

**Imperial College
London**

**Active Constraints for Robotic Surgery
in Deforming Tissue**

Stuart Bowyer

Department of Mechanical Engineering
Imperial College London

July 2014

This thesis is submitted for the degree of Doctor of Philosophy at Imperial College London

© Stuart Bowyer 2014

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.

Declaration of Originality

I confirm that the research presented within this thesis is my own work, with the exception of the elements which have been appropriately identified and referenced.

Stuart Bowyer

24th July 2014

Abstract

Active constraints are collaborative human-robot control algorithms which have a well-established history of use within robot assisted surgery research. This control strategy anisotropically regulates the motion of a human user in such a way that it effectively combines the competencies of surgeons and robots, allowing for improved clinical outcomes and surgeon experience. The significant majority of research previously presented for active constraints focuses on their application to static procedures, where the surgical environment is assumed to be rigid throughout. In this thesis, several research contributions are presented which assist with applying active constraints in surgical procedures within deforming soft tissue. The primary contribution is the formulation of a novel haptic control algorithm, based on friction, which can effectively guide a surgeon in both positioning and orienting a surgical instrument, while guaranteeing that the haptic interaction is energetically dissipative. The proven dissipative formulation of these ‘dynamic frictional constraints’ ensures that the surgeon always has overall control of a procedure and makes the system resilient to limitations and errors in the robot’s comprehension of the surgical environment. To apply active constraints within deforming tissue, it is necessary to compute the geometric relationship between the surgical instruments and the constrained anatomy. A novel bounding volume is proposed which, when used in a hierarchy, exploits the limits of soft tissue deformations to increase the resolution of constraint geometries that can be used at stable control rates. The experimental validation of these research contributions in a clinically realistic nerve dissection simulation and in non-clinical dynamic path-following tasks, shows significant benefits to the user in several metrics characterising surgical accuracy and precision. These results demonstrate that the proposed enhancements of active constraints could lead to increased surgeon performance, fewer complications and improved clinical outcomes in soft tissue surgical procedures.

Acknowledgements

I first wish to express my greatest thanks to my supervisor, Dr Ferdinando Rodriguez y Baena, for his direction and advice in my research. Perhaps most of all, I would like to thank Ferdinando for his apparent unwavering confidence in me, throughout good results, bad results, seemingly endless revisions of a certain paper and the odd piece of broken hardware. It has made for a really rewarding experience.

I am also very grateful for the assistance of Professor Brian Davies who, in addition to providing many useful ideas and suggestions, was the architect of active constraints in surgery, and without whom, my research project could not possibly have provided me with such an interesting and compelling challenge.

Of course, I am enormously indebted to the many past, present and associate members of the Mechatronics in Medicine Laboratory who helped me in many aspects of my research and, just as importantly, provided me with a great environment for the last few years. In approximate order of appearance, this includes: Matthew Oldfield, Seong Young Ko, Ryo Takeda, Yaroslav Tenzer, Tassanai Parittotokkarn, Elnaz Nobari, Antonio Forte, Jassim Alqabandi, Josh Petersen, Sandra Nwokeoha, Fangde Liu, Riccardo Secoli, Chris Burrows, Trevor Hawke, Alex Leibinger, Will Blythe, Hadi El Daou, Stefano Galvan, Elisa Baretta, Josh Giles and countless other interns and project students.

I would like to thank Mr Asit Arora and Mr Neil Tolley for their advice, ideas and positivity about the clinical applications of my research. Helping me with their already oversubscribed time was incredibly generous and really helped to bring my research to a meaningful conclusion.

I also gratefully acknowledge the Engineering and Physical Sciences Research Council (EPSRC) for funding my studentship and the Active Constraints Technologies for Ill-defined or Volatile Environments (ACTIVE) project FP7-ICT-2009-6-270460 for contributing to my research costs.

Finally, I would like to thank all of my friends and family, outside of college, who have supported me throughout.

