The found point is information-theoretically optimal – there is no point that yields more information. Similar to the way *utility* and *worth* accommodate arguments which counteract optimal information gain, one can reason about properties of consecutive model beliefs relative to each other, properties which depart from the information theoretic optimality. Costs of moving and other functional properties constrain the manifold in which to search for information. The costs of the movement suggested by the controller, however, depend on the model. Thus, it is worth considering how the model changes between consecutive interactions. For instance, single informative evidence may bring strong perturbations in the model \mathcal{GP} to become \mathcal{GP}^1 , where the total variance of the latter decreases: $\mathbb{V}^1 < \mathbb{V}$. As opposed to that, multiple less informative attempts could gradually lead to the same result: $\mathcal{GP} \rightarrow \mathcal{GP}^i \rightarrow \mathcal{GP}^{ii}$ with $\mathbb{V}^{ii} = \mathbb{V}^1$. While the information measure is not at its optimum for each attempt, indirect costs might turn out to improve. Imagine that the new model \mathcal{GP}^1 is not well interpretable from the position of \mathcal{GP} , while the links in the alternative chain of models might be easier to deal with. To be more concrete: assume we are in the GPISP world, have an observation and the finger is on the surface of the current belief (a blob). For

the purpose of exploring an object, sliding a finger along an unknown surface contrasted with re-grasping can give intuition for the effect. Sliding surely does not maximise the information gain – new observations are near to old ones and deliver redundant information –, but the estimate grows gradually and predictably. As a result, this strategy might need less assumptions and find the shape efficiently. At higher moving cost, one can gain more information per grasp attempt by re-grasping. But the topology of the estimate would likely change a lot in the beginning: from one blob after the first observation, there might be two blobs after the second, three after the third, and only then connect to a single one. The change in topology might have undesired effects, e.g. confuse the controller. One way to phrase this notion is as trading impact off for the certainty/predictability of the chosen point. A way to formalise it might be (in the above case) to penalise non-smoothness in consecutive models. For instance, in the GPISP world, an objective which favours sliding a finger might be the minimization of

$$v = \int \mathbb{V}_{i+1} + \int_{S_i} f(\mathbf{x})_{i+1}^2 d\mathbf{x}, \quad (6.10)$$

for the current shape S_i , possible estimate of the GP potential in the next step $f(\mathbf{x})_{i+1}$. The first term corresponds to the integrated variance along the model, the second term tries to bound the deviation from the current model. These considerations are closely related to what Bengio et al. (2009) term Curriculum learning. In their seminal work, they are concerned with describing what ‘simple’ means in the context of different data sets. In particular, they assume all the data points are available previous to learning, as well as a ‘teacher’ who provides the curriculum. Here we are simultaneously in an active learning setting. Selection criteria like the above bridge between the two concepts and give rise to an *active curriculum learning*.

How are *utility*, *value* and *worth* related? In summary, *value* is related

mainly to exploration scenarios and quantifies the information gain; by *utility* we denote how good a configuration (state) is in terms of finding quickly a valid grasp – this quantity is interesting when exploitation is involved; *worth* allows to taint the pure information gain with other objectives relevant to the last attempt only. They correspond to the following three scenarios. (i) Produce as precise model as possible. The goal is purely information-theoretical. (ii) For an unknown object, find a way to grasp it as soon as possible. The goal is to obtain as little information as needed to somehow grasp it (model quality is irrelevant beyond this). (iii) For an unknown object, find a good grasp. Intermediate steps are free to improve the model w.r.t. the final task. All three notions result from the choice of specific mix of objective functions. For instance, the effect of the second term in (6.10) could be achieved by higher weights on the movement costs. However, this carries different semantic. One does not necessarily need a name for each mix of objectives, but it is convenient to be able to separate the different mixes according to what they relate to.

6.7 Summary

Transforming sensor (im)precision into variance of the GP posterior distribution GPISP is capable of conveying sensor uncertainty to model uncertainty. Building on this feature, in the present chapter we contribute a control law which offers a way to give preference to regions of the model with particular certainty level. In turn, by adding this to a grasp controller we introduce the notion of explore-grasp and exploit-grasp primitives – the former tending to grasp at unknown regions, the latter to grasp at better known points. We illustrate the effects of such controller on idealised particles. The different flavours of the controller can be used in higher level policies which properly trade exploration off for exploitation. To enable an educated trade-off, we provide GPISP specific measures of global confidence.

We interpret the uncertainty-aware control objectives in established active learning terms and draw connections between grasping unknown objects and Bayesian function maximisation.

Section 6.6 suggests settings in which the information gain discounted by the cost to obtain data point is not the whole story. Additional trade-off against other application specific constraints may provide for improved learning and control.

We do not make assumptions about object shape apart from a general smoothness prior. This includes uncertainty across a wide range of unknown objects, in contrast to only positional uncertainty of known shapes or only convex surfaces handled by some previous approaches. However, in our current methods we assume fixed objects: We do not have model of how interaction moves the object, nor a prior how objects could move independently.

Chapter 7

Integrated demonstration systems

The previous chapters elaborate ways to process sensor information and obtain a geometric model of an object equipped with a local measure for uncertainty; to control a robotic arm to grasp this object; and to benefit from the uncertainty measure. The present chapter finally looks into the integration of the individual contributions to give them the meaning we motivated this work with: an integrated approach to manipulation which takes into account both sensor and motor aspects of a system and let them profit by each other.

The blind touching scenario described in the introduction amounts in terms of exploration-exploitation as discussed in Section 6.4 to a pure explore-grasp policy for iterative reactive grasping. In the following we describe an implementation of this policy integrated in a general framework for whole body grasp motion. In fact, this chapter contributes two flavours of the policy – as a trajectory planning for a Schunk arm and hand; and as a feedback controller for a Barret WAM. Finally, we demonstrate the power of uncertainty-aware object estimation with

- simulated arm: we conduct an experiment with a simulated arm-hand

system to illustrate and quantitatively evaluate the benefits of uncertainty-aware control;

- a human-controlled arm in simulation shows that concavity is not an issue;
- kinesthetically controlled arm demonstrating the feasibility of sensing in real world;
- autonomously exploring arm in simulation showing an elaborate feedback setting.

7.1 Explorative grasp sequence with trajectory planning

For the evaluation of the uncertainty control laws developed in the previous chapter we look at the task of estimating an object from tactile feedback. The robot repeatedly grasps at unknown areas guided by the information-theoretic arguments behind the control law: this is a sequence of explore-grasps. Here each grasp trajectory is planned previous to execution.

7.1.1 Grasping sequence

The reactive grasping follows the schema outlined in Algorithm 2. For each grasp a kinematic motion trajectory is specified in configuration space. For given object model (\mathcal{GP}), robot kinematic chain and environment, a sequence of joint angles $\mathbf{Q} = \{\mathbf{q}_i\}_{0:N}$ needs to be found which starts in a known home position $\mathbf{q}_0 = \mathbf{0}$ and ends in a feasible grasp configuration while avoiding collisions with other objects and staying within the allowed joint limits. This postulates an optimisation problem which is solved by *plan()*. Its definition is subject of the next Section 7.1.2. The routine *play()* follows the trajectory produced by *plan()* in a feedback loop until it registers a *contact*. A contact provides a tactile observation which is used to update the object model. If

Algorithm 2 Reactive grasping

```

 $\mathcal{GP}$ : add initial random observation
loop
   $\mathbf{Q} = \text{plan}(\mathcal{GP})$ 
   $(\hat{\mathbf{Q}}, \text{contacts}) = \text{play}(\mathcal{GP}, \text{world})$ 
  if not contacts then
     $(\hat{\mathbf{Q}}, \text{contacts}) = \text{close}()$ 
  end if
   $\mathcal{GP} = \text{update}(\text{contacts})$ 
   $\text{backw}(\hat{\mathbf{Q}})$ 
end loop

```

no contact is registered until the end of the trajectory, the fingers begin to $\text{close}()$ until contact with the object or self collision. (Can happen if the plan or the model turn out to be bad.) The real executed trajectory is remembered ($\hat{\mathbf{Q}}$). Finally, the arm plays the trajectory backwards and starts over with an updated model.

7.1.2 Trajectory optimisation

The forward trajectory is a result of an optimisation problem which is constructed from several tvs implementing feasibility criteria and heuristics for hand and finger orientation to lead to a good grasp. The tvs introduce costs which arise from the difference between desired and present value of motion features. For a given configuration the task cost is given by $c_t(\mathbf{q}) = \sum_1^n \rho_i (y_i^* - y_i(\mathbf{q}))^2$. For each step of a discretised trajectory, for each task variable \mathbf{y}_i there is a target y_i^* and precision ρ_i . The sum over all time steps constitutes the total task cost of the trajectory: $C_t = \sum_1^T c(\mathbf{q}_t)$. The cost function combines the GPISP grasp criterion tvs, and the tvs for collision and joint limits avoidance. In addition, the movement costs which reflect the flexibility of the kinematic chain are given by $c_g(\mathbf{q}) = \mathbf{q}^T W \mathbf{q}$ for a metric W . The GPISP tvs are responsible for the final grasp, while the intermediate trajec-

ory is governed by energy efficiency (movement costs). This briefly restates the trajectory optimisation problem we developed a scheme for in Section 5.5. Here the heuristics from Section 5.5.3 find an application to simplify the problem.

7.1.3 Tasks

The minimisation of the cost function leads to simultaneous satisfaction of the tvs according to the desired precision. The tvs from Section 5.3.4 account for collision avoidance, joint limit avoidance, wrist orientation, finger positioning and orientation. Collision tvs work on proximities between meshes or point clouds; joints have hard-coded hardware specific upper and lower limits and a dedicated tv penalizes going near the limits directly in configuration space. Position and orientation tvs are object centric – they are defined in terms of the GPISP model, rather than in Euclidean space, i.e. $y_i = y_i(f, \nabla f, \dots)$. Here an additional task variable is included which implements the uncertainty-awareness. It is defined as in Section 6.2.2.1 for the variance on the three finger tips: $y_V = (V_{1:3})$. This is the place to implement a variant of the explore-grasp and exploit-grasp: if the rest of the controller is left as is and the variance is conditioned $y_V^* = \mathbf{0}$, it tends to grasp well known regions; for $y_V^* = V_{max}$ it tends to explore new regions. (Note that V_{max} is the initial variance, the variance in the absence of data, $V_{max} = k(\mathbf{0}, \mathbf{0})$. With new experience new observations deliver more and more information and the variance can only decrease. And this holds for every \mathbf{x} .) In the experiments in the next sections the focus is on the exploration behaviour, therefore they use the latter setting for the controller.

7.1.4 An experiment in simulation

The simulated robot is a Schunk arm with Schunk dexterous hand. The arm has 7 serial DoF; the hand is three-fingered, with 2 DoF per finger, rotating

wrist, rotating ‘thumb’ and negatively coupled DoF for rotating the other two fingers, summing to 7 DoF, too. The arm configuration is as in Figure 5.7: it resembles the DoF of a human arm; it is mounted on a fixed base. The real hand is equipped with two sensor arrays per finger, at the tips and phalanges. This sensor feedback is not modeled in the simulation, it is assumed that the whole surface of the robot can detect contacts. There are no other sensor modalities available, in particular no visual information: thus, the initial guess about the position of the object is externally provided to the robot as a fictive tactile contact.

The simulator (Toussaint, 2010b) works with rigid bodies each comprising one or more shapes. The robot parts are modelled as triangle meshes. We do not use full physics simulation due to the stiffness of the position-controlled real arm which makes gravity and dynamics impact on the control negligible; and due to the fact that the target object is assumed fixed. Body proximities are computed among the convex shapes of bodies and to dense point clouds (the latter for the possibly concave intermediate GPIS beliefs). The proximities are used to simulate sensor feedback as well as collision detection.

Trajectories are planned prior to each execution of a grasp and then played in the simulator. The planning employs the motion features described in the above subsections together with movement costs compiled to an objective function as in Section 2.2.2. Using AICO (Toussaint, 2009) we factor the optimisation problem and infer the final trajectory of joint configurations.

Within this setup and using the reactive grasping loop described in Section 7.1 we conduct experiments to illustrate the effects of uncertainty-aware grasping. To this end, we place an invisible for the robot object within its work area, as in Figure 7.1. The position is uniformly randomly sampled in two dimensions, as is the appearance of the object among three predefined – lengthy, small and big one. Different positions on the table imply different kinematic constraints and ensure we test large region of the configuration space. The three different geometries account for the following cases:

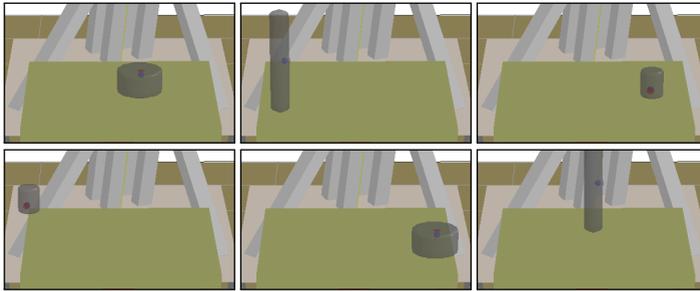


Figure 7.1: Example environment samples used in the experiment: uniformly distributed position and object type. A red dot denotes the initial random observation from the surface.

- Small objects which mostly fit in the hand of the robot.
- Objects which are the same size as the above in two dimensions, but are much longer in the third dimension. They need to be explored along this principal dimension. Possibly a single approach direction would do the job.
- Larger objects which hence need to be explored from multiple approach directions.

We expect the latter two to emphasize the merits of our method, since efficient exploration would demand to carefully select the grasps according to the current uncertainty information.

In the experimental setting we assume a fixed object in reaching distance and provide an initial, vague guess about the object – this is realized as a random observation from the surface given to the GPISF in the beginning. (In the visualisation it appears as a small sphere at that point. The sphere results from the GP prior.) The controller tries to grasp relying only on the available model. In its early attempts it is deemed to fail due to large discrepancy to the true object shape. On its trajectory the robot arm collides with the real object. This sensory event provides an observation of the function value ($f(\mathbf{x}_i) = 0$)

and the gradient ($\frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{x}} = \mathbf{n}_s$) for the GPISP and the model gets incrementally updated. After the collision the robot arm goes back and starts a new attempt to grasp the updated model.

A critical part of the repetitive grasping procedure is the contact which delivers the information for the model update. We assume that it can be detected at any point of the robot body, which is feasible in simulation but needs to be taken care of with existing real-world robots. Current hardware features tactile arrays sensors on very limited area of the body – mostly at the fingertips. More widespread are force sensors which would allow to detect a contact *somewhere* at the kinematic subtree below the sensor, but do not provide the exact place. For instance, the strain gauges found in the Barret WAM hand similarly allow to detect finger contact (the kinematic subtree made up of fingertip and link), but not the position of the contact on the finger. This leads to very unreliable approximation of contact point and normal, hardly useful for estimation. Nevertheless, a combination of the above sensors can be used to at least *detect* a contact which is not at the tactile array (or otherwise not precisely localised). This can enable proper reaction so that the procedure is safe, secure and does not stuck, for instance break the sequence, start an attempt from a different position, etc.

The experimental results compare two controllers which employ very similar sets of TVs for trajectory optimisation; the only difference is that naïve-control lacks the variance TV. For a random sample of the environment, i.e. the same object at same position with same initial observation, each controller is started once. Each run makes up to 20 successive grasps, incrementally updating its belief about the real object. Figure 7.2 shows a sequence of explore-grasps based on an updating GPISP estimate of a real object. The grey cylinder is the true object which is unknown to the system. The small blue sphere in the left most picture is the initial belief, resulting from the provided initial observation. The GPISP belief is shown as blue shape; this is the zero-level set of the MAP estimate of a GP conditioned on the observed points and normals. By

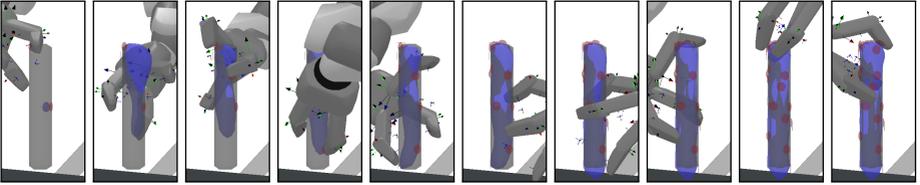


Figure 7.2: An explorative grasp sequence. Transparent grey: real object, unknown to the robot; blue: belief; red: contact points. Estimate after 2,4,5,6,8,11,13,15,16,19,20 grasps.

explore-grasping the belief, the robot receives new information which leads to better approximation of the real object.