Contents

Declaration of Originality	3
Abstract	4
Acknowledgements	5
Contents	9
List of Figures	11
List of Tables	12
List of Abbreviations	14
1 Introduction	15
1.1 Background	15
1.1.1 Robotic Surgery	15
1.1.2 Robot assisted Surgery	16
1.1.3 Active Constraints	18
1.2 Research Problem	19
1.3 Research Hypothesis	21
1.4 Research Contributions	22
1.5 Thesis Structure	23
2 Active Constraints/Virtual Fixtures	25
2.1 Introduction	25
2.2 Definitions of Terms	26
2.2.1 Active Constraints	26
2.2.2 Teleoperated vs. Hands-on Devices	27
2.2.3 Impedance vs. Admittance Device Control	27
2.2.4 Regional vs. Guidance Active Constraints	28
2.2.5 Attractive vs. Repulsive Active Constraints	30
2.2.6 Unilateral vs. Bilateral Active Constraints	30
2.2.7 Static vs. Dynamic Active Constraints	31
2.3 Generalised Active Constraint Framework	31
2.4 Constraint Definition Methods	33
2.4.1 Point Constraints	33

2.4.2	Linear Constraints	34
2.4.3	Parametric Curve Constraints	34
2.4.4	Hyperplanar Constraints	35
2.4.5	Parametric Surface Constraints	35
2.4.6	Polygonal Mesh Constraints	36
2.4.7	Point Cloud Constraints	36
2.4.8	Volumetric Primitive Constraints	36
2.4.9	Explicitly Described Constraints	37
2.4.10	Artificial Neural Network Constraints	37
2.5	Constraint Evaluation Methods	38
2.5.1	Spatial Partitioning and Bounding Volume Hierarchies	39
2.5.2	Feature Tracking Algorithms	39
2.6	Constraint Enforcement Methods	40
2.6.1	Simple Functions of Constraint Proximity	40
2.6.2	Proxy and Linkage Simulation	41
2.6.3	Non-Energy Storing Constraints	44
2.6.4	Potential Fields	45
2.6.5	Reference Direction Fixtures	47
2.6.6	Constrained Joint Optimisation	50
2.6.7	Passive Constraint Enforcing Mechanisms	51
2.7	Advanced Constraint Extensions	53
2.7.1	Adaptive Constraints	53
2.7.2	Compliance in Tools and Robot Mechanisms	55
2.7.3	Dynamic Active Constraints	55
2.7.4	Multi-Handed Active Constraints	57
2.7.5	Active Constraints on Handheld Manipulators	58
2.8	Discussion	58
2.8.1	Literature Overview	58
2.8.2	Research Trends and Directions	61
2.9	Summary	63
3	Deformation Invariant Bounding Sphere Hierarchies	64
3.1	Introduction	65
3.2	Related Work	67
3.2.1	Space Partitioning Hierarchies	68
3.2.2	Bounding Volume Hierarchies	69
3.2.3	Convexity Based Methods	71
3.2.4	Hardware Acceleration	72
3.3	Deformation Invariant Bounding Spheres	73
3.3.1	Surface ‘Pinned’ Bounding Volumes	73
3.3.2	Assumption of Limited Surface Stretch	75
3.3.3	Hierarchical construction	77
3.3.4	Real-time proximity queries	80
3.4	Experiments	82
3.4.1	Randomised Positioning and Deformation Experiment	82
3.4.2	Beating Heart Experiment	83
3.5	Results	84

3.5.1	Randomised Positioning and Deformation Experiment	84
3.5.2	Beating Heart Experiment	87
3.6	Discussion	87
3.7	Summary	91
4	Dynamic Frictional Constraints	93
4.1	Introduction	93
4.2	Formulation of Dynamic Frictional Constraints	99
4.2.1	Elastoplastic Friction Model	99
4.2.2	Using Friction for Constraints	100
4.2.3	Frictional Redirection	103
4.3	Experiments	108
4.4	Results	110
4.5	Discussion	112
4.6	Summary	113
5	Dynamic Frictional Constraints in Translation and Rotation	115
5.1	Introduction	115
5.2	Extending Dynamic Frictional Constraints to SE(3)	119
5.2.1	Elasto-plastic Friction Model in SE(3)	120
5.2.2	Elasto-plastic Friction in $\mathbb{R}^3$	121
5.2.3	Dynamic Frictional Constraints in $\mathbb{R}^3$	122
5.2.4	Dynamic Frictional Constraints in SO(3)	124
5.3	Coupled Frictional Redirection between Translation and Rotation	127
5.3.1	Combining Translation and Rotation	128
5.3.2	Elastic Potential Energies in $\mathbb{R}^6$	130
5.3.3	Conical Projections of Potential Energies	132
5.4	Experiments	133
5.4.1	Experimental Apparatus	133
5.4.2	Experimental Procedure	135
5.5	Results	137
5.6	Discussion	142
5.7	Summary	148
6	A Dynamic Proxy Formulation for Deforming Geometries	149
6.1	Introduction	149
6.2	Computing Constraint-based Proxy Motion using Gauss' Principle	152
6.3	An Implementation for Deforming Bodies	153
6.4	Discussion	157
6.5	Summary	158
7	Integrated Trials within a Surgical Simulation Environment	159
7.1	Introduction	159
7.2	Simulated Nerve Dissection Concept	161
7.3	A Complete 6-D Dynamic Active Constraint System	162
7.3.1	Deformation Invariant Bounding Sphere Hierarchy	164
7.3.2	Dynamic Proxy Simulation	165
7.3.3	Dynamic Frictional Constraints	167

7.3.4	Physical Soft Tissue Simulation	167
7.3.5	Hardware Interface	171
7.3.6	3-D Visualisation	172
7.4	Experiments	173
7.5	Results	175
7.6	Discussion	178
7.7	Summary	183
8	Conclusions and Future Work	185
8.1	Conclusions	185
8.2	Limitations and Future Work	189
	Bibliography	193
	Appendix A Proofs	209
A.1	Simplification of the elasto-plastic friction model	209
A.2	Dissipativity of dynamic frictional constraints in $\mathbb{R}^3$	211
A.3	Dissipativity of dynamic frictional constraints in $\text{SO}(3)$	214

List of Figures

1.1	Photos of examples of conventional and targeted environments in which active constraints can be applied.	20
2.1	Example teleoperated and hands-on robots.	28
2.2	Example regional and guidance constraints.	29
2.3	Example attractive and repulsive constraints.	30
2.4	Example unilateral and bilateral constraints.	31
2.5	Generalised active constraint implementation framework.	32
2.6	Example illustrations of the constraint representations described within the literature. . .	34
2.7	Illustration of a proxy-based constraint enforcement.	42
2.8	Illustration of a potential field active constraint configuration.	46
2.9	Illustration of a reference direction fixture where a user's force is decomposed into preferred (permitted by the constraint) and non-preferred (resisted by the constraint) components.	47
3.1	An example of how surface deformation can cause tightly fitting pinned bounding volumes to fail and how a conservative bounding sphere can overcome this.	75
3.2	Two dimensional illustration of the method for defining the pinning vertex, v_c	78
3.3	Example of orienting a dividing plane so that it is orthogonal to the eigenvector associated with the largest eigenvalue of surface vertices' covariance matrix.	79
3.4	Multi-resolution brain mesh used within the randomised validation of the deformation invariant bounding sphere method.	83
3.5	Image from the proximity query experiment between a beating heart mesh and a human positioned capsule.	84
3.6	Log-log plots of mean total search time against mesh size for the four proximity query methods used in the experiments with sphere and capsule tool primitives.	87
3.7	Plot of the crossover refit ratios above which deformation invariant bounding sphere hierarchies are more efficient.	89
4.1	Illustrations of the two base interaction cases for conventional dynamic active constraints. .	95
4.2	Two-port network to illustrate the energetic interaction between a user and a dynamic active constraint system.	96
4.3	Illustration of the elastoplastic friction model created by Dupont <i>et al.</i>	100
4.4	Illustrations of the anisotropy characteristics of simple frictional constraints in unconstrained, boundary and constrained regions.	104
4.5	Illustration of the parameters used within the frictional redirection method.	106
4.6	Sequence illustrating the three penetration states of the frictional redirection method. . .	107

4.7	Annotated view of the display used within the experimental validation of the constraint strategies.	109
4.8	Experimental results of the three performance metrics for 17 test subjects.	111
5.1	Illustration of a surgical scenario in which both the position and orientation of the complex surgical tool must be controlled to ensure successful and safe operation.	116
5.2	Illustrations of two active constraint configurations in which a translation only constraint would suffer from poor performance.	117
5.3	Illustration of the proposed translational dynamic frictional constraints.	123
5.4	Illustration of the conical projection method on the surface of a unit quaternion 3-sphere for rotational dynamic active constraints.	125
5.5	Illustrations of the concept of coupled frictional redirection between translation and rotation in a two degree of freedom system.	129
5.6	Illustration of the potential energy mapping which converts a spring displacement vector into the potential energy space.	131
5.7	Simulation of the dynamic steady-hand game used within the experiments.	134
5.8	A photo of the experimental apparatus used in the simulation of the dynamic steady-hand game to evaluate dynamic frictional constraints in SE(3).	135
5.9	Distributions of the contact percentage across the six constraint methods.	139
5.10	Distributions of the translational RMSE across the six constraint methods.	139
5.11	Distributions of the rotational RMSE across the six constraint methods.	139
5.12	Distributions of the mean translational velocity across the six constraint methods.	140
5.13	Distributions of the mean rotational velocity across the six constraint methods.	140
5.14	Distributions of the mean translational power across the six constraint methods.	140
5.15	Distributions of the mean rotational power across the six constraint methods.	141
5.16	Distributions of the mean total power across the six constraint methods.	141
6.1	Illustration of a proxy and virtual linkage configuration for a scalpel-like surgical tool.	151
6.2	Illustration of the conventional proxy motion scheme.	154
6.3	Illustration of the proxy motion scheme used in the dynamic active constraint which avoids the necessity for continuous collision detection.	156
7.1	Preview of the clinical simulation developed for the validation of dynamic active constraints.	162
7.2	System diagram for the software used within the simulated surgical environment.	163
7.3	Images of the multi-representation models used for the physical soft tissue simulation.	169
7.4	The alignment of the master haptic devices with the simulated surgical instruments.	172
7.5	The dissection process used by the test subjects during the experimental validation.	174
7.6	Experimental results of the four primary performance metrics across the three constraint methods.	177
7.7	Experimental results of the three secondary performance metrics across the three constraint methods.	179