7.1.4.1 Measurements

The comparison of the controllers records on each grasp two statistics related to how well the model agree with reality:

- Approximation error, or how far the estimate deviates from the real surface. We define similarity as the common volume of true object and belief object (v_c) penalized by the overestimated volume (v_o), all normalized to the total volume of the real object v_t . This number, the second term in (7.1), is at most 1 but due to the overestimation term, it can drop below 0. Subtracting it from 1 gives the approximation error:

$$\epsilon = 1 - \frac{v_c - v_o}{v_t}. \quad (7.1)$$

This quantity cannot drop below 0, which means perfect match, and can get arbitrarily high as result of gross overestimation. It is designed to capture the impact of the controller, rather than the common understanding of the term ‘similarity’. Thus, it would rate poorly an exact copy of an object with a systematic error in one dimension, i.e. the measure is not rotationally or translationally invariant. Furthermore, it

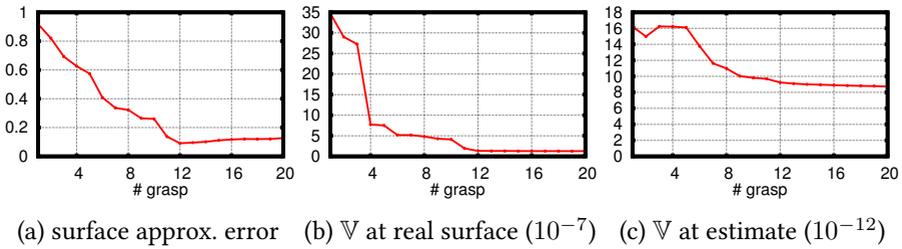


Figure 7.3: Statistics collected over single grasp sequence of 20 grasp attempts (cf. Figure. 7.2).

does not account for visual similarity, for instance, in terms of ϵ , a solid cube would be quite poor estimation of a closed box with thin walls, since most of the inner volume would be overestimation. While the rotational/translational invariance is welcome, one need to have in mind the solid vs. hollow property when evaluating a geometric controller. Figure 7.3a shows the approximation error development for a single sequence of explore-grasps.

- Total variance, or how uncertain the model is, measured on the surface of the true object. This is the quantity defined in Equation (6.8), an averaged numerical integration of the model variance along the true surface, S . An example plot of these measurements is Figure 7.3b.

The model variance along the estimated surface, ω' defined in Equation (6.9), is an accessible to the system approximation of ω . Figure 7.3c plots the measurement for the grasp sequence from Figure 7.2. Roughly at the same time as ω flattens, so does ω' . In this particular case, the slope of ω' turns out to be a good hint whether more exploration delivers more information.

Figure 7.4a and 7.4b show the evolution of these statistics with each next grasp, averaged over multiple runs. However, they mask the identity of the two runs which correspond to a single environment sample. To make this

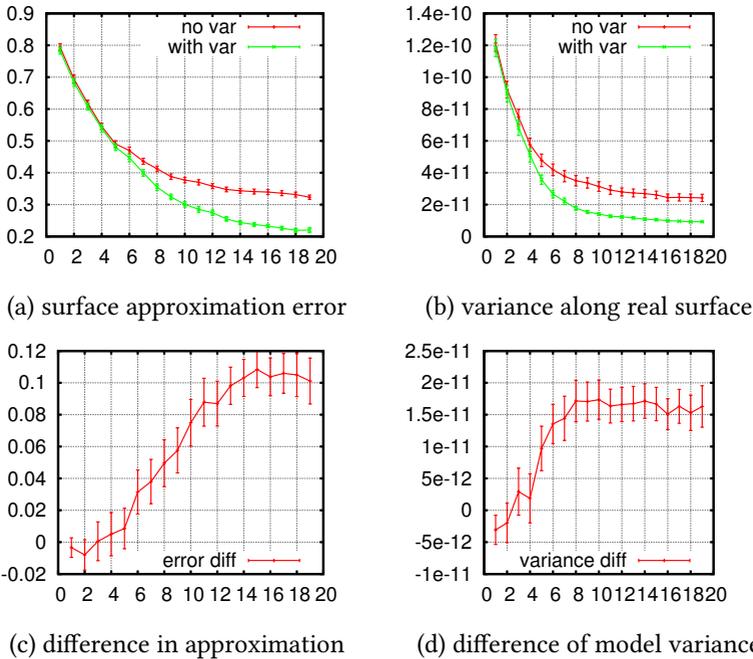


Figure 7.4: Statistics collected over about 2x hundred runs (one run *with* and one *without* notion of uncertainty), each performing 20 attempts to grasp with incrementally updating model. Error bars show plus/minus one standard deviation of the estimator.

correspondence more explicit, in 7.4c and 7.4d we plot the average of the difference of approximation quality with and without notion of uncertainty and, respectively, the difference of model variance along the true surface.

7.1.4.2 Results

The measurements give preference to the approach aware of model uncertainty. The estimation error is nearly identical for the first 4-5 grasps, but the neutral approach flattens out earlier. The added bias to grasp unknown regions leads to better estimates. In addition, it helps to reach same quality

estimates much earlier: already after 10 grasps the augmented controller performs better than the other after 20. The setup includes a table in front of the arm and the objects are placed above it. The collision avoidance task repels the hand from the table and makes it harder to explore the objects from below. This leads to either over- or underestimation in the lower part, which explains why the estimation error does not drop below 0.2.

The average uncertainty on the real surface, ω , also supports the suggestion that the use of model variance in the control law observes the object in more informed way. After 6–7 grasps the one controller is as confident as the other after 20, i.e. even with comparable estimation error the variance control law has gained more valuable observation points.

7.2 Estimation of non-convex object with GPISP

The results from the experiment in Section 7.1.4 quantitatively demonstrate the benefit of uncertainty awareness. A qualitative feature is about to slip away unnoticed. Most approaches to grasping are designed for convex objects. There are various ways to make them play relatively well with non-convex objects, for instance one may take the convex hull of the object and be prepared to accept poorer success rate. More educatedly, one can decompose an object in convex parts and grasp a single convex part under various constraints, again taking more complexity and poorer performance into account. In contrast, GPISP does not put restrictions on convexity. The very nature of the potential field, of the control laws defined so far, and of the sensing algorithm, allow to work with arbitrary objects. The previous experiment did not accent on this aspect.

Figure 7.5 shows a humanoid robot grasping a T-shaped object. The pictures are taken from a simulation environment with physical simulation of rigid body interactions. The humanoid robot is modeled with primitive shapes (capped cylinders) connected by joints. Its control loop is governed by an ob-

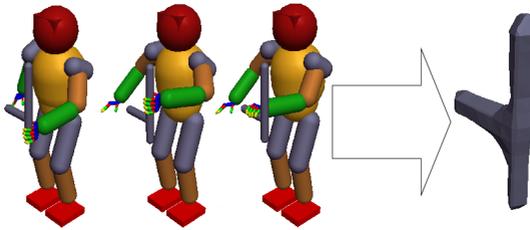


Figure 7.5: Belief about T-shaped object after reading out the contacts from 3 scripted grasps.

jective function which resolves redundancy with a null-space approach a la Liegeois. Tasks include among others balancing and maintaining a 2D position of the body (notice the right arm counter-balancing the movements of the left hand). The tasks relevant to this demonstration are the 6D position and orientation of the left hand and a grasp synergy coupling the fingers to a single DoF (open/close all fingers together). These tasks can be controlled by the user to place the hand near an object and grasp. The object is made from two long capped cylinders with a radius suitable for the robot hand. The closing fingers contact the object and the detected contacts deliver simulated tactile observations. They feed a GPISP model which approximates the shape sensed for by the hand. The left side of the figure shows the robot grasping the true object made up of cylinders. The right side is a visualisation of the GPISP belief after collecting the observations. It is a mesh representation of the implicit surface defined by the maximum a posteriori estimation of the GPISP.

With only a few grasps the system builds up a fair internal GPISP representation of the shape, and the concavity is obviously not an issue. However, the control here is completely scripted: a human expert navigates the hand and closes the fingers at the proper place.

The above is quasi complementary to the particle demonstration in Section 6.3: there we abstract the actuators but include active control, while above the control is manual but the actuators are modeled.

7.3 Feedback scheme for non-convex object estimation

The particles from the demonstration in Section 6.3 abstract kinematic and mass constraints, thus they are very responsive and lightweight. On the other extreme is the full fledged global trajectory optimisation from Section 7.1 capable to deal with complex grasp problems. Here with the particle scenario in mind, we go some steps closer to reality. We present a scheme for feedback control of a robot arm for object estimation. It is simple, yet it takes into account the kinematics of the arm, the joint limits, and heuristically deals to some extent with collisions.

7.3.1 Setup

The simulation environment is the same as in Section 7.2. It is used to simulate an arm able to localise contacts on its end effector. The particular target platform is the Barret WAM. An accurate model of the arm is placed on a table (Figure 7.7). In front of it, in the robot's workspace we put a target object inspired by a valve handle. It is modelled as a hexagon with 3 spokes. All sides and spokes are capped cylinders and all live in a plane, despite some observers perceive the visual illusion of a wire model of a cube. (Note that the valve handle used in the kinesthetic sensing in Section 7.4 is different: a ring with a diameter, i.e. it is oval and has 2 rather than 3 spokes.)

The kinematic redundancy is resolved by identifying the end effector task and its null space and projecting additional tasks – self-collision avoidance and joint limits – in the null-space. Other collisions are not taken into account.

The hand configuration remains static during execution: two of the fingers are closed and fixed (in joint space) and excluded from the Jacobian of the end effector task. The end effector is the tip of the third finger. Contacts are registered only between this finger tip and the valve. The Barret WAM has tactile arrays at the fingertips, which are not modelled here; the contact is measured at a small sphere virtually attached to the tip.

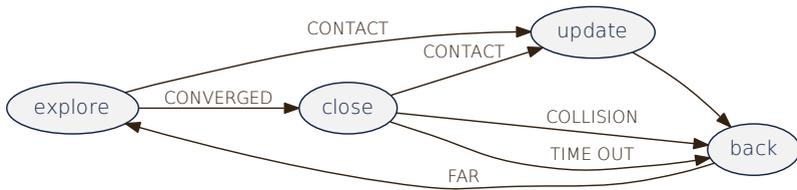


Figure 7.6: State machine of a feedback estimation sequencer. The edge labels denote the occurrence of an event which triggers the transition between states. States denote a control ‘regime’.

Contacts deliver evidence for updating the GPISP model, i.e. the belief about the shape. The end effector task works on this model only, i.e. the system is unaware of the true valve in any way. Touch is the only modality which provides observations, starting with a virtual one somewhere on the surface. The robot is then moved by uncertainty-aware control laws to actively collect observations from the surface.

7.3.2 Feedback control

The exploration starts with an externally provided initial observation and the corresponding belief. The robot attempts to obtain information from the most uncertain parts of the surface. Upon detecting a contact the model is updated and the potential repulses the effector until it is away from the surface, then it starts again exploring. If the effector is on what the system believes is the surface, but does not feel contact (the model overestimates), it follows blindly and optimistically the direction normal to the belief surface for a while, hoping to reach the real surface. If this works, it updates and retracts normally; if no contact is registered, it simply retracts without updating the model. This can be formalised as the state automaton in Figure 7.6. *Explore* refers to the execution of the explore-grasp primitive working on a GPISP model with single end

effector. A CONTACT stops the execution, *updates* the model and switches to the *back* primitive. The difference between *back* and *explore* is whether they follow the positive or the negative potential gradient, resulting in attraction or repulsion. *Close* borrows its name from what would happen in a 3-finger scenario if the hand reached the belief surface without contact: the fingers would reflex-like close until they self-collide or contact the object (*close()* in Algorithm 2 has the same semantic). Here it is rather a linear motion continuing in the last direction suggested by the surface normal. In more practical terms, there is a threshold of the potential value, below which the transition is triggered. The FAR potential value, on the other hand, is used to denote when the repulsion can go into attraction again, i.e. from *back* to *explore*. Changing *explore* with *exploit* makes the state machine describe a robust-grasping controller. And with an additional logic module to decide on exploration vs. exploitation on each entrance of this state, one can realise arbitrary policies for interactive grasping of unknown objects. Algorithm 3 gives a sketch of the control loop in pseudo-code.

7.3.3 Notes on collision avoidance

For the sake of keeping the feedback controller simple, it does not employ grasp object collision tvs. (Collision avoidance for known objects can be integrated, if there were obstacles; collisions with the target object are non-trivial since all the system has access to is the belief. Why is collision handling here different than in Section 7.1? What makes the difference is the assumption that the contact can be not only registered but localised and directly brings information. Here contacts can be localised at limited areas only. The rest is at best uninformative.) A collision tv would be limitedly useful in the beginning anyway, when the belief and the true object differ strongly. The rudimentary collision avoidance is due to – already in an early reach stage – orienting the hand along the negative potential gradient. This makes the arm

Algorithm 3 Variance feedback control for estimation

Require: potential $f()$, forward kinematic ϕ ;**Require:** robot start position \mathbf{q}_0 ;**Require:** initial observation \mathbf{o}_0 (and model $f \sim \mathcal{GP} | \mathbf{o}_0$);**Require:** movement costs and task costs (grasp objective)
 $c_g(\mathbf{q}), c_t(f(\mathbf{q}), \mathbb{V}[f(\mathbf{q})], \mathbf{q}, \dots)$ **Require:** $l(\mathbf{q}) = c_g(\mathbf{q}) + c_t(f(\mathbf{q}), \mathbb{V}[f(\mathbf{q})], \mathbf{q}, \dots)$ \triangleright differentiable cost function**Require:** $\tau_{\text{NEAR}}, \tau_{\text{FAR}}$: thresholds on f ; and τ_p : max path length

```

1:  $\mathbf{q} \leftarrow \mathbf{q}_0$ 
2:  $\mathcal{S} \leftarrow \text{grasp}$   $\triangleright$  state machine initialisation ( $\text{grasp} \in \{\text{exploit}, \text{explore}\}$ )
3: loop
4:   switch  $\mathcal{S}$  do
5:     case grasp:
6:        $\Delta\mathbf{q} \propto \frac{\partial}{\partial\mathbf{q}} (l(f(\mathbf{q}), \dots))$   $\triangleright$  step to max var and min potential
7:       if CONTACT then  $\mathcal{S} \leftarrow \text{update}$ 
8:       if  $f(\mathbf{q}) < \tau_{\text{NEAR}}$  then  $\mathcal{S} \leftarrow \text{close}$   $\triangleright$  CONVERGED ( or long time near surface)
9:     case update:
10:       $\Delta\mathbf{q} \leftarrow \mathbf{0}$ 
11:       $\mathbf{o}_i \leftarrow (\phi(\mathbf{q}), \mathbf{t}_n)$   $\triangleright$  get new observation (point, normal)
12:       $f \sim \mathcal{GP} | \mathbf{o}_{0:i}$   $\triangleright$  update
13:       $\mathcal{S} \leftarrow \text{back}$ 
14:     case back:
15:       $\Delta\mathbf{q} \propto \frac{\partial}{\partial\mathbf{q}} (l(-f(\mathbf{q}), \dots))$   $\triangleright$  invert potential, go to max var and max potential
16:      if  $\tau_{\text{FAR}} < f(\mathbf{q})$  then  $\mathcal{S} \leftarrow \text{grasp}$   $\triangleright$  far from surface
17:     case close:
18:       $\Delta\mathbf{q}$  s.t.  $\phi(\mathbf{q} + \Delta\mathbf{q})$  continues the linear motion in euclidean space
19:      if CONTACT then  $\mathcal{S} \leftarrow \text{update}$ 
20:      if COLLISION then  $\mathcal{S} \leftarrow \text{back}$ 
21:      if  $\tau_p < \|\mathbf{x} - \mathbf{x}'\|$  then  $\mathcal{S} \leftarrow \text{back}$   $\triangleright$  time out (or max path length)
22:       $\mathbf{q} \leftarrow \mathbf{q} + \Delta\mathbf{q}$ 
23:   end loop

```

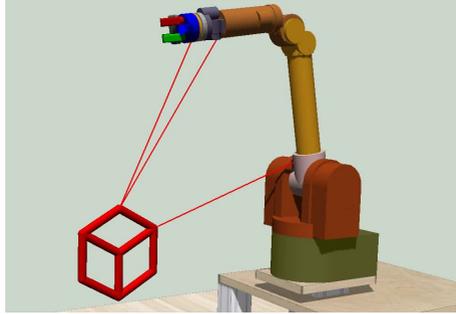


Figure 7.7: Barret WAM and valve handle in simulation.

‘follow’ the hand. In other words, when still FAR the objective function puts most priority on the hand orientation along the direction the finger should go (negative gradient). Once this is the case, the end effector approaches the surface belief and the arm follows, effectively avoiding the working space which contains the object. Of course, this is not guaranteed to work well. If it fails, the arm collides with the object and one can only rely on the compliance of the hardware to recover.

7.3.4 Demonstration

Figure 7.7 shows the simulation setup. The valve is modeled as a hexagon with spokes. Snapshots from a run of the feedback controller are shown in Figure 7.8. The transparent wire-models of the bodies at the most images allow to see the growing estimate (gray) with new observations. The corresponding models in Figure 7.9 are colored according to the variance. This is how the belief develops by collecting more information from the surface. The coloring provides insight why the particular contact point is chosen. Note the locality of the behaviour here compared to the planned trajectory scheme in the experiment from Section 7.1.4. There, a grasp was more likely to deliver an observation from the other side because the motion started from a far point. Here the sequencer does not take the actuator that much back. Consequently,

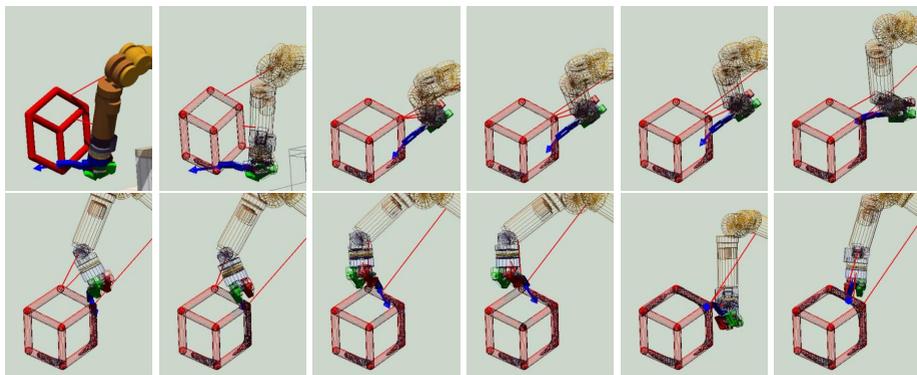


Figure 7.8: Estimates of a valve handle produced by sensing with one finger of a WAM arm in simulation.

sliding near the belief surface to a higher variance, the end effector tries to go around the 'end' of the surface and contacts the real surface just there.

7.4 Kinesthetically moved arm with real feedback

The simulation used for the experimental results above is quite precise, but it remains model of the real world which abstracts physical interactions assumed not important. It may turn out that the simulation neglects too aggressively, thus we seek to demonstrate this aspects of our algorithm on a real robot system. One of the assumptions in the trajectory optimisation experiment section is that the robot can detect contacts with its whole body surface, like humans do. This is not the case in practice to date: robotic hardware can either measure torques in a joint or have very limited area covered with tactile arrays. The former detect whether the kinematic subtree (starting at this joint) experiences external forces; the latter yield a relatively precise pressure distribution but for small part of the body. Special care is to be taken to ensure that contacts happen where sensors can measure their effects. The previous section relaxes the above assumption and only relies on contacts on the fin-

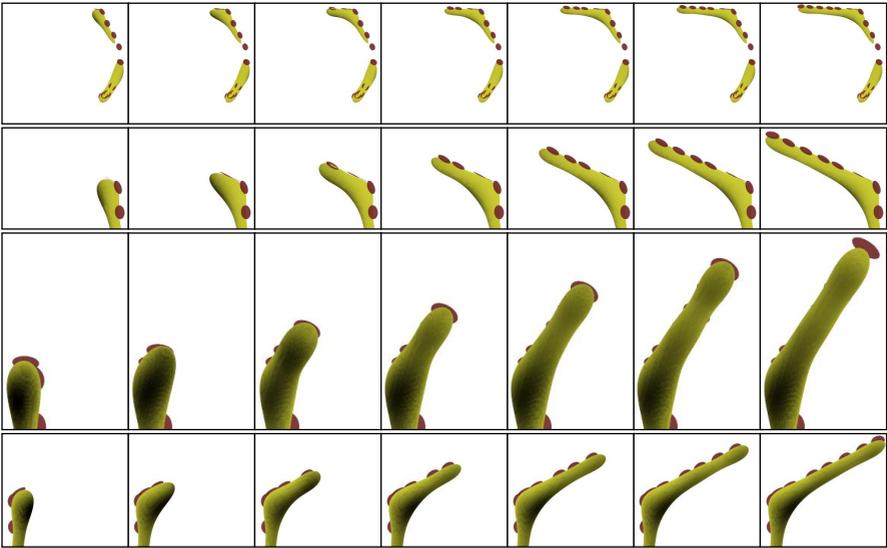


Figure 7.9: GPISP belief evolving with seven successive surface observations in simulation. The rows show the same process from different perspective.

gertip. It still assumes quite precise estimate of the contact derived directly from the kinematic chain. In a further experiment, we investigate whether this is feasible on a real robot.

The setup includes a Barret WAM and a valve handle which is to be estimated by sensing with the arm. Figure 7.10 shows the setup. The WAM is equipped with tactile sensors on the finger tips, torque sensors in the arm joints, and a strain gauge per finger. For the experiment we fold two of the fingers of the robot, as in the previous section, and touch the object with the tip of the third finger. A contact is assumed when a strain gauge reports a value beyond certain threshold. We use a gravity compensation controller: the system maintains a stable configuration as long as it does not experience other external forces. In this setup a human can move the arm, orient the hand and direct it to the object so that the fingertip contacts the surface. The position and orientation of the fingertip (and thus of the contact) is computed

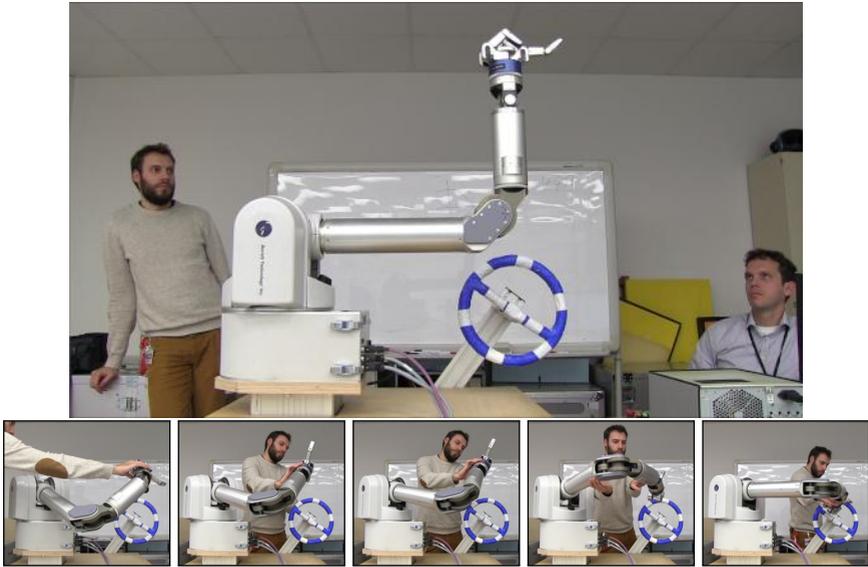


Figure 7.10: The setup used for the kinesthetic sensing demonstration as well as for transferring the sequencer to real hardware: A Barret WAM arm and hand and a valve handle in white and blue stripes.

from the kinematic model of the arm (the one used in the previous simulation section). The control is intentionally not taken too precise, i.e. the position of the contact is noisy (it is not necessarily precisely at the tip and not fully aligned with the surface).

Figure 7.11 presents the result of the kinesthetic estimation: each row shows different views of the final estimate after a sequence of contacts. The red disks depict the position as measured by the strain gauges readings and the orientation of the tip as inferred from forward kinematics in the moment of contact. Note that the discs tend to be on the one side of the valve plane only, yet the smoothness prior causes the estimate to extend in space. It rounds up on the opposite side resulting in a complete estimate. Consistent with the lack of observations the variance is much higher on the opposite side of the model. This effect is best seen in the upper run – worst of the four – which falsely

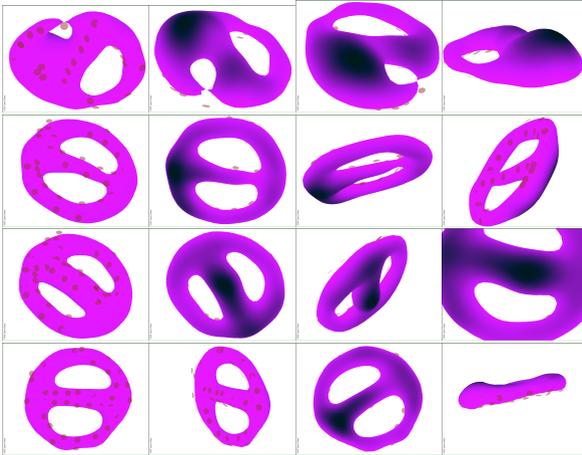


Figure 7.11: Estimates of a valve handle produced by sensing with an WAM arm kinesthetically controlled by a human. Disks show the contact points; surface color encodes the model uncertainty of the region. Each row shows four views of a single run.

predicts a huge bump where no such exists, but its dark color warns that the model confidence is low there. The second run, on the other hand, predicts much more confidently one part of the spoke for which there are slanted contacts on both inner sides of the spoke.

Given the imprecision of the sensing and the number of contacts the results demonstrate that the acquisition of a fairly good working model is possible with real sensors and under moderately controlled circumstances. Using tactile array patterns or other more precise modalities can improve accuracy and reliability.

7.5 Summary

The GPISP task spaces for estimation and grasping can be used to handle uncertainty in variety of settings. A particular benefit is that GPISP plays well with concave objects, both for estimation and for grasping. The most important res-

ult so far is the realisation of a policy which uses exclusively explore-grasps: it integrates the novel control laws into a holistic framework for generating motion for repeating reactive grasps. The experiment conducted with it illustrates that uncertainty-aware sensory-motor control brings accuracy and efficiency, generalising well across objects of different sizes and positions. However, transferring this implementation to real hardware is challenging. In further demonstrations, we relax one by one assumptions which may hinder such transfer. In another setting, we use the Barret WAM arm as a kinematic chain with single end effector. It shows a closed-loop flavour of the framework: a simpler approach to repetitive grasping which does not rely on contact registration at every body part, but takes advantage of the compliance of the hardware. Its goal is to sense for an object at a limited body area only, the finger tip. While dropping one assumption, it still relies on simulated tactile data. With a kinesthetically moved arm we verify that obtaining real tactile data is feasible: decent GPISP models, including the variance features, are acquirable in a real world setting.

The proper implementation in real systems calls for sensors to at least register contacts which are not locatable and for compliant arms which gracefully recover from unexpected contacts.

Chapter 8

Conclusions

In this chapter we summarise the most important contributions of the thesis, emphasise strengths and discuss limits which suggest directions for further research.

8.1 Summary

We are often captured between the walls of our senses and habits. They judge what is right, plausible, obvious. The habits impose strong priors on the principles behind this adverse and uncertain world: they proved to be good for surviving in it. However, when we want to explore new structures, we mostly need to go beyond the superficially plausible. This is the case when we want to introspectively reason about ourselves and what we consider intelligent. In this thesis, we act at a low-level, usually hardly perceivable, aspect of intelligence – the sensory-motor primitives.

The shape of an object can be described as points from its surface or patches which follow the surface. This is obvious because the surface is mostly what we see in our visually dominated world and what we intentionally deal with. But when the purpose of the description is different, an alternative de-

scription may be better. Gaussian process implicit surface potentials are about estimation and interaction. The GPISP idea still suggests that the surface of the object is most important; the surface gives rise to the potential field. At the same time, GPISP does not ease the retrieval of the surface. In particular, unlike a mesh it is not a visual representation which contains the surface information explicitly and can be directly drawn, polygon by polygon. A GPISP model targets other applications: it allows to interact with an object by moving in its virtual potential field. While giving up an obvious merit in one domain, it explains better another domain which is not obvious to deal with – manipulation.

Acquiring shape models – for the sake of manipulation – cannot be isolated from manipulation: it is manipulation itself. Data is rarely just there, it needs to be collected, which is matter of interaction with a shape. To this end, the representation offers both task spaces for navigation around a surface and the possibility to update the model with new data – providing common ground for integration of sensing and moving. The task spaces are focused on the shape geometry. The features that they extract care about motion and interaction. For instance, 2 and 20 meter away the potential hardly differs, while in the vicinity of the surface it closely resembles the distance to the object.

A GPISP model describes the surface as generated by a function defined everywhere. This allows to add one more dimension to the representation: confidence. Starting with a prior for the uncertainty, it takes the confidence of the measurements into account and translates them altogether into the variance of a distribution over shapes. The variance available that conveniently across the model, in turn, can be used as motion feature. Moving along or against its gradient are tasks which bias manipulation to known or unknown parts of the model. They make for faster and more accurate exploration or for more robust manipulation.

Putting it in a single dense sentence: we think of GPISP and the suggested tasks as a reminiscent of an object-centric, geometric, learnable, uncertainty-

aware, constructive grasp affordance.

In the introduction we referred to changes of viewing angle. Turning from visual representation to potentials can be considered one. Lots of them need to be explored and learned from before we understand the principles of what we call intelligence to an extent which allows to replicate it, create intelligence. A lot of work has been done,

...But, I should warn you, this is no time for complacency. No, there are still many things, and I cannot emphasize this too strongly, *not* on top of other things. I myself, on my way here this evening, saw a thing that was not on top of another thing in any way. (shame!) Shame indeed...

It could be hardly said better than the President of the Royal Society for Putting Things on top of Other Things does (Chapman et al., 1970). In contrast to them, we are confident that we are not ‘a meaningless body of men that had gathered together for no good purpose’: grasping and manipulation are not solved; connection between low level intelligence and high level reasoning is not solved; links between intelligent hardware and intelligent algorithms are not established; building intelligent machines is not solved. Research which reveals these principles is worth. ‘...we flourish,’ the president adds.

8.2 Future research suggestions

Some limitations of GPISP straightaway suggest directions in which to seek for solutions. Furthermore, there is potential to extend and combine with other approaches.

8.2.1 Representations business

An interesting view on shapes is provided when examining structural similarities. Instead of traditional approaches based on primitives, Pokorny et al. (2013) make very few assumptions on the structure which allows to reason

on very general level – e.g. presence of holes in the shape. Other topological arguments lead to the writhe-matrix representation and interaction meshes in Zarubin et al. (2012) which very naturally support other kind of motions, closely related to interaction. However, they do not intrinsically have a notion of near or far, for instance; the writhe can tell that an object is sufficiently enclosed (caged) by the hand, but can hardly show the way to establish contact. With GPISP these shortcomings can potentially be alleviated. How to integrate both is open question, though. Is there a way to find the topology of a surface encoded in a GPISP model bypassing the explicit polygonisation is a worthwhile question.

Alternative representations may be used to the end of relaxing the assumption that the objects are fixed. Inspiration can be borrowed from researchers which work on orthogonal types of uncertainty: Hsiao et al. examine object pose hypotheses for known objects. Maintaining a belief over the orientation and having a rudimentary model of how interaction impacts pose could be a starting point.

In the very depths of our representation there are also blank areas to explore. An interesting and relevant question is whether there exist structured priors which imply, e.g., cylinder or sphere as implicit surface. A related direction is the search for covariance functions which can better describe edges. This can make the GP suffer less computational overhead due to less data needed for retaining high accuracy.