List of Tables

1.1	Summary of the advantageous attributes of humans and robots which are combined by active constraint systems.	19
2.1	Tabulated summary of research contributions from the key literature within this survey. .	60
3.1	Comparative performance for proximity queries between randomly positioned spheres and deforming meshes of differing sizes.	85
3.2	Comparative performance for proximity queries between randomly positioned capsules and deforming meshes of differing sizes.	86
3.3	Comparative performance for proximity queries between a capsule and a deforming heart mesh.	87
4.1	Constraint parameters used within the experimental procedure.	110
4.2	Results for the three assistance methods with the three evaluation metrics.	111
4.3	Results of Tukey's honest significant difference test.	111
5.1	Parameters for translation and rotation used in the frictional constraints.	136
5.2	Summary of the results for the six constraint methods across the primary metrics. . . .	138
5.3	Summary of the results for the six constraint methods across the secondary metrics. . .	138
5.4	Results of Tukey's honest significant difference tests for the primary metrics at the 95 % confidence interval.	143
7.1	Parameters used in the dynamic proxy simulation for the nerve dissection task.	166
7.2	Parameters used in the frictional constraints for the nerve dissection task.	167
7.3	Parameters used in the mechanical models.	169
7.4	Summary of the results for the three constraint methods across the primary metrics. . .	176
7.5	Summary of the results for the three constraint methods across the secondary metrics. .	176
7.6	Results of pairwise statistical significance tests for means and variances at the 95 % confidence interval.	176

List of Abbreviations

AABB	Axis aligned bounding boxes
ACROBOT	Active constraint robot
AEP	Azimuthal equidistant projection
ANOVA	Analysis of variance
BD-Trees	Bounded deformation trees
CAD	Computer aided design
CAM	Computer aided manufacture
CPU	Central processing unit
CT	Computed tomography
CTG	Constrained tool geometry
CVT	Continuously variable transmission
DFC3R	Rotational dynamic frictional constraints
DFC3T	Translational dynamic frictional constraints
DFC6	Dynamic frictional constraints in 6-D
DFC6C	Coupled dynamic frictional constraints in 6-D
DIBS	Deformation invariant bounding sphere
DOF	Degree of freedom
DOP	Discrete oriented polytopes
FC6	Frictional constraints in 6-D
FEM	Finite element method
FPGA	Field programmable gate array
GPGPU	General purpose computing on graphics processor units
GPU	Graphics processing unit
HMM	Hidden Markov model

HSD	Honest significant difference
n-D	n-dimensional space
NIR	Near infra-red
NURBS	Non-uniform rational B-splines
OBB	Oriented bounding boxes
OROCOS	Open robot control software
P-TER	Passive trajectory enhancing robot
PADyC	Passive arm with dynamic constraints
PC	Personal computer
RIO	Robotic arm interactive orthopedic system (MAKO Surgical Corp.)
RLN	Recurrent laryngeal nerve
RMSE	Root mean square error
SE(3)	Special Euclidean group in three dimensions
SLERP	Spherical linear interpolation
SO(3)	Special orthogonal group of 3-D rotations
SOFA	Simulation open framework architecture
UC	Unconstrained
UDP	User datagram protocol
V-Clip	Voronoi-clip
VTK	The visualization toolkit (Kitware Inc.)