8.2.2 Tactile array patterns

We assumed a rather rudimentary way of tactile data acquisition which is more appropriate to the strain gauges of a finger than to a tactile array. In fact, what tactile arrays deliver is a pressure distribution. Two simplistic ways to obtain points and normals are to i) take each texel as full observation or ii) aggregate neighbours to a single point (and normal). More elaborately,

one can reason about what specific local surface properties induce which distribution. This has the potential to reduce the data streams without losing precision. Examples include slim 2D Gaussian which translates to an edge; or correction of the sensor normal by the direction of the principal component of the distribution.

8.2.3 Object recognition as active learning and model selection

A demonstration scenario which is more similar to the task Bob accomplishes in the Introduction is to recognise previously known objects, rather than estimate. There is an obvious extension to the exploration algorithm which in addition ranks the known surface models by similarity to the current belief. But this scenario offers even more: the estimation can benefit from the prior over all existing objects. They further constrain how the posterior interprets the evidence. Furthermore, this can be interpreted as a simultaneous active model selection and active learning: it both finds the best parametrisation and the best generating model by cleverly guiding the end effector to meaningful information.

8.2.4 Transfer to real robots, alternative scenarios

In Chapter 7, we demonstrate the feasibility of different aspects of a system running on real robots, however, the realisation stays away. The partial results shown here need to be integrated into a complete demonstration: a non compliant robot, should be at least made aware of detecting unlocalized and treat them as a failure, leading to retract and next try; on the other hand, compliant hardware should tolerate such contacts and be able to run the feedback control loop with small modification. Topics which are necessary to look at include the problems arising from limited tactile sensors capable of registering contacts, more sophisticated grasp-specific explore-exploit policies, as well as more sophisticated controllers. An example for the latter is the easy

but naïveralisation of the *back()* step in Algorithm 2. More elaborate ways to regrasp include sliding the fingers along the variance projection on the tangent plane, while staying on zero-level and maintaining contact with the true surface. Li et al. (2013) consider similar problem: they use local tactile patterns for tactile servoing – sliding a finger along surface, tracking local features, etc.

8.2.5 Learning surface priors

The process of estimating from sensory information depends on several hyperparameters and the choice of covariance function for the Gaussian process. We argued that the embodiment suggests a quite strong prior which we in practice subsume to a single number. The more realistic and more adequate Bayesian way could be to have a nontrivial prior. Such priors can be obtained by identifying (from robotics point of view) relevant contexts, objects and goals, and learn in a supervised or unsupervised way their relations to the hyper-parametrisation.

8.2.6 Mixtures of GPs for shape estimation

Yet another way to stay at sane data rates and have very fine estimates is to exploit the fact that shapes have different level of detail in different parts. The parametrisation of single vanilla GP does not allow to capture this heterogeneity. A way around this challenge is to use non-stationary parameters or to setup different GP for different regions.

Such separation makes sense also from the perspective of what is the purpose of the model: it may be suitable to use one GP for control, for collecting observations, and other for evaluating them. In other words, the different GPs work on the same observation set, they merely interpret them differently.

8.2.7 No information left behind

When a robot moves, it implicitly collects information about the object and environment, even when it does not contact surfaces. To get information, in terms of GPISP, means to modify the posterior belief. What comes into a GPISP changes either the MAP estimate or the confidence of the model. We want additional information to modify this distribution – either the mean, the variance or both. An example for the benefit from such information is the following: imagine two bubbles resulting from two surface observations and empty space between them. This emptiness can result from the smoothness prior (narrow covariance) but also from really absent surface. Thus, the model is not confident in this area before seeing more evidence. A way to resolve this ambiguity is to move a finger between the two bubbles. Even without a contact, the variance should drop. So far, we do not support this. What are decent ways to do this?

Point and normal observations are not suitable for information in the absence of contacts. It is unnatural to fit what is not a contact into contact terms and leads easily to, e.g., violation of the monotonicity and thus of the interpretation as potential. A promising direction is to separately estimate a ‘prior’ for the variance while moving. This results in a non-stationary ‘prior’ which is used in the GPISP. A second GP can be used to this end, fed by data from trajectories.

There are two pieces of information we are after: i) use information from moving collision-free through space; and ii) learn from a contact which we expect but does not happen. The above suggestion addresses the first case. The second case occurs due to overestimation of a surface so that a manipulation trajectory expects to come in contact, but can never reach the true surface hidden behind the estimate. Currently we resolve this issue in control: by blindly following the most recent encountered gradient. We do not have mechanism to change the posterior without seeing a contact so far.

We flourish.

Appendix A

Derivatives related to GPISP

The tvs that implement the grasp heuristics in the controller in Chapter 5 need Jacobian and Hessian of the potential field associated to an implicit surface (IS). We derive them for parametric shapes (where no learning is involved) and for learned ones (GPISP).

A.1 Basic shape derivatives

We parametrize the IS representation of basic shapes by (a subset of) the object center \mathbf{c} , radius r , height h , orientation \mathbf{z} , and scale factor s . The surface is given as the zero-level set of

$$\phi = 1 - e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2},$$

where $d = d(\mathbf{c}, r, h, \mathbf{z}, s)$. Note: in this realisation, ϕ depends only on the distance to surface d ; $\phi(\mathbf{x}) = 0 \iff$ ‘ \mathbf{x} is a surface point’; $\phi(\mathbf{x})$ tends to 1 when \mathbf{x} *far* from the object; the notion of *far* is governed by s ;

We need the first and second derivative of ϕ w.r.t. \mathbf{x} .

A.1.1 Gradient

For the gradient of ϕ we have:

$$\begin{aligned} \left(\frac{\partial \phi}{\partial x_i} \right)_{i=1..3} &= \frac{\partial \phi}{\partial \mathbf{x}} = \frac{\partial \phi}{\partial d} \frac{\partial d}{\partial \mathbf{x}} \\ &= -e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} (-(d+1)) \frac{\partial d}{\partial \mathbf{x}} \\ &= e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} (d+1) \frac{\partial d}{\partial \mathbf{x}}. \end{aligned}$$

A.1.2 Hessian

For the Hessian: $\frac{\partial^2 \phi}{\partial x_j \partial x_i}$ with $i, j \in 1..3$

$$\begin{aligned} \frac{\partial^2 \phi}{\partial x_i \partial x_j} &= \frac{\partial}{\partial x_j} \left(\frac{\partial \phi}{\partial x_i} \right) = \frac{\partial}{\partial x_j} \left(\frac{\partial \phi}{\partial d} \frac{\partial d}{\partial x_i} \right) \\ &= \frac{\partial d}{\partial x_i} \frac{\partial}{\partial x_j} \left(\frac{\partial \phi}{\partial d} \right) + \frac{\partial \phi}{\partial d} \frac{\partial}{\partial x_j} \left(\frac{\partial d}{\partial x_i} \right) \\ &= \frac{\partial d}{\partial x_i} \frac{\partial}{\partial x_j} \left(\frac{\partial \phi}{\partial d} \right) + \frac{\partial \phi}{\partial d} \frac{\partial^2 d}{\partial x_i \partial x_j} \\ &= \frac{\partial d}{\partial x_i} \frac{\partial}{\partial d} \left(\frac{\partial \phi}{\partial d} \right) \frac{\partial d}{\partial x_j} + \frac{\partial \phi}{\partial d} \frac{\partial^2 d}{\partial x_i \partial x_j} \\ &= \frac{\partial d}{\partial x_i} \frac{\partial^2 \phi}{\partial d^2} \frac{\partial d}{\partial x_j} + \frac{\partial \phi}{\partial d} \frac{\partial^2 d}{\partial x_i \partial x_j}. \end{aligned}$$

Eventually, $\frac{\partial d}{\partial x_k}$ is known for any k , and $\frac{\partial \phi}{\partial d} = e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} (d+1)$. Remains to calculate $\frac{\partial^2 d}{\partial x_i \partial x_j}$ and $\frac{\partial^2 \phi}{\partial d^2}$. The former must be provided by the particular

GraspObject (e.g. sphere, cylinder), the latter is derived in the following.

$$\begin{aligned}
 \frac{\partial^2 \phi}{\partial d^2} &= \frac{\partial}{\partial d} \left(\frac{\partial \phi}{\partial d} \right) = \frac{\partial}{\partial d} \left(e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} (d+1) \right) \\
 &= e^{\frac{1}{2}} \frac{\partial}{\partial d} \left(e^{-\frac{1}{2}(d+1)^2} \right) (d+1) + e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} \frac{\partial (d+1)}{\partial d} \\
 &= e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} \frac{\partial}{\partial d} \left(-\frac{1}{2} (d+1)^2 \right) (d+1) + e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} 1 \\
 &= -\frac{1}{2} e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} \frac{\partial (d+1)^2}{\partial d} (d+1) + e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} \\
 &= -\frac{1}{2} e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} 2(d+1)(d+1) + e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} \\
 &= -e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} (d+1)^2 + e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} \\
 &= -e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} ((d+1)^2 - 1) = -e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2} d(d+2).
 \end{aligned}$$

Finally, substituting what we derived and using $E := e^{\frac{1}{2}} e^{-\frac{1}{2}(d+1)^2}$, we get for the Hessian

$$\frac{\partial^2 \phi}{\partial x_i \partial x_j} = -d(d+2)E \frac{\partial d}{\partial x_i} \frac{\partial d}{\partial x_j} + d(d+1) \frac{\partial^2 d}{\partial x_i \partial x_j},$$

and in vector notation, with $\mathbf{H} = \mathbf{H}_{3 \times 3} := \frac{\partial^2 d}{\partial \mathbf{x}} = \left(\frac{\partial^2 d}{\partial x_i \partial x_j} \right)_{i,j}$

$$\begin{aligned}
 \nabla \nabla \phi &= \frac{\partial^2 \phi}{\partial \mathbf{x}^2} \\
 &= -d(d+2)E \left(\frac{\partial d}{\partial \mathbf{x}} \right) \left(\frac{\partial d}{\partial \mathbf{x}} \right)^T + d(d+1)\mathbf{H}.
 \end{aligned}$$

Now we have the gradient and the Hessian of an abstract parametric shape in terms of $\frac{\partial d}{\partial \mathbf{x}}$ and $\frac{\partial^2 d}{\partial \mathbf{x}}$. These terms vary from one shape to another. Thus they need to be provided by the particular shape. This is the subject of the following sections.

A.1.3 Sphere derivatives

For the IS representation of a sphere with $d = s(\|\mathbf{x} - \mathbf{c}\| - r)$, the gradient and Hessian are:

$$\begin{aligned}
 \frac{\partial^2 d}{\partial x_i \partial x_j} &= \frac{\partial}{\partial x_j} \left(\frac{\partial d}{\partial x_i} \right) = \frac{\partial}{\partial x_j} \left(s \frac{x_i - c_i}{\|\mathbf{x} - \mathbf{c}\|} \right) \\
 &= s \frac{\partial}{\partial x_j} \left(\frac{x_i - c_i}{\|\mathbf{x} - \mathbf{c}\|} \right) \\
 &= s \frac{1}{\|\mathbf{x} - \mathbf{c}\|^2} \left(\frac{\partial(x_i - c_i)}{\partial x_j} \|\mathbf{x} - \mathbf{c}\| - \frac{\partial \|\mathbf{x} - \mathbf{c}\|}{\partial x_j} (x_i - c_i) \right) \\
 &= s \frac{1}{\|\mathbf{x} - \mathbf{c}\|^2} \left(\delta_{ij} \|\mathbf{x} - \mathbf{c}\| - \frac{\partial \|\mathbf{x} - \mathbf{c}\|}{\partial x_j} (x_i - c_i) \right) \\
 &= s \frac{1}{\|\mathbf{x} - \mathbf{c}\|^2} \left(\delta_{ij} \|\mathbf{x} - \mathbf{c}\| - \frac{(x_i - c_i)(x_j - c_j)}{\|\mathbf{x} - \mathbf{c}\|} \right) \\
 &= s \delta_{ij} \|\mathbf{x} - \mathbf{c}\|^{-1} - s(x_i - c_i)(x_j - c_j) \|\mathbf{x} - \mathbf{c}\|^{-3} \\
 &= s \|\mathbf{x} - \mathbf{c}\|^{-1} \left(\delta_{ij} - (x_i - c_i)(x_j - c_j) \|\mathbf{x} - \mathbf{c}\|^{-2} \right).
 \end{aligned}$$

A.1.4 Infinite cylinder

An infinite cylinder is a cylinder with finite radius and infinite length. The distance to the surface of an infinite cylinder with center \mathbf{c} , radius r , orientation \mathbf{z} and scale s , with dot product $\langle \cdot, \cdot \rangle$ and Euclidean norm $\|\mathbf{a}\| = \sqrt{\mathbf{a}^T \mathbf{a}}$ is

$$d = s(\|(\mathbf{x} - \mathbf{c}) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}\| - r).$$

The intuition for the gradient of the field is that it points outwards and is normal to the surface. The Hessian should reflect the fact that the gradient does not change along the z axis. Formally, the gradient of the distance is

then

$$\nabla d = s \frac{(\mathbf{x} - \mathbf{c}) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}}{\|(\mathbf{x} - \mathbf{c}) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}\|},$$

and each component has the form

$$\frac{\partial d}{\partial x_i} = s \frac{(x_i - c_i) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle z_i}{\|(\mathbf{x} - \mathbf{c}) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}\|}.$$

Let $\mathbf{a} := (\mathbf{x} - \mathbf{c}) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}$. Then $\mathbf{a}_i = (x_i - c_i) - \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle z_i$. To double check and convince ourselves, the analytical derivation of the above in terms of \mathbf{a} , i.e. $\nabla d = s \frac{\mathbf{a}_i}{\|\mathbf{a}\|}$, follows.

$$\begin{aligned} \frac{\partial d}{\partial x_i} &= \frac{\partial s(\|\mathbf{a}\| - r)}{\partial x_i} = s \frac{\partial \|\mathbf{a}\|}{\partial x_i} = s \frac{1}{2\|\mathbf{a}\|} \frac{\partial \mathbf{a}^T \mathbf{a}}{\partial x_i} \\ &= s \frac{1}{2\|\mathbf{a}\|} \frac{\partial \sum_k \mathbf{a}_k^2}{\partial x_i} = s \frac{1}{2\|\mathbf{a}\|} \sum_k \frac{\partial \mathbf{a}_k^2}{\partial x_i} \\ &= s \frac{1}{2\|\mathbf{a}\|} \left(\frac{\partial \mathbf{a}_i^2}{\partial x_i} + \sum_{k \neq i} \frac{\partial \mathbf{a}_k^2}{\partial x_i} \right) = s \frac{1}{2\|\mathbf{a}\|} \left(2\mathbf{a}_i \frac{\partial \mathbf{a}_i}{\partial x_i} + \sum_{k \neq i} \frac{\partial \mathbf{a}_k^2}{\partial x_i} \right) \\ &= s \frac{1}{2\|\mathbf{a}\|} \left(2\mathbf{a}_i(1 - \mathbf{z}_i \mathbf{z}_i) + \sum_{k \neq i} \frac{\partial \mathbf{a}_k^2}{\partial x_i} \right) \\ &= s \frac{1}{2\|\mathbf{a}\|} \left(2\mathbf{a}_i(1 - \mathbf{z}_i \mathbf{z}_i) + \sum_{k \neq i} 2\mathbf{a}_k \frac{\partial \mathbf{a}_k}{\partial x_i} \right) \\ &= s \frac{1}{2\|\mathbf{a}\|} \left(2\mathbf{a}_i(1 - \mathbf{z}_i \mathbf{z}_i) + 2 \sum_{k \neq i} \mathbf{a}_k (-\mathbf{z}_k \mathbf{z}_i) \right) \end{aligned}$$