Chapter 1

Introduction

1.1 Background

1.1.1 Robotic Surgery

The first use of a robotic device in a surgical procedure was in 1985, when an industrial robot was used to guide a biopsy needle taking a sample from a brain lesion [Kwoh et al., 1988]. In this procedure, the robot was substituting for a stereotactic frame [Horsley and Clarke, 1908], which is a mechanical fixture widely used in neurosurgery, that is rigidly fixed to the patient's skull to provide alignment for a needle or probe while it is inserted into the brain. When using a stereotactic frame, surgeons first image the patient's anatomy, using a technique such as X-ray computed tomography (CT), and then set the pose of the fixture, based on the generated images, using a series of scales on the device. This adjustment process can be "tedious and time consuming" [Kwoh et al., 1988] and therefore the robot was employed to position itself autonomously once the needle's trajectory was specified computationally. In this first robotic surgical system, the researchers were taking advantage of two of the fundamental properties of robots, namely: that they can have high accuracy and precision in positioning and movement; and that they can be tightly integrated with other technology. The surgical importance of the first advantage is clear; however, it was the integration with other clinical technology, specifically imaging, which gave the robot a significant advantage over the non-robotic procedure.

Following on from the robotic guided neurosurgical biopsy, the first fully autonomous robotic surgical procedure was a transurethral resection of the prostate by the 'Probot' [Davies et al., 1989, 1991]. After

early trials with a six degree of freedom industrial robot, the Probot was a custom designed, four degree of freedom, motorised safety frame which had a restricted kinematic structure that ensured patient safety during the resection. Similarly to the earlier work in neurosurgery, the Probot was directly integrated with a preoperative, ultrasound, imaging system and a surgeon-computer interface. These elements of the Probot system meant that the surgeon could model the cut sequence directly within the patient's anatomy and transfer these plans for accurate enactment by the robot. Though the surgeon had to instruct the Probot to start each motion in the resection, the safety considerations in the system's custom design meant that, during each cut, the surgeon's only role was to observe the robot and interrupt it if deemed necessary.

In addition to the Probot, there have been a wide range of surgical robots researched which employ a similar form of autonomy, primarily in the field of orthopaedic surgery. In these systems, sometimes likened to the design and manufacture 'CAD/CAM' process [Taylor and Stoianovici, 2003], surgeons generate a surgical plan, based on preoperative imaging (CAD), which is undertaken by a robot (CAM). ROBODOC (Curexo Technology Corp.) is a robot of this form which was developed for resecting bone, to allow prosthesis placement, in hip and knee replacement surgeries [Paul et al., 1992; Taylor et al., 1990]. There have also been several other autonomous robots conceived for orthopaedic surgery, including: the industrial robot based CASPAR (formerly of URS Ortho GmbH & Co. KG) [Siebert and Mai, 2001] and the bone mountable ARTHROBOT [Kwon et al., 2002].

However, while autonomous robots have been the focus of a great deal of research, very few of them have reached clinical application, and only the ROBODOC was widely commercialised. Inspecting the literature, it is apparent that the primary challenge with these systems is that, irrespective of safety measures and redundancy, they are distancing the human surgeon from the procedure and this can lead to anxiety for patients, surgeons and clinical regulators [Davies, 1996].

1.1.2 Robot assisted Surgery

Though robotic systems have several benefits for surgery there are important areas in which their skills are currently inferior to surgeons, primarily with regards to interpreting the surgical environment and responding to unpredicted events. Robots are capable of operating in increasingly unstructured environments [Garcia et al., 2007]; however, the typical surgical environment is highly unstructured due to tissue deformation, cutting and intra- and inter-patient variation. For these reasons, there is a sig-

nificant amount of literature focussed on ‘robot assisted surgery’ which seeks to combine the different competencies of surgeons and robots.

Within robot assisted surgery, the robot itself is often treated as an ‘intelligent tool’ which is used to improve or extend the capabilities of the surgeon, rather than a device which performs surgery for itself. The da Vinci Surgical System (Intuitive Surgical, Inc.) is a robot assisted surgery system which typifies this concept [Guthart and Salisbury, 2000]. The da Vinci uses a master-slave teleoperation arrangement to improve the ease with which surgeons can perform a range of laparoscopic procedures. To use the da Vinci, the surgeon sits at a master console with a 3-D feed from the laparoscope and two six degrees of freedom handheld devices which can be manipulated to drive the laparoscopic instruments on the slave robot. In this system, the robot is not required to understand the nature of the procedure or make any clinical decisions for itself. Instead, the surgeon retains control of the surgery and decides how the instruments should move. The precise control of the da Vinci slave robots means that it is able to scale the user’s motion providing greater precision and dexterity, and the system’s integration within the surgical environment allows it to present relevant information to the surgeon in a more efficient and effective way. As of 2013, Intuitive Surgical, Inc. claim that approximately 1.5 million surgeries have been performed using the da Vinci surgical system [Intuitive Surgical, Inc., 2013], making it the most commercially successful surgical robot to date [Haidegger et al., 2009].