With some rearranging

$$\begin{aligned}
 &= s \frac{1}{\|\mathbf{a}\|} \left(\mathbf{a}_i (1 - \mathbf{z}_i \mathbf{z}_i) + \sum_{k \neq i} \mathbf{a}_k (-\mathbf{z}_k \mathbf{z}_i) \right) \\
 &= s \frac{1}{\|\mathbf{a}\|} \sum_k \mathbf{a}_k (\delta_{ik} - \mathbf{z}_k \mathbf{z}_i) \\
 &= s \frac{1}{\|\mathbf{a}\|} \left(\sum_k \mathbf{a}_k \delta_{ik} - \mathbf{z}_i \sum_k \mathbf{a}_k \mathbf{z}_k \right) \\
 &= s \frac{1}{\|\mathbf{a}\|} (\mathbf{a}_i - \mathbf{z}_i \mathbf{a}^T \mathbf{z}).
 \end{aligned}$$

And $\mathbf{a}^T \mathbf{z} = 0$ since $\mathbf{a} \perp \mathbf{z}$, thus

$$\frac{\partial d}{\partial x_i} = s \frac{\mathbf{a}_i}{\|\mathbf{a}\|} \quad \square$$

Now deriving once more, we obtain for the Hessian

$$\begin{aligned}
 \frac{\partial^2 d}{\partial x_i \partial x_j} &= \frac{\partial s \|\mathbf{a}\|^{-1} \mathbf{a}_i}{\partial x_j} \\
 &= s \frac{\partial}{\partial x_j} \left(\frac{\mathbf{a}_i}{\|\mathbf{a}\|} \right) = s \frac{1}{\|\mathbf{a}\|^2} \left(\frac{\partial \mathbf{a}_i}{\partial x_j} \|\mathbf{a}\| - \mathbf{a}_i \frac{\partial \|\mathbf{a}\|}{\partial x_j} \right) \\
 &= s \frac{1}{\|\mathbf{a}\|^2} \left(\frac{\partial \mathbf{a}_i}{\partial x_j} \|\mathbf{a}\| - \mathbf{a}_i \frac{1}{\|\mathbf{a}\|} \mathbf{a}_j \right) \\
 &= s \frac{1}{\|\mathbf{a}\|^2} \left((\delta_{ij} - \mathbf{z}_j \mathbf{z}_i) \|\mathbf{a}\| - \mathbf{a}_i \frac{1}{\|\mathbf{a}\|} \mathbf{a}_j \right) \\
 &= s \frac{1}{\|\mathbf{a}\|} \left(\delta_{ij} - \mathbf{z}_j \mathbf{z}_i - \mathbf{a}_i \mathbf{a}_j \frac{1}{\|\mathbf{a}\|^2} \right).
 \end{aligned}$$

Note that this can be rewritten (for the sake of efficient computation where

intermediate results from gradient components can be reused) as

$$= s \frac{1}{\|\mathbf{a}\|} \left(\delta_{ij} - \mathbf{z}_j \mathbf{z}_i - \frac{\partial \|\mathbf{a}\|}{\partial x_i} \frac{\partial \|\mathbf{a}\|}{\partial x_j} \right).$$

In vector notation it translates to

$$\begin{aligned} \frac{\partial^2 d}{\partial \mathbf{x}^2} &= \left(\frac{\partial^2 d}{\partial x_i \partial x_j} \right)_{i,j} \\ &= s \frac{1}{\|\mathbf{a}\|} \left(\mathbf{I}_3 - \mathbf{z}\mathbf{z}^T - \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} \right). \end{aligned}$$

A.1.5 Finite cylinder

A finite cylinder is defined by center \mathbf{c} , radius r , orientation \mathbf{z} height h and scale s . For the gradient and Hessian of a finite cylinder we consider three cases for \mathbf{x} :

- \mathbf{x} between the cutting planes of the cylinder,
- \mathbf{x} outside the cutting planes, inside the extended (uncut) cylinder,
- \mathbf{x} outside the cutting planes, outside the extended cylinder.

Between the bottom and the top, the function behaves like in the infinite case. Derivations for $\frac{\partial d}{\partial x}$ and $\frac{\partial^2 d}{\partial \mathbf{x}^2}$ of the infinite cylinder apply to the case of point between the cutting planes of the finite cylinder. The intuition for a point in the extension is that the gradient is constant in the x and y directions and thus the Hessian is zero. For a \mathbf{x} outside the cylinder extension the nearest point lies on a cylinder edge. The gradient lies in the line connecting \mathbf{x} and that point. The final picture is as following.

A.1.5.1 Between cutting planes and outside the tube

Section A.1.4 applies to this case as well.

A.1.5.2 Between cutting planes and inside the tube and closer to the side than to a lid

Section A.1.4 applies to this case, too.

A.1.5.3 Between cutting planes and closer to a lid than to the side

The following Section A.1.5.4 derives this case, too.

A.1.5.4 Inside tube extension (outside cutting planes)

Let $\mathbf{b} := \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}$. Then following apply:

$$\begin{aligned}
 d &= s(\|\mathbf{b}\| - \frac{1}{2}h) \\
 \frac{\partial d}{\partial \mathbf{x}} &= \frac{\partial}{\partial \mathbf{x}} \left(s(\|\mathbf{b}\| - \frac{1}{2}h) \right) = s \left(\frac{\partial \|\mathbf{b}\|}{\partial \mathbf{x}} - \frac{\partial}{\partial \mathbf{x}} \left(\frac{1}{2}h \right) \right) \\
 &= s \frac{\partial \sqrt{\mathbf{b}^T \mathbf{b}}}{\partial \mathbf{x}} = s \frac{\partial}{\partial \mathbf{x}} \left(\sqrt{\langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle^2 \mathbf{z}^T \mathbf{z}} \right) \\
 &= s \sqrt{\mathbf{z}^T \mathbf{z}} \frac{\partial}{\partial \mathbf{x}} \left(\sqrt{\langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle^2} \right) = s \|\mathbf{z}\| \frac{\partial}{\partial \mathbf{x}} \left(\sqrt{\langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle^2} \right).
 \end{aligned}$$

Since \mathbf{z} gives only the direction, in our convention it is unit, i.e. $\|\mathbf{z}\| = 1$. The inner product $\langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle$ can be negative. Let $p := \text{sgn} \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle$, then

$$\begin{aligned}
 \frac{\partial d}{\partial \mathbf{x}} &= sp \frac{\partial \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle}{\partial \mathbf{x}} = sp \frac{\partial \langle \mathbf{x}, \mathbf{z} \rangle}{\partial \mathbf{x}} = sp \frac{\partial \mathbf{x}^T \mathbf{z}}{\partial \mathbf{x}} \\
 &= sp \mathbf{z}.
 \end{aligned}$$

Note that the norm of \mathbf{b} can be written as

$$\begin{aligned}
 \|\mathbf{b}\| &= \sqrt{\mathbf{b}^T \mathbf{b}} = \sqrt{\langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle^2 \mathbf{z}^T \mathbf{z}} = \sqrt{\langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle^2} \\
 &= p \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle = p \langle \mathbf{x} - \mathbf{c}, \mathbf{z} \rangle \mathbf{z}^T \mathbf{z} = p \mathbf{b}^T \mathbf{z}.
 \end{aligned}$$

So, $p\mathbf{z} = \frac{\mathbf{b}}{\|\mathbf{b}\|}$ and finally

$$\frac{\partial d}{\partial \mathbf{x}} = s \frac{\mathbf{b}}{\|\mathbf{b}\|}.$$

For the Hessian, the result is:

$$\frac{\partial^2 d}{\partial \mathbf{x}^2} = \frac{\partial}{\partial \mathbf{x}} \left(\frac{\partial d}{\partial \mathbf{x}} \right) = \frac{\partial}{\partial \mathbf{x}} \left(s \frac{\mathbf{b}}{\|\mathbf{b}\|} \right) = 0,$$

i.e. no change in gradient when varying \mathbf{x} infinitesimally – always pointing orthogonal to the planes (zero curvature).

A.1.5.5 Outside extension, outside cutting planes

Let $\mathbf{v} := \frac{\mathbf{a}}{\|\mathbf{a}\|} (\|\mathbf{a}\| - r) + \frac{\mathbf{b}}{\|\mathbf{b}\|} (\|\mathbf{b}\| - \frac{1}{2}h)$. (Intuition: \mathbf{v} is the vector from the nearest edge (lid meeting wall) point to \mathbf{x} .) Then,

$$d = s \|\mathbf{v}\|.$$

By the result from Sections A.1.4 and A.1.5.4 for the first derivative of the two terms, for the gradient we have:

$$\frac{\partial d}{\partial \mathbf{x}} = s \frac{\mathbf{v}}{\|\mathbf{v}\|}.$$

Now we derive once more to obtain the Hessian in terms of $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}$.

$$\begin{aligned}
 \frac{\partial^2 d}{\partial \mathbf{x}^2} &= s \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{v}}{\|\mathbf{v}\|} \right) = s \frac{1}{\|\mathbf{v}\|^2} \left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}} \|\mathbf{v}\| - \mathbf{v} \left(\frac{\partial \|\mathbf{v}\|}{\partial \mathbf{x}} \right)^T \right) \\
 &= s \frac{1}{\|\mathbf{v}\|^2} \left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}} \|\mathbf{v}\| - \frac{1}{2\|\mathbf{v}\|} \mathbf{v} \left(\frac{\partial \mathbf{v}^T \mathbf{v}}{\partial \mathbf{x}} \right)^T \right) \\
 &= s \frac{1}{\|\mathbf{v}\|^2} \left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}} \|\mathbf{v}\| - \frac{1}{2\|\mathbf{v}\|} 2\mathbf{v} \left(\mathbf{v} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^T \right) \\
 &= s \frac{1}{\|\mathbf{v}\|} \left(\frac{\partial \mathbf{v}}{\partial \mathbf{x}} - \frac{\mathbf{v} \mathbf{v}^T}{\|\mathbf{v}\|^2} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right). \tag{A.1}
 \end{aligned}$$

The derivative of \mathbf{v} required:

$$\begin{aligned}
 \frac{\partial \mathbf{v}}{\partial \mathbf{x}} &= \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{a}}{\|\mathbf{a}\|} (\|\mathbf{a}\| - r) + \frac{\mathbf{b}}{\|\mathbf{b}\|} (\|\mathbf{b}\| - \frac{1}{2}h) \right) \\
 &= \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{a}}{\|\mathbf{a}\|} (\|\mathbf{a}\| - r) \right) + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} (\|\mathbf{b}\| - \frac{1}{2}h) \right) \\
 &= \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{a}}{\|\mathbf{a}\|} \right) (\|\mathbf{a}\| - r) + \frac{\mathbf{a}}{\|\mathbf{a}\|} \frac{\partial}{\partial \mathbf{x}} (\|\mathbf{a}\| - r) \\
 &\quad + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\mathbf{b}}{\|\mathbf{b}\|} \frac{\partial}{\partial \mathbf{x}} \left(\|\mathbf{b}\| - \frac{1}{2}h \right) \\
 &= \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{a}}{\|\mathbf{a}\|} \right) (\|\mathbf{a}\| - r) + \frac{\mathbf{a}}{\|\mathbf{a}\|} \frac{\partial \|\mathbf{a}\|}{\partial \mathbf{x}} \\
 &\quad + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\mathbf{b}}{\|\mathbf{b}\|} \frac{\partial \|\mathbf{b}\|}{\partial \mathbf{x}}.
 \end{aligned}$$

Noting that a term is a reminiscent of the Hesse of the infinite cylinder up to a scaling,

$$C_{\infty}^H := \frac{1}{\|\mathbf{a}\|} \left(\mathbf{I}_3 - \mathbf{z} \mathbf{z}^T - \frac{\mathbf{a} \mathbf{a}^T}{\|\mathbf{a}\|^2} \right),$$

makes the notation somewhat more compact

$$\begin{aligned}
&= C_\infty^H(\|\mathbf{a}\| - r) \\
&\quad + \frac{\mathbf{a}}{\|\mathbf{a}\|} \frac{\partial \|\mathbf{a}\|}{\partial \mathbf{x}} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\mathbf{b}}{\|\mathbf{b}\|} \frac{\partial \|\mathbf{b}\|}{\partial \mathbf{x}} \\
&= C_\infty^H(\|\mathbf{a}\| - r) \\
&\quad + \frac{\mathbf{a}}{\|\mathbf{a}\|} \frac{\mathbf{a}}{\|\mathbf{a}\|} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\mathbf{b}}{\|\mathbf{b}\|} \frac{\partial \|\mathbf{b}\|}{\partial \mathbf{x}} \\
&= C_\infty^H(\|\mathbf{a}\| - r) + \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\mathbf{b}}{\|\mathbf{b}\|} \frac{\partial \|\mathbf{b}\|}{\partial \mathbf{x}} \\
&= C_\infty^H(\|\mathbf{a}\| - r) + \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\mathbf{b}}{\|\mathbf{b}\|} \mathbf{z} \|\mathbf{z}\| \\
&= C_\infty^H(\|\mathbf{a}\| - r) + \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \frac{\|\mathbf{z}\| \mathbf{b}\mathbf{z}^T}{\|\mathbf{b}\|} \\
&= C_\infty^H(\|\mathbf{a}\| - r) + \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{b}}{\|\mathbf{b}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \mathbf{z}\mathbf{z}^T \\
&= C_\infty^H(\|\mathbf{a}\| - r) + \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} + \frac{\partial}{\partial \mathbf{x}} \left(\frac{\mathbf{z}}{\|\mathbf{z}\|} \right) (\|\mathbf{b}\| - \frac{1}{2}h) + \mathbf{z}\mathbf{z}^T.
\end{aligned}$$

Since $\mathbf{b} \parallel \mathbf{z}$ and $\frac{\mathbf{b}}{\|\mathbf{b}\|}$ is the unit vector along \mathbf{z} direction (however, note that it may be $\pm \mathbf{z}$), finally

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \frac{\|\mathbf{a}\| - r}{\|\mathbf{a}\|} \left(\mathbf{I}_3 - \mathbf{z}\mathbf{z}^T - \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} \right) + \frac{\mathbf{a}\mathbf{a}^T}{\|\mathbf{a}\|^2} + \mathbf{z}\mathbf{z}^T.$$

Substituting this in (A.1) makes the Hesse complete.

A.2 Gaussian process derivatives

In order to incorporate the gradient observations in the covariance vector \mathbf{k} and the Gram matrix \mathbf{K} we need the derivative of the covariance function, as pointed out by Rasmussen and Williams (2006, p.191).

A.2.1 Covariances

This is a list of the covariance functions between function values, derivative values and mixed, which we define later for our particular GPs. They go between:

- function values at \mathbf{x}_a and \mathbf{x}_b : $k_{f,f}(\mathbf{x}_a, \mathbf{x}_b)$,
- function value at \mathbf{x}_a and derivative value at \mathbf{x}_b : $k_{f,d}(\mathbf{x}_a, \mathbf{x}_b)$,
- derivative value at \mathbf{x}_a and function value at \mathbf{x}_b : $k_{d,f}(\mathbf{x}_a, \mathbf{x}_b)$,
- derivative values at \mathbf{x}_a and \mathbf{x}_b : $k_{d,d}(\mathbf{x}_a, \mathbf{x}_b)$,
- second derivative at \mathbf{x}_a and function at \mathbf{x}_b : $k_{dd,f}(\mathbf{x}_a, \mathbf{x}_b)$,
- second derivative at \mathbf{x}_a and derivative at \mathbf{x}_b : $k_{dd,d}(\mathbf{x}_a, \mathbf{x}_b)$.

To not clutter the notation even more we take for granted the components in which the derivatives are taken, i.e. instead of writing $k_{d_{x_2}d_{x_3},f}(\mathbf{x}_a, \mathbf{y}_a)$, we stay at $k_{dd,f}(\mathbf{x}_a, \mathbf{x}_b)$ unless it is confusing to do so.

A.2.2 Observations, Gramian, covariance vector

We have \tilde{n} function value observations and \dot{n} derivative observations and $\bar{n} := \tilde{n} + \dot{n}$. The Gram matrix has

- $k_{f,f}(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j < \tilde{n}$,
- $k_{f,d}(\mathbf{x}_i, \mathbf{x}_j)$ for $i < \tilde{n}, j > \tilde{n}$ and

- $k_{d,d}(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j > \tilde{n}$.

The vector of the covariances of a point \mathbf{x} with all observations is

$$(k_{f,f}(\mathbf{x}, \mathbf{x}_1), \dots, k_{f,d}(\mathbf{x}, \mathbf{x}_{\tilde{n}}))^T =: \mathbf{k}.$$

The Gaussian process posterior mean is given by $f(\mathbf{x}) = \mathbf{k}\mathbf{G}^{-1}\mathbf{y}$.