In addition to the da Vinci, there are several other robot assisted surgery systems which rely on a similar teleoperation arrangement to improve the ergonomics and usability for the surgeon at a master console, with the precise controlled motion of a slave device. These systems include the AESOP voice activated endoscope (Intuitive Surgical, Inc.) [Sackier and Wang, 1994] and the Sensei X Robotic Catheter System (Hansen Medical, Inc.) for atrial fibrillation [Wazni et al., 2009].

Another set of robot assisted surgery systems exist which, similarly to the neurosurgical biopsy assistant in [Kwoh et al., 1988], use a robot to position surgical guides or jigs. In these systems, the robot never cuts tissue itself and it is generally deactivated and has its brakes applied before the surgeon introduces the instruments and performs a procedure. To assist with a total knee replacement, an industrial robot was used in [Kienzle et al., 1992] to guide in the drilling of holes for the placement of cutting jigs. As the robot in this system was calibrated to the patient’s leg, it was necessary to rigidly immobilise the leg to ensure accuracy. The SpineAssist (Mazor Robotics Ltd.) spine mounted drill guide robot overcomes this problem by being directly attached to the patient’s bone structure so that it remains solid with the underlying anatomy during movement in surgery [Shoham et al., 2007], a similar approach

to which is taken by the APEX Robotic Technology (OMNIlife science, Inc.) for knee replacement surgery. These systems, and those which are similar, provide an efficient method for guiding surgeons during a procedure. However, the guides which the robots can carry are generally similar to those used in conventional surgeries, restricting the extension of these methods to new and novel procedures.

1.1.3 Active Constraints

‘Active constraints’, commonly cited synonymously with ‘virtual fixtures’, are a set of collaborative robot control strategies in which the user’s motion is intelligently regulated within an environment by a robot [Rosenberg, 1993]. Though initially conceived in other fields, active constraints have widely been researched as a method for robot assisted surgery, due to the fact that they allow for an effective combination of the attributes of surgeons and robots. Active constraints can be considered as a generalisation of the robotic guides or jigs, mentioned above, which can assist a surgeon in more complicated tasks. Rather than rigidly holding a guide for the surgeon at an accurate pose within the surgical workspace, the robot itself simulates the guide, complying to the surgeon’s commands when in a correct configuration and resisting the surgeon when moving in a direction that the guide would not permit.

The earliest application of the anisotropic guidance of active constraints in surgery was by the AC-ROBOT (formerly of The Acrobot Company Ltd.) [Ho et al., 1995; Jakopcic et al., 2003a] in the same knee replacement procedure as was considered by the guide holding robot described above. Rather than simply guiding the surgeon while drilling holes into which a cutting jig could be located, the complete cutting jig was removed and the user physically manipulated the cutter at the end of the robot in a ‘hands-on’ fashion. By varying the stiffness of the robot in response to the surgeon’s motion, the jig itself was simulated, keeping the surgeon’s cuts within the region requiring resection. In addition to the ACROBOT, active constraints have been commercially applied in orthopaedic surgery with the RIO robotic arm interactive orthopedic system (MAKO Surgical Corp.) [Hagag et al., 2011].

The increased flexibility of active constraints over their physical counterparts, as demonstrated in the early research, means that they can potentially be applied to many more intricate surgical tasks in which physical guides have never previously been implemented. For this reason, there has been a wealth of subsequent research which utilises the general concept of anisotropic motion regulation in human-robot collaboration to assist in a range of clinical and non-clinical tasks. In all of the literature generated from this research, the active constraint system attempts to combine one or more of the beneficial attributes

Humans	Robots
Comprehensive knowledge of patient/procedure	Accuracy and precision
General adaptability	Integration with other technologies
Diverse range of senses	Untiring strength and endurance
Legally responsible	Environmental flexibility

Table 1.1: Summary of the advantageous attributes of humans and robots which are combined by active constraint systems.

of human users and robots, as shown in table 1.1.