A.2.3 Gradient

Taking the gradient of the GP posterior mean yields

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_i} \right)_i = \left(\frac{\partial \mathbf{k}\mathbf{G}^{-1}\mathbf{y}}{\partial x_i} \right)_i.$$

Since $\mathbf{G}^{-1}\mathbf{y}$ is constant w.r.t. any x_i ,

$$\nabla f(\mathbf{x}) = \left(\frac{\partial \mathbf{k}}{\partial x_i} \right)_i \mathbf{G}^{-1}\mathbf{y}.$$

Note that $\mathbf{G}^{-1}\mathbf{y}$ is a \tilde{n} -vector, $\nabla f(\mathbf{x})$ is a d -vector, $\left(\frac{\partial \mathbf{k}}{\partial x_i} \right)_i$ is $d \times \tilde{n}$. We use $\dot{\mathbf{k}}$ to denote the vector which contains the derivatives of the covariances of a point \mathbf{x} with all observations:

$$\dot{\mathbf{k}} := (k_{d,f}(\mathbf{x}, \mathbf{x}_1), \dots, k_{d,f}(\mathbf{x}, \mathbf{x}_{\tilde{n}}), \dots, k_{d,d}(\mathbf{x}, \mathbf{x}_{\tilde{n}}))^T.$$

Note: derivatives are taken w.r.t. \mathbf{x} , i.e. \mathbf{x}_i considered fixed.

A.2.4 Hessian

The Hessian of a GP posterior mean is defined as

$$\nabla \nabla f(\mathbf{x}) = \left(\frac{\partial \nabla f(\mathbf{x})}{\partial x_i} \right)_i = \left(\frac{\partial^2 \mathbf{k}\mathbf{G}^{-1}\mathbf{y}}{\partial x_i \partial x_j} \right)_{i,j}.$$

Again, $\mathbf{G}^{-1}\mathbf{y}$ is constant w.r.t \mathbf{x} and eventually

$$\nabla\nabla f(\mathbf{x}) = \left(\frac{\partial^2 \mathbf{k}}{\partial x_i \partial x_j} \right)_{i,j} \mathbf{G}^{-1}\mathbf{y}.$$

Note that $\left(\frac{\partial^2 \mathbf{k}}{\partial x_i \partial x_j} \right)_{i,j}$ is $d \times d \times \bar{n}$, $\nabla\nabla f(\mathbf{x})$ is $d \times d$. We use $\ddot{\mathbf{k}}$ to denote the vector which contains the second derivatives of the covariances of a point \mathbf{x} with all observations:

$$\ddot{\mathbf{k}} := (k_{dd,f}(\mathbf{x}, \mathbf{x}_1), \dots, k_{dd,f}(\mathbf{x}, \mathbf{x}_{\bar{n}}), \dots, k_{dd,d}(\mathbf{x}, \mathbf{x}_{\bar{n}}))^T.$$

Algorithm 4 compute Hessian

```

in:  $\mathbf{x} \in \mathbb{R}^d$ 
out:  $\mathbf{H} \in \mathbb{R}^{d \times d}$ 
given  $\mathbf{G}^{-1}\mathbf{y} \in \mathbb{R}^{\bar{n}}$                                 ▷ inverse Gram and response vector
new  $\boldsymbol{\kappa} \in \mathbb{R}^{d \times d \times \bar{n}}$ 
for  $n$  in  $1..\tilde{n}$  do
  for  $i, j$  in  $1..d$  do
     $\boldsymbol{\kappa}_{n,i,j} = k_{d x_i d x_j, f}(\mathbf{x}, \mathbf{x}_n)$ 
  end for
end for
for  $n$  in  $1..\hat{n}$  do
  for  $i, j$  in  $1..d$  do
     $\boldsymbol{\kappa}_{n+\bar{n},i,j} = k_{d x_i d x_j, d x_n}(\mathbf{x}, \mathbf{x}_n)$     ▷  $x_n$ : observed component in  $\mathbf{x}_n$ 
  end for
end for
 $H = \boldsymbol{\kappa} \mathbf{G}^{-1}\mathbf{y}$ 

```

A.2.5 Definition of covariances

Using notation: $\mathbf{x}_i, \mathbf{x}_j$ points (vectors); x_j (scalar), the component in which we take the derivative; $\frac{\partial f(\mathbf{x}_j)}{\partial x_d}$ the observed derivative in component x_d at

Algorithm 5 compute Hessian

```

in:  $\mathbf{x} \in \mathbb{R}^d$ 
out:  $\mathbf{H} \in \mathbb{R}^{d \times d}$ 
given  $\mathbf{G}^{-1} \mathbf{y} \in \mathbb{R}^n$                                 ▷ inverse Gram and response vector
new  $\kappa_{\mathbf{n}} \in \mathbb{R}^{d \times d}$ 
for  $n$  in  $1..\tilde{n}$  do
  for  $i, j$  in  $1..d$  do
     $\kappa_{n\tilde{i}, j} = k_{d_{x_i}, d_{x_j}, f}(\mathbf{x}, \mathbf{x}_{\mathbf{n}})$ 
  end for
   $H+ = \kappa_n(\mathbf{G}^{-1} \mathbf{y})_n$ 
end for
for  $n$  in  $1..\hat{n}$  do
  for  $i, j$  in  $1..d$  do
     $\kappa_{n+\tilde{n}, i, j} = k_{d_{x_i}, d_{x_j}, d_{x_n}}(\mathbf{x}, \mathbf{x}_{\mathbf{n}})$     ▷  $x_n$ : observed component in  $\mathbf{x}_{\mathbf{n}}$ 
  end for
   $H+ = \kappa_n(\mathbf{G}^{-1} \mathbf{y})_{n+\tilde{n}}$ 
end for

```

derivative-observation point \mathbf{x}_j . \mathbf{x}_{i_j} the j th component of vector \mathbf{x}_i , δ_{ij} being Kronecker's delta, we have:

- plain squared exponential covariance:

$$\sigma_p^2 e^{\frac{1}{2} \sigma_w^{-2} (\mathbf{x}_1 - \mathbf{x}_2)^T (\mathbf{x}_1 - \mathbf{x}_2)} =: k_{f,f}(\mathbf{x}_1, \mathbf{x}_2)$$

(we omit the constant σ_p in the following for the sake of clarity)

- mixed covariances. Note that $k_{f,d}(\cdot, \cdot)$ is not symmetric.

$$k_{f,d}(\mathbf{a}, \mathbf{b}) = k \left(f(\mathbf{a}), \frac{\partial f(\mathbf{b})}{\partial x_e} \right) = \frac{\partial k(\mathbf{a}, \mathbf{b})}{\partial x_e} = \sigma_w^{-2} (\mathbf{a}_e - \mathbf{b}_e) k_{f,f}(\mathbf{a}, \mathbf{b}),$$

$$k_{d_{x_e}, f}(\mathbf{a}, \mathbf{b}) = k_{f, d_{x_e}}(\mathbf{b}, \mathbf{a}),$$

$$k_{f, d_{x_e}}(\mathbf{b}, \mathbf{a}) = -k_{f, d_{x_e}}(\mathbf{a}, \mathbf{b}) = \sigma_w^{-2} (\mathbf{b}_e - \mathbf{a}_e) k_{f,f}(\mathbf{a}, \mathbf{b}),$$

- covariances between derivatives. $k_{d,d}(\cdot, \cdot)$ is symmetric. Note that the one derivative is taken w.r.t. the one arg, the other with respect to the other.

$$\begin{aligned}
 k_{d,d}(\mathbf{a}, \mathbf{b}) &= k \left(\frac{\partial f(\mathbf{a})}{\partial x_e}, \frac{\partial f(\mathbf{b})}{\partial x_s} \right) = \frac{\partial^2 k(\mathbf{a}, \mathbf{b})}{\partial x_e \partial x_s} \\
 &= \frac{\partial}{\partial x_e} (\sigma_w^{-2} (\mathbf{a}_s - \mathbf{b}_s) k(\mathbf{a}, \mathbf{b})) \\
 &= \frac{\partial}{\partial x_e} (k_{f,d_{x_s}}(\mathbf{a}, \mathbf{b})) , \text{ think } \mathbf{b} \text{ const} \\
 &= \frac{\partial}{\partial x_s} (k_{d_{x_e},f}(\mathbf{a}, \mathbf{b})) , \text{ think } \mathbf{a} \text{ const}
 \end{aligned}$$

$$= \left\{ \begin{array}{l}
 x_e = x_s, \quad = \sigma_w^{-2} \left(\frac{\partial(\mathbf{a}_s - \mathbf{b}_s)}{\partial x_e} k(\mathbf{a}, \mathbf{b}) + \frac{\partial k(\mathbf{a}, \mathbf{b})}{\partial x_e} (\mathbf{a}_s - \mathbf{b}_s) \right) \\
 \quad = \sigma_w^{-2} (k(\mathbf{a}, \mathbf{b}) + \sigma_w^{-2} (\mathbf{b}_e - \mathbf{a}_e) k(\mathbf{a}, \mathbf{b}) (\mathbf{a}_s - \mathbf{b}_s)) \\
 \quad \text{Note } (\mathbf{b}_e - \mathbf{a}_e): \text{ we take 2nd derivative w.r.t. } a! \\
 \quad \text{next, we swap the sign. voila.} \\
 \quad = \sigma_w^{-2} (1 - \sigma_w^{-2} (\mathbf{a}_e - \mathbf{b}_e) (\mathbf{a}_s - \mathbf{b}_s)) k(\mathbf{a}, \mathbf{b}) \\
 \quad \text{swap and change sign} \\
 x_e \neq x_s, \quad = \sigma_w^{-2} (\mathbf{a}_s - \mathbf{b}_s) \frac{\partial k(\mathbf{a}, \mathbf{b})}{\partial x_e} \\
 \quad = \sigma_w^{-2} (\mathbf{a}_s - \mathbf{b}_s) \sigma_w^{-2} (\mathbf{b}_e - \mathbf{a}_e) k(\mathbf{a}, \mathbf{b}) \\
 \quad = -\sigma_w^{-2} \sigma_w^{-2} (\mathbf{a}_s - \mathbf{b}_s) (\mathbf{a}_e - \mathbf{b}_e) k(\mathbf{a}, \mathbf{b})
 \end{array} \right.$$

$$= \sigma_w^{-2} (\delta_{ij} - \sigma_w^{-2} (\mathbf{a}_e - \mathbf{b}_e) (\mathbf{a}_s - \mathbf{b}_s)) k(\mathbf{a}, \mathbf{b}).$$

- covariances between second derivative and function value

$$\begin{aligned}
 k_{dd,f}(\mathbf{a}, \mathbf{b}) &= \frac{\partial k_{d,f}(\mathbf{a}, \mathbf{b})}{\partial x_e} = -\frac{\partial k_{f,d}(\mathbf{a}, \mathbf{b})}{\partial x_e}, \text{ deriv. w.r.t. } a \\
 &= -k_{d,d}(\mathbf{a}, \mathbf{b}) \\
 &= -\sigma_w^{-2} (\delta_{ij} - \sigma_w^{-2}(\mathbf{a}_e - \mathbf{b}_e)(\mathbf{a}_s - \mathbf{b}_s)) k(\mathbf{a}, \mathbf{b}).
 \end{aligned}$$

- covariances between second derivative and derivative. (In our use: second derivative of covariance between function value and observed derivative value.)

$$\begin{aligned}
 k_{dd,d}(\mathbf{a}, \mathbf{b}) &= k_{d_{x_e} d_{x_l} d_{x_s}}(\mathbf{a}, \mathbf{b}) = \frac{\partial k_{d,d}(\mathbf{a}, \mathbf{b})}{\partial x_e} \\
 &= \frac{\partial}{\partial x_e} (\sigma_w^{-2} (\delta_{ls} - \sigma_w^{-2}(\mathbf{a}_l - \mathbf{b}_l)(\mathbf{a}_s - \mathbf{b}_s)) k(\mathbf{a}, \mathbf{b})) \\
 &= \sigma_w^{-2} \left(\delta_{ls} \frac{\partial k(\mathbf{a}, \mathbf{b})}{\partial x_e} - \sigma_w^{-2} \frac{\partial}{\partial x_e} ((\mathbf{a}_l - \mathbf{b}_l)(\mathbf{a}_s - \mathbf{b}_s)k(\mathbf{a}, \mathbf{b})) \right), x_e \text{ w.r.t } a \\
 &= \sigma_w^{-2} \left(\delta_{ls} k_{d,f}(\mathbf{a}, \mathbf{b}) - \sigma_w^{-2} \frac{\partial}{\partial x_e} ((\mathbf{a}_l - \mathbf{b}_l)(\mathbf{a}_s - \mathbf{b}_s)k(\mathbf{a}, \mathbf{b})) \right), x_e \text{ w.r.t } a.
 \end{aligned}$$

Using $(abc)' = a'bc + a(bc)' = a'bc + ab'c + abc'$:

$$\begin{aligned}
 &= \sigma_w^{-2} (\delta_{ls} k_{d,f}(\mathbf{a}, \mathbf{b}) - \sigma_w^{-2} \frac{\partial(\mathbf{a}_l - \mathbf{b}_l)}{\partial x_e} (\mathbf{a}_s - \mathbf{b}_s) k(\mathbf{a}, \mathbf{b}) \\
 &\quad - \sigma_w^{-2} (\mathbf{a}_l - \mathbf{b}_l) \frac{\partial(\mathbf{a}_s - \mathbf{b}_s)}{\partial x_e} k(\mathbf{a}, \mathbf{b}) \\
 &\quad - \sigma_w^{-2} (\mathbf{a}_l - \mathbf{b}_l)(\mathbf{a}_s - \mathbf{b}_s) k_{d,f}(\mathbf{a}, \mathbf{b})).
 \end{aligned}$$

With $\frac{\partial(\mathbf{a}_l - \mathbf{b}_l)}{\partial x_e} = \delta_{el}$ and $k_{d,f}(\mathbf{a}, \mathbf{b}) = -k_{f,d}(\mathbf{a}, \mathbf{b})$ and knowing $k_{f,d}(\mathbf{a}, \mathbf{b})$

from above

$$\begin{aligned}
&= \sigma_w^{-2}(\delta_{ls}k_{d,f}(\mathbf{a}, \mathbf{b}) - \sigma_w^{-2}\delta_{el}(\mathbf{a}_s - \mathbf{b}_s)k(\mathbf{a}, \mathbf{b}) \\
&\quad - \sigma_w^{-2}\delta_{es}(\mathbf{a}_l - \mathbf{b}_l)k(\mathbf{a}, \mathbf{b}) \\
&\quad + \sigma_w^{-2}(\mathbf{a}_l - \mathbf{b}_l)(\mathbf{a}_s - \mathbf{b}_s)\sigma_w^{-2}(\mathbf{a}_e - \mathbf{b}_e)k(\mathbf{a}, \mathbf{b})).
\end{aligned}$$

Finally, after resolving the other $k_{d,f}(\mathbf{a}, \mathbf{b})$, taking away common $k(\mathbf{a}, \mathbf{b})$ and σ_w^{-2} and using notation $d_l := (\mathbf{a}_l - \mathbf{b}_l)$ gives the more readable and compact:

$$= \sigma_w^{-2}\sigma_w^{-2}(-\delta_{ls}d_e - \delta_{el}d_s - \delta_{es}d_l + \sigma_w^{-2}d_l d_s d_e) k(\mathbf{a}, \mathbf{b}).$$