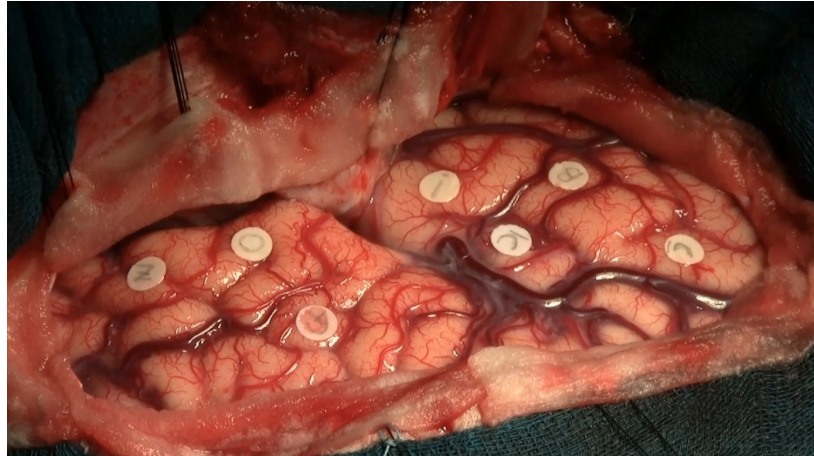
1.2 Research Problem

In the active constraint literature there are many methods and systems described which advance significantly the earliest concepts of active constraints in a variety of ways. However, almost none of the methods researched have found their way into commercial surgical systems and real clinical applications, or even show signs of developing in this way. The only exceptions to this have been the ACROBOT and the RIO. In applying active constraints to clearly defined procedures within orthopaedic surgery, these two systems achieved a vital simplification of the surgical environment which meant that they could apply active constraints in an effective and safe way. This critical simplification was that they could disregard all soft tissue within the patient's anatomy and only consider the robot's interactions with rigid bone. In procedures using these two systems, the surgeon makes the incisions in the soft tissue covering the patient's bones manually, and the soft tissue is then retracted away from the surgical site for the remaining procedure, as shown in figure 1.1a. This retraction means that the active constraints and the robot only need to consider the bone anatomy, which was scanned pre-operatively via CT imaging and was rigidly fixed (or tracked) intra-operatively.

One of the fundamental obstructions to current active constraint research being applied to more clinical tasks is that there are very few procedures in which simplifications to entirely rigid environments can be made. Instead, the overwhelming majority of clinical procedures involve working within close proximity to, or actively interacting with, deforming soft tissue structures with complex geometries. Though active constraints have the potential for great flexibility with regards to the forms of assistance they can offer to surgeons, in clinical practice they are regularly tied to procedures in which physical fixtures can be used, and are still often preferred.



(a) Conventional



(b) Targeted

Figure 1.1: Photos of examples of (a) conventional and (b) targeted [ACT, 2012] environments in which active constraints can be applied. The example conventional active constraint environment is from a unicompartmental knee replacement where the soft tissue has been retracted to show the femur, which is milled for the fitting of a prosthesis. The example targeted active constraint environment is from a neurosurgical procedure where the tissue is actively moving due to pulsation and will deform both locally and globally as the surgeon interacts with it.

In order to utilise the potential of active constraints for new and innovative surgical procedures, not just those in which physical guides are already in use, it is generally necessary to implement so-called ‘dynamic active constraints’ [Kwok et al., 2013]. Dynamic active constraints are a form of active constraints which dynamically change in response to deformation and movement within the operating environment, allowing for their use with soft tissue structures. The type of environment in which dynamic active constraints can be applied is shown in figure 1.1b. In this environment, the surgeon is interacting with soft tissue which moves by itself due to physiological phenomena such as pulsation and brain shift, and will deform and change as the surgeon proceeds.

Within the current literature there are comparatively few methods for dynamic active constraints, none of which constitute complete implementations within physical robots and clinical procedures. All of the methods presented consider abstracted or simplified dynamic tasks because there are still several challenges which must be overcome for them to be applied.

One of the primary challenges when applying dynamic active constraints to practical, soft tissue environments is how the relative configuration of the surgical instrument from the geometry of the active constraint can be computed. There are many methods within the literature for computing this relative configuration in static environments; however, these methods fail to work when the geometries con-

sidered deform over time. In addition to this difficulty, it is widely agreed upon within the literature that, for haptic systems, such as active constraint systems, the rate at which new robot commands are computed should be in the order of 1 kHz. Due to the limited speed of methods currently available, the update rate requirement can mean that constraint geometries must be simplified or approximated. As soft tissue tends to form complex surfaces, this restriction upon constraint geometry complexity within deforming environments can reduce their efficacy at assisting surgeons. This is a significant barrier to the widespread application of dynamic active constraints.