A.3 Jacobian of potential field align task variable

PotentialFieldAlignTaskVariable, $\mathbf{y} \in \mathbb{R}^d$ is a motion feature which, with the proper reference, aligns the z -axes of d shapes to the gradient of a potential function. Let \mathbf{z}_i be the z -vector of the i -th shape. Assume the gradient and Hessian of the potential field are available (as derived in the above sections). Let \mathbf{q} be the joint configuration point in configuration space; for each shape, $\mathbf{z}_i := \phi^z(\mathbf{q})$ and $\mathbf{x}_i := \phi(\mathbf{q})$: the current orientation and position of the shape computed from forward kinematics; we also know the Jacobians: $\frac{\partial \phi^z(\mathbf{q})}{\partial \mathbf{q}} =: \mathbf{J}^z(\mathbf{q}) =: \mathbf{J}^z$ and $\frac{\partial \phi(\mathbf{q})}{\partial \mathbf{q}} =: \mathbf{J}(\mathbf{q}) =: \mathbf{J}$; $\psi = \psi(\mathbf{x})$ is the potential field value and its gradient (w.r.t. \mathbf{x}) is $\nabla \psi(\mathbf{x})$. Then the value of \mathbf{y} , i.e. the update rule is given, for each time step, as $\mathbf{y}_i = \langle \nabla \psi, \mathbf{z}_i \rangle \forall i \in 1..d$. The Jacobian

of the variable is

$$\begin{aligned}
 \frac{\partial \mathbf{y}_i}{\partial q} &= \frac{\partial}{\partial \mathbf{q}} (\langle \nabla \psi, \mathbf{z}_i \rangle) = \frac{\partial}{\partial \mathbf{q}} (\langle \nabla \psi(\mathbf{x}_i), \mathbf{z}_i \rangle) \\
 &= \frac{\partial}{\partial \mathbf{q}} ((\nabla \psi(\mathbf{x}_i))^T \mathbf{z}_i) = \frac{\partial \nabla \psi(\mathbf{x}_i)^T}{\partial \mathbf{q}} \mathbf{z}_i + \nabla \psi(\mathbf{x}_i)^T \frac{\partial \mathbf{z}_i}{\partial \mathbf{q}} \\
 &= \frac{\partial \nabla \psi(\mathbf{x}_i)^T}{\partial \mathbf{q}} \mathbf{z}_i + \frac{\partial \psi(\mathbf{x}_i)^T}{\partial \mathbf{x}_i} \mathbf{J}^z(\mathbf{q}) \\
 &= \left(\frac{\partial^2 \psi(\mathbf{x}_i)}{\partial \mathbf{x}^2} \frac{\partial \mathbf{x}}{\partial \mathbf{q}} \right)^T \mathbf{z}_i + \frac{\partial \psi(\mathbf{x}_i)^T}{\partial \mathbf{x}_i} \mathbf{J}^z(\mathbf{q}) \\
 &= \left(\frac{\partial \mathbf{x}_i^T}{\partial \mathbf{q}} \frac{\partial^2 \psi(\mathbf{x}_i)}{\partial \mathbf{x}^2} \right)^T \mathbf{z}_i + \frac{\partial \psi(\mathbf{x}_i)^T}{\partial \mathbf{x}_i} \mathbf{J}^z(\mathbf{q}) \\
 &= \left(\frac{\partial \phi(\mathbf{q})^T}{\partial \mathbf{q}} \frac{\partial^2 \psi(\mathbf{x}_i)}{\partial \mathbf{x}^2} \right)^T \mathbf{z}_i + \frac{\partial \psi(\mathbf{x}_i)^T}{\partial \mathbf{x}_i} \mathbf{J}^z(\mathbf{q}).
 \end{aligned}$$

With $\mathbf{G} := \frac{\partial \psi(\mathbf{x}_i)}{\partial \mathbf{x}_i}$ and $\mathbf{H} := \frac{\partial^2 \psi(\mathbf{x}_i)}{\partial \mathbf{x}^2}$,

$$\frac{\partial \mathbf{y}_i}{\partial q} = (\mathbf{J}^T \mathbf{H}^T) \mathbf{z}_i + \mathbf{G}^T \mathbf{J}^z,$$

and since $\mathbf{H} = \mathbf{H}^T$, finally

$$\frac{\partial \mathbf{y}_i}{\partial q} = (\mathbf{J}^T \mathbf{H}) \mathbf{z}_i + \mathbf{G}^T \mathbf{J}^z.$$

As target, reference value, we set here vector $-\mathbf{1}$ to orient the body along the negative gradient: $y^* = -\mathbf{1}$.

Appendix B

On robot's self according to Avicenna

The title is misleading; we do not really intend to search for the self. Nevertheless, the thought experiment proposed by Avicenna provides interesting structure for reasoning about the capabilities of a robot.

B.1 Floating man

Centuries before René Descartes' reasoning on dualism, the Arabic philosopher Avicenna, influenced by the ancient Greek philosophers, believes that the mind is distinct from the body, and our mind is what makes us up, our self. To illustrate this, he comes up with a thought experiment in which he detaches a man from his body parts and senses. Avicenna argues that none of these removals impact the man's self-consciousness. This inspires the following thought experiment with a robot, to illustrate that only having both sensing and actuation in the loop together makes it even possible to think about nontrivial actions towards a goal.

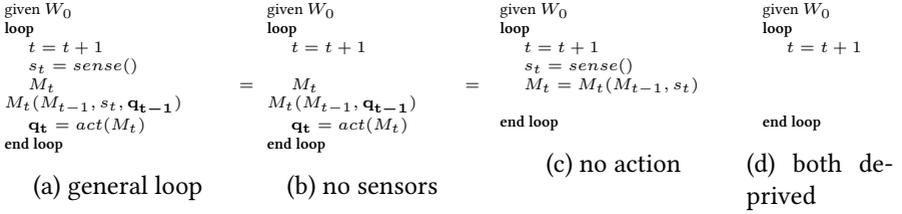


Figure B.1: The life loop of a robot with and without sensors or/and actuators.

B.2 Floating robot

Consider a general robot with certain kinematic structure and sensors. Let the robot have N DOF, n of which it can actuate, $|\mathbf{q}| = n$. In addition, a number of sensors which deliver readings s_t in the t -th time step, and prior world model M_t which changes over time depending on the sensors. (Note: The assumed outcome of an action is part of the world model). Furthermore, it can sense and act according to its belief, which changes the world and the world model.

The ‘life’ of the robot is governed by the general loop outlined in Figure B.1a. Let us consider three distinct parts – sensors, actuators, and world model (belief, M).

B.2.1 Life in sensory darkness

If we remove the sensing ability, the robot ends up acting blindly only according to its world model without getting any feedback about what it believes it changes in the world (Figure B.1b). This rules out informed interaction and thus any kind of adaptation to changing goals or environments. Since it does not perceive any change, it is questionable whether it makes difference for the agent how it acts anyway. This is probably the configuration people have in mind when insulting someone as ‘robot’: blindly acting, following one’s pre-programmed goal and moves.

B.2.2 Life as passive witness

If we remove the kinematic chain, or even only the controllability of its DOF, but retain some abstract ability to sense, the robot is not capable to actively change its environment. So it can only update a model of the world but without taking advantage of this, not to speak of actively shaping its sensing: it is merely an observer of a random fraction of the world. The result is the loop in Figure B.1c. The usefulness of the knowledge which can be gained by mere observation is questionable in two aspects: it cannot be taken advantage of in any way – apart from having the knowledge just for the sake of it; and it is exposed to the randomness of the fraction of the reality which happens to be perceived.

B.2.3 Life with eternal prior

Removing both sensing and actuation, we end up having an a priori model forever with no chance to observe, reason about, understand or change the world, Figure B.1d.

At this point, we might argue that the self of the robot is the given knowledge about the world. For us, it remains impossible to carry out the thought experiment, though, since we are outside the loop. We are external to the robot existence and only from inside one can answer the question whether one retains one's self with losing sensing or actuation ability.

Note that if we are to be strict in suppressing physical activity, we need to plug out the processor running the loop, since, for instance, its heat is a communication channel to the outside world. But that would render the whole experiment pointless.

B.2.4 Sensory motor control

Our point here is that only having *both sensing and actuation in the loop together* provides a robot capable of updating its model about the world and acting upon this model to achieve certain goal.

B.2.5 Abstract actuation

The above setting assumes a kinematic robot but it can be translated to one which actuators impact the world in different way – e.g. by sending e-mails. However, the restriction to a set of capabilities remains.

Suppose we remove the body of the robot but allow it to manipulate objects: we give it actuators which are abstract in the sense that they do not need to obey any kinematic or other limitations. We believe, that the embodiment (the set of possible ways to impact the world) which poses the limitations accelerates learning, rather than harm it. It provides a relevance measure which directs exploration parts of the model which are relevant to the capabilities. Thus, making a robot capable of unconstrained manipulation makes the problem more difficult, not simpler.

Literature

- N. Amenta, S. Choi, and R. K. Kolluri. The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–266. ACM, 2001.
- V. Andersen, H. Aanæs, and J. A. Bærentzen. Surfel based geometry reconstruction. 2010.
- I. Asimov. *I, robot*. HarperCollins UK, 1950.
- A. H. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer graphics and Applications*, 1(1):11–23, 1981.
- T. Bayes and R. Price. An Essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a Letter to John Canton, AMFRS. *Philosophical Transactions of the Royal Society of London* 53, pages 370–418, 1763.
- Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.
- D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner. Grasp planning in complex scenes. In *Proceedings of the 7th International Conference on Humanoid Robots*, pages 42–48. IEEE, 2007.

- N. A. Bernstein. The co-ordination and regulation of movements. 1967.
- P. J. Besl and R. C. Jain. Three-dimensional object recognition. *ACM Computing Surveys (CSUR)*, 17(1):75–145, 1985.
- A. Bicchi. On the closure properties of robotic grasping. *The International Journal of Robotics Research*, 14(4):319–334, 1995.
- A. Bicchi and V. Kumar. Robotic grasping and contact: A review. In *Proceedings of the International Conference on Robotics and Automation*, volume 1, pages 348–353. IEEE, 2000.
- A. Bierbaum and M. Rambow. Grasp affordances from multi-fingered tactile exploration using dynamic potential fields. In *Proceedings of the 9th Conference on Humanoid Robots*, pages 168–174. IEEE, 2009.
- A. Bierbaum, M. Rambow, T. Asfour, and R. Dillmann. A potential field approach to dexterous tactile exploration of unknown objects. In *Proceedings of the 8th International Conference on Humanoid Robots*, pages 360–366. IEEE, 2008.
- J. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics (TOG)*, 1(3):235–256, 1982.
- J. Bloomenthal. Bulge elimination in implicit surface blends. In *Implicit Surfaces*, volume 95, pages 7–20, 1995.
- J. Bohg, A. Morales, T. Asfour, and D. Kragic. Data-driven grasp synthesis—a survey. 2013.
- R. M. Bolle and B. C. Vemuri. On three-dimensional surface reconstruction methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):1–13, 1991.

- T. Bretl. Minimum-time optimal control of many robots that move in the same direction at different speeds. *IEEE Transactions on Robotics*, 28(2):351–363, 2012.
- R. A. Brooks. Elephants don't play chess. *Robotics and autonomous systems*, 6(1):3–15, 1990.
- E. Brown, N. Rodenberg, J. Amend, A. Mozeika, E. Steltz, M. R. Zakin, H. Lipson, and H. M. Jaeger. Universal robotic gripper based on the jamming of granular material. *Proceedings of the National Academy of Sciences*, 107(44):18809–18814, 2010.
- S. R. Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. <http://euclid.ucsd.edu/~sbuss/ResearchWeb/ikmethods/>, 2005.
- B. Calli, M. Wisse, and P. Jonker. Grasping of unknown objects via curvature maximization using active vision. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 995–1001. IEEE/RSJ, 2011.
- K. Čapek. *RUR (Rossum's Universal Robots): a play in three acts and an epilogue*. Humphrey Milford, 1928.
- S. Caselli, C. Magnanini, F. Zanichelli, and E. Caraffi. Efficient exploration and recognition of convex objects based on haptic perception. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3508–3513. IEEE, 1996.
- A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *AAAI*, volume 94, pages 1023–1028, 1994.

- G. Chapman, J. Cleese, T. Gilliam, E. Idle, T. Jones, M. Palin, C. Cleveland, and N. Innes. Royal society for putting things on top of other things. TV show: Live from the Grill-O-Mat; Monty Python's Flying Circus, October 1970.
- C. W. Chen, T. S. Huang, and M. Arrott. Modeling, analysis, and visualization of left ventricle shape and motion by hierarchical decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):342–356, 1994.
- E. V. Chernyaev. Marching cubes 33: Construction of topologically correct isosurfaces. *Institute for High Energy Physics, Moscow, Russia, Report CN/95-17*, 1995.
- S. Chiaverini, G. Oriolo, and I. D. Walker. Kinematically redundant manipulators. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, pages 245–268. Springer, 2008.
- V. N. Christopoulos and P. R. Schrater. Grasping objects with environmentally induced position uncertainty. *PLoS computational biology*, 5(10):e1000538, 2009.
- D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. Technical report, DTIC Document, 1995.
- B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996. ISBN 0897917464.
- Y. Das and W. Boerner. On radar target shape estimation using algorithms for reconstruction from projections. *IEEE Transactions on Antennas and Propagation*, 26(2):274–279, 1978.

- B. de Finetti. *Probabilità e induzione: Induction and probability*. Cooperativa Libreria Universitaria Editrice Bologna, 1993.
- R. Dechter and J. Pearl. Generalized best-first search strategies and the optimality of A*. *Journal of the ACM (JACM)*, 32(3):505–536, 1985.
- R. Deimel and O. Brock. A compliant hand based on a novel pneumatic actuator. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 01–07, 2013.
- R. Diankov, S. S. Srinivasa, D. Ferguson, and J. Kuffner. Manipulation planning with caging grasps. In *Proceedings of the 8th International Conference on Humanoid Robots*, pages 285–292. IEEE, 2008.
- S. Dragiev, M. Toussaint, and M. Gienger. Gaussian process implicit surface for shape estimation and grasping. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*. IEEE, 2011a.
- S. Dragiev, M. Toussaint, and M. Gienger. Re-grasping for Gaussian process implicit surface. In *Workshop on Autonomous Grasping at International Conference on Robotics and Automation (ICRA)*. IEEE, 2011b. URL <http://mailless.org/~stanio/publications/2011icra>.
- S. Dragiev, M. Toussaint, and M. Gienger. Tactile exploration and grasping unknown objects by actively learning Gaussian process implicit shape potentials. In *Robotics: science and systems conference, workshop on Active learning in robotics: exploration, curiosity, and interaction*. IEEE, 2013a.
- S. Dragiev, M. Toussaint, and M. Gienger. Uncertainty aware grasping and tactile exploration. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*. IEEE, 2013b. URL <http://mailless.org/~stanio/publications/2013icra>.

- A. D'Souza, S. Vijayakumar, and S. Schaal. Learning inverse kinematics. In *Proceedings of the International Conference on Intelligent Robots and Systems*, volume 1, pages 298–303. IEEE, 2001.
- J. L. Elman. Connectionist models of cognitive development: where next? *Trends in cognitive sciences*, 9(3):111–117, 2005.
- C. Eppner, G. Bartels, and O. Brock. A compliance-centric view of grasping. Technical report, Department of Computer Engineering and Microelectronics, Technische Universität Berlin, February 2012. URL http://www.robotics.tu-berlin.de/fileadmin/fg170/Publikationen_pdf/technical_report_sfa.pdf.
- R. Fabio. From point cloud to surface: the modeling and visualization problem. In *International Workshop on Visualization and Animation of Reality-based 3D Models*, volume 34, page 5, 2003.
- T. Feix, H.-B. Schmiedmayer, J. Romero, and D. Kragić. A comprehensive grasp taxonomy. In *Robotics: science and systems conference, Workshop on understanding the human hand for advancing robotic manipulation*, 2009.
- V. Gallese. Embodied simulation: From neurons to phenomenal experience. *Phenomenology and the cognitive sciences*, 4(1):23–48, 2005.
- J. J. Gibson. The theory of affordances. In R. E. Shaw and J. Bransford, editors, *Perceiving, acting, and knowing*. Lawrence Erlbaum Associate, 1977.
- J. J. Gibson. *The ecological approach to visual perception*. Routledge, 1986.
- S. F. F. Gibson and B. Mirtich. A survey of deformable modeling in computer graphics. Technical Report TR97-19, MERL - Mitsubishi Electric Research Laboratories, 201 Broadway, Cambridge, MA 02139, Nov. 1997.

- M. Gienger, M. Toussaint, and C. Goerick. Task maps in humanoid robot manipulation. In *Proceedings of the International Conference on Intelligent Robot and Systems (IROS)*, volume 9. IEEE/RSJ, 2008.
- M. Gienger, M. Toussaint, and C. Goerick. Whole-body motion planning – building blocks for intelligent systems. In *Motion Planning for Humanoid Robots*, pages 67–98. Springer, 2010.
- A. Girard, C. Rasmussen, J. Candela, and R. Murray-Smith. Gaussian process priors with uncertain inputs – application to multiple-step ahead time series forecasting. *Advances in Neural Information Processing Systems*, pages 545–552, 2003. ISSN 1049-5258.
- S. Glover. Separate visual representations in the planning and control of action. *Behavioral and Brain Sciences*, 27(01):3–24, 2004.
- C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof. Grasp planning via decomposition trees. In *Proceedings of the International Conference on Robotics and Automation*, pages 4679–4684. IEEE, 2007.
- M. A. Goodale, A. D. Milner, L. Jakobson, D. Carey, et al. A neurological dissociation between perceiving objects and grasping them. *Nature*, 349(6305):154–156, 1991.
- W. W. Hager. Updating the inverse of a matrix. *SIAM review*, 31(2):221–239, 1989.
- P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
- R. S. Hartenberg and J. Denavit. *Kinematic synthesis of linkages*. McGraw-Hill New York, 1964.

- R. Haschke, J. Steil, I. Steuwer, and H. Ritter. Task-oriented quality measures for dextrous grasping. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 689–694, 6 2005.
- T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition, 2009.
- C. Hernández Esteban and F. Schmitt. Silhouette and stereo fusion for 3D object modeling. *Computer Vision and Image Understanding*, 96(3):367–392, 2004.
- G. A. Hollinger, B. Englot, F. Hover, U. Mitra, and G. S. Sukhatme. Uncertainty-driven view planning for underwater inspection. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 4884–4891. IEEE, 2012.
- K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez. Task-driven tactile exploration. In *Robotics: Science and Systems Conference*. The MIT Press, 2010.
- K. Huebner and D. Kragic. Selection of robot pre-grasps using box-based shape approximation. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 1765–1770. IEEE, 2008.
- H. Intraub. Rethinking Scene Perception: A Multisource Model. *Psychology of learning and motivation*, pages 231–264, 2010. ISSN 0079-7421.
- N. Jetchev and M. Toussaint. Task space retrieval using inverse feedback control. In *Proceedings of the 28th International Conference on Machine Learning*, pages 449–456, 2011.
- N. Jetchev, T. Lang, and M. Toussaint. Learning grounded relational symbols from continuous data for abstract reasoning. In *Autonomous Learning*

- workshop at the International Conference on Robotics and Automation (ICRA)*, 2013.
- P. Jiménez. Survey on model-based manipulation planning of deformable objects. *Robotics and Computer-Integrated Manufacturing*, 28(2):154–163, 2012.
- R. Joshi and A. C. Sanderson. Shape matching from grasp using a minimal representation size criterion. In *Proceedings of the International Conference on Robotics and Automation*, pages 442–449. IEEE, 1993.
- D. Katz and O. Brock. Manipulating articulated objects with interactive perception. In *Proceedings of the International Conference on Robotics and Automation*, pages 272–277. IEEE, 2008.
- L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.
- O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90, 1986.
- O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, 3(1):43–53, 1987.
- P. Khosla and R. Volpe. Superquadric artificial potentials for obstacle avoidance and approach. In *Proceedings of the International Conference on Robotics and Automation*, pages 1778–1784. IEEE, 1988.
- R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International joint conference on artificial intelligence*, volume 14, pages 1137–1145, 1995.
- A. Kolmogorov. *Foundations of the theory of probability*. 1950.

- K. P. Körding and D. M. Wolpert. The loss function of sensorimotor learning. *Proceedings of the National Academy of Sciences of the United States of America*, 101(26):9839–9842, 2004.
- O. Kroemer, R. Detry, J. Piater, and J. Peters. Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems*, 2010. ISSN 0921-8890.
- J. J. Kuffner Jr and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings of the International Conference on Robotics and Automation*, volume 2, pages 995–1001. IEEE, 2000.
- T. Lang and M. Toussaint. Planning with noisy probabilistic relational rules. *Journal of Artificial Intelligence Research*, 39(1):1–49, 2010.
- P. Langley. *Elements of machine learning*. Morgan Kaufmann, 1996.
- P. S. Laplace. *Théorie analytique des probabilités*. Ve. Courcier, 1814.
- S. M. LaValle. Rapidly-exploring random trees a new tool for path planning. Technical Report TR 98-11, Iowa State University, Computer Science Dept., 1998.
- T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of graphics tools*, 8(2):1–15, 2003.
- Q. Li, C. Schürmann, R. Haschke, and H. Ritter. A control framework for tactile servoing. In *Robotics: Science and Systems conference*, 2013.
- A. Liegeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(12):868–871, 1977.

- M. Likhachev, G. J. Gordon, and S. Thrun. ARA*: Anytime A* with provable bounds on sub-optimality. In *Advances in Neural Information Processing Systems*, 2003.
- W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *ACM Siggraph Computer Graphics*, volume 21, pages 163–169. ACM, 1987.
- V. Lumelsky and A. Stepanov. Dynamic path planning for a mobile automaton with limited information on the environment. *IEEE Transactions on Automatic Control*, 31(11):1058–1063, 1986.
- M. T. Mason, S. S. Srinivasa, and A. S. Vazquez. Generality and simple hands. In *Robotics Research*, pages 345–361. Springer, 2011.
- D. J. MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.
- M. T. Mason. Mechanics and planning of manipulator pushing operations. *The International Journal of Robotics Research*, 5(3):53–71, 1986.
- Z. McCarthy and T. Bretl. Mechanics and manipulation of planar elastic kinematic chains. In *International Conference on Robotics and Automation (ICRA)*, pages 2798–2805. IEEE, 2012.
- D. Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, 1982.
- G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori. The iCub humanoid robot: an open platform for research in embodied cognition. In *Proceedings of the 8th workshop on performance metrics for intelligent systems*, pages 50–56. ACM, 2008.
- A. T. Miller and P. K. Allen. Graspit! A versatile simulator for robotic grasping. *Robotics & Automation Magazine, IEEE*, 11(4):110–122, 2004.

- A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen. Automatic grasp planning using shape primitives. In *Proceedings of the International Conference on Robotics and Automation*, volume 2, pages 1824–1829. IEEE, 2003.
- M. Minsky. A framework for representing knowledge. 1974.
- T. M. Mitchell. *The discipline of machine learning*. Carnegie Mellon University, School of Computer Science, Machine Learning Department, 2006.
- L. Montesano and M. Lopes. Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions. *Robotics and Autonomous Systems*, 60(3):452–462, 2012.
- H. Moravec. *Mind children*. Cambridge Univ Press, 1988.
- A. Murata, L. Fadiga, L. Fogassi, V. Gallese, V. Raos, and G. Rizzolatti. Object representation in the ventral premotor cortex (area F5) of the monkey. *Journal of neurophysiology*, 78(4):2226–2230, 1997.
- K.-R. Müller, S. Mika, G. Ratsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.
- Y. Nakamura and H. Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of dynamic systems, measurement, and control*, 108:163–171, 1986.
- J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757, 2008.
- Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H. Seidel. Multi-level partition of unity implicit. In *ACM SIGGRAPH 2005 Courses*, page 173. ACM, 2005.

- A. M. Okamura, N. Smaby, and M. R. Cutkosky. An overview of dexterous manipulation. In *Proceedings of the International Conference on Robotics and Automation*, volume 1, pages 255–262. IEEE, 2000.
- H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling. Learning symbolic models of stochastic domains. *Journal of Artificial Intelligence Research (JAIR)*, 29: 309–352, 2007.
- R. Pfeifer, M. Lungarella, O. Sporns, and Y. Kuniyoshi. On the information theoretic implications of embodiment – principles and methods. In M. Lungarella, F. Iida, J. Bongard, and R. Pfeifer, editors, *50 years of artificial intelligence: essays dedicated to the 50th anniversary of artificial intelligence*, pages 76–86. Springer, 2007.
- R. Pike. The text editor sam. *Software: Practice and Experience*, 17(11):813–845, 1987.
- R. Pike. Notes on programming in C. http://doc.cat-v.org/bell_labs/pikestyle, 1989. [Online; accessed 2014-01-29].
- F. Pokorný, J. Stork, and D. Kragic. Grasping objects with holes: a topological approach. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*. IEEE, 2013.
- D. Precup, R. S. Sutton, and S. Singh. Theoretical results on reinforcement learning with temporally abstract options. In *Machine Learning: ECML-98*, pages 382–393. Springer, 1998.
- N. Pugeault. Early cognitive vision: Feedback mechanisms for the disambiguation of early visual representation. 2008.
- M. Raibert, K. Blankespoor, G. Nelson, R. Playter, et al. Bigdog, the rough-terrain quadruped robot. In *Proceedings of the 17th IFAC World Congress*, pages 10823–10825, 2008.

- C. Rasmussen and C. Williams. *Gaussian processes for machine learning*. Springer, 2006.
- K. Rawlik, M. Toussaint, and S. Vijayakumar. An approximate inference approach to temporal optimization in optimal control. In *Advances in Neural Information Processing Systems*, pages 2011–2019, 2010.
- F. Reuleaux. The kinematics of machinery: Outlines of a theory of machines. German original (1875). Translated by A. Kennedy, 1876.
- E. Rimon and J. W. Burdick. Mobility of bodies in contact. Part I. A 2nd-order mobility index for multiple-finger grasps. *IEEE Transactions on Robotics and Automation*, 14(5):696–708, 1998.
- E. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, 1992.
- A. Rodriguez, M. T. Mason, and S. Ferry. From caging to grasping. *The International Journal of Robotics Research*, 31(7):886–900, 2012.
- M. Rolf, J. J. Steil, and M. Gienger. Goal babbling permits direct learning of inverse kinematics. *IEEE Transactions on Autonomous Mental Development*, 2(3):216–229, 2010.
- B. Rooks. The harmonious robot. *Industrial Robot: An International Journal*, 33(2):125–130, 2006.
- S. Russell and P. Norvig. *Artificial intelligence: A modern approach*. 2010.
- R. B. Rusu and S. Cousins. 3D is here: Point cloud library (PCL). In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1–4. IEEE, 2011.
- R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008.

- A. Sahbani, S. El-Khoury, and P. Bidaud. An overview of 3D object grasp synthesis algorithms. *Robotics and Autonomous Systems*, 60(3):326–336, 2012.
- Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura. The intelligent ASIMO: System overview and integration. In *Proceedings of the International Conference on Intelligent Robots and Systems*, volume 3, pages 2478–2483. IEEE, 2002.
- A. Saxena, J. Driemeyer, and A. Ng. Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, 27(2):157, 2008.
- S. Schaal. Perception–Action–Learning. In O. Brock, editor, *Berlin Summit on Robotics. Conference report*. Technische Universität Berlin, Alexander von Humboldt Foundation, 2011.
- D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, 2002.
- S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 519–528. IEEE, 2006.
- S. Sen, G. Sherrick, D. Ruiken, and R. Grupen. Choosing informative actions for manipulation tasks. In *Proceedings of the International Conference on Humanoid Robots (Humanoids)*, pages 721–726. IEEE, 2011.
- S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim. Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot. In *Proceedings of the International Conference on Robotics and Automation*, pages 3307–3312. IEEE, 2013.
- B. Siciliano and O. Khatib. *Handbook of robotics*. Springer, 2008.

- J. Sinapov and A. Stoytchev. Learning and generalization of behavior-grounded tool affordances. In *Proceedings of the International Conference on Development and Learning (ICDL)*, pages 19–24. IEEE, 2007.
- S. Singh, M. R. James, and M. R. Rudary. Predictive state representations: A new theory for modeling dynamical systems. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 512–519. AUAI Press, 2004.
- A. J. Smola, B. Schölkopf, and K.-R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11(4):637–649, 1998.
- P. Sollich. Probabilistic methods for support vector machines. *Advances in neural information processing systems*, 12:349–355, 2000.
- N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.
- J. Steffen, R. Haschke, and H. Ritter. Experience-based and tactile-driven dynamic grasp control. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 2938–2943. IEEE, 2007.
- F. Steinke, B. Schölkopf, and V. Blanz. Support vector machines for 3D shape processing. In *Computer Graphics Forum*, volume 24, pages 285–294, 2005.
- A. Stoytchev. Some basic principles of developmental robotics. *IEEE Transactions on Autonomous Mental Development*, 1(2):122–130, 2009.
- F. Stulp, E. Theodorou, J. Buchli, and S. Schaal. Learning to grasp under uncertainty. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5703–5708. IEEE, 2011.

- R. Suárez, M. Roa, and J. Cornellà. Grasp quality measures. Technical Report IOC-DT-P 2006-10, Universitat Politècnica de Catalunya, Institut d'Organització i Control de Sistemes Industrials, 2006.
- R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.
- P. Svestka and M. H. Overmars. Coordinated motion planning for multiple car-like robots using probabilistic roadmaps. In *Proceedings of the International Conference on Robotics and Automation*, volume 2, pages 1631–1636. IEEE, 1995.
- J. W. Tangelder and R. C. Veltkamp. A survey of content based 3D shape retrieval methods. *Multimedia tools and applications*, 39(3):441–471, 2008.
- J. Tegin, S. Ekvall, D. Kragic, J. Wikander, and B. Iliev. Demonstration-based learning and control for automatic grasping. *Intelligent Service Robotics*, 2(1):23–30, 2009.
- D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *ACM Siggraph Computer Graphics*, volume 21, pages 205–214. ACM, 1987.
- A. Tikhonov and V. Y. Arsenin. *Methods for solving ill-posed problems*, 1979.
- E. Todorov. Optimality principles in sensorimotor control. *Nature neuroscience*, 7(9):907–915, 2004.
- E. Todorov and Z. Ghahramani. Analysis of the synergies underlying complex hand manipulation. In *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*, volume 2, pages 4637–4640. IEEE, 2004.

- M. Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1049–1056. ACM, 2009.
- M. Toussaint. A Bayesian view on motor control and planning. In O. Sigaud and J. Peters, editors, *From motor to interaction learning in robots*. Springer, 2010a.
- M. Toussaint. Open robot simulation toolkit. Programming library documentation (online, visited 2014-02-12), 2010b. URL <http://ipvs.informatik.uni-stuttgart.de/mlr/marc/source-code/libORS.10.1.pdf>.
- M. Toussaint, N. Plath, T. Lang, and N. Jetchev. Integrated motor control, planning, grasping and high-level reasoning in a blocks world using probabilistic inference. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 385–391. IEEE, 2010.
- V. N. Vapnik. An overview of statistical learning theory. *Neural Networks, IEEE Transactions on*, 10(5):988–999, 1999.
- S. Vasudevan, F. Ramos, E. Nettleton, and H. Durrant-Whyte. Non-stationary dependent Gaussian processes for data fusion in large-scale terrain modeling. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1875–1882. IEEE, 2011.
- C. Walder, B. Schölkopf, and O. Chapelle. Implicit surface modelling with a globally regularised basis of compact support. In *Computer Graphics Forum*, volume 25, pages 635–644. Wiley Online Library, 2006.
- K. Waldron and J. Schmiedeler. Kinematics. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, pages 9–33. 2008.
- O. Williams and A. Fitzgibbon. Gaussian process implicit surfaces. *Gaussian Proc. in Practice*, 2007.

- W. A. Wolovich and H. Elliott. A computational technique for inverse kinematics. In *Proceedings of the 23rd Conference on Decision and Control*, volume 23, pages 1359–1363. IEEE, 1984.
- K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proceedings of the ICRA workshop on best practice in 3D perception and modeling for mobile manipulation*, volume 2, 2010.
- K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd international conference on Machine learning*, pages 1012–1019. ACM, 2005.
- D. Zarubin, V. Ivan, M. Toussaint, T. Komura, and S. Vijayakumar. Hierarchical motion planning in topological representations. In *Robotics: Science and Systems*, 2012.
- H. Zha, T. Hoshida, and T. Hasegawa. A recursive fitting-and-splitting algorithm for 3-D object modeling using superquadrics. In *Proceedings of the 14th International Conference on Pattern Recognition*, volume 1, 1998.