Another barrier to the utilisation of dynamic active constraints arises from the fact that the haptic interaction between the surgeon and the active constraint can be compromised when the constraint is dynamic. As a method for robot assisted surgery, active constraints are used to combine the abilities of humans and robots, as summarised in table 1.1. When implementing a dynamic active constraint within deforming tissue, movement in the soft tissue causes movement in the constraint geometry. Within most active constraint formulations, as they are currently defined, constraint movement can lead to constraints moving the surgical tools in an autonomous manner, i.e. without intention from the surgeon. When this happens, the active constraint system loses all of the benefits of the human involvement shown in table 1.1 and it becomes like the many autonomous surgical robots. As explained, autonomous surgical robots are arguably less desirable and proven commercially less successful than those which perform robot assisted surgery, and therefore this behaviour presents a challenge.

1.3 Research Hypothesis

The primary hypothesis of this thesis is that, by overcoming the challenges associated with implementing dynamic active constraints within deforming soft tissue strictures, dynamic active constraints can be applied to assist surgeons in increasingly unstructured procedures, thus improving surgical outcomes.

Specifically, this hypothesis consists of two components which relate to the two research problems outlined above for dynamic active constraints as they currently exist within the literature. Firstly, it is hypothesised here that a hierarchical search structure can be constructed which exploits the spatial coherence of deforming soft tissue structures to allow for the computationally efficient evaluation of the relative configuration between surgical instruments and constraint geometries during deformation. By utilising the properties of localised deformations within soft tissue, it is expected that highly resolved constraint geometries can be used which thoroughly recreate the underlying tissue anatomy, ensuring

accurate surgical assistance and improved effectiveness at the required haptic rate.

Secondly, it is hypothesised here that, by simulating the dissipative properties of friction, a new form of haptic control for dynamic active constraints can be created, which provides precise assistance to a surgeon, while ensuring that constraint geometry motion never causes autonomous motion of the surgical tools. These properties will mean that dynamic active constraints do not generate autonomous motion, protecting the fundamental balance of human and robot control over the surgery, thus giving dynamic active constraints the greatest chance of clinical approval and use within practical surgical procedures.

1.4 Research Contributions

Two major contributions to the state of the art in active constraint research have been generated while investigating the hypotheses presented above. These research contributions are summarised as follows:

- The concept for a new form of active constraint, a ‘dynamic frictional constraint’, has been proposed, which is based on a well established friction model and achieves performance that is comparable to a conventional formulation, while critically ensuring that the haptic controller never generates or stores energy and the surgeon retains overall control of the procedure at all times. The dynamic frictional constraint concept has been formalised and extended to fully constrain surgical instruments in both translation and rotation. This formulation allows the constraint to communicate far more haptic information to the surgeon regarding the directions in which they should, and should not, move. Additionally, this formulation has been proven to be energetically dissipative, again guaranteeing that the surgeon retains control. This research is presented in chapters 4 and 5.
- A novel bounding volume for use within a hierarchical search structure, a ‘deformation invariant bounding sphere’, which exploits the properties of deformations within soft tissue, has been defined. By eliminating the necessity of computationally demanding hierarchy refits and rebuilds, this structure has been shown to permit the use of significantly increased resolution constraint surface representations against well established methods from the literature, while ensuring the necessary control rates. This research is presented in chapter 3.

In addition to these major contributions, two secondary contributions of the research were produced. These are as follows:

- An extended proxy-based method has been developed for simulating the dynamic response of a rigid body colliding with another and ensuring non-penetration. The extension obviates the need for continuous collision detection between the objects in these scenarios, and can be efficiently used to define the target pose for a surgical instrument to which a dynamic active constraint has been applied, at the necessary control rate. This research is presented in chapter 6.
- The application of dynamic active constraints to assist in nerve dissection in soft tissue is proposed. This scenario is conceptually validated in simulation using all of the individual active constraint components developed throughout the thesis. This research is presented in chapter 7.

At present, four peer-reviewed publications have been produced from these research contributions. These are listed below:

S.A. Bowyer, B.L. Davies, and F. Rodriguez y Baena, “Active Constraints/Virtual Fixtures: A Survey”, in IEEE Transactions on Robotics, vol. 30, no. 1, pp. 138-157, February 2014.

S.A. Bowyer, and F. Rodriguez y Baena, “Deformation invariant bounding spheres for dynamic active constraints in surgery”, in Proceedings of the Institute of Mechanical Engineers, Part H: Journal of Engineering in Medicine, vol. 228, no. 4, pp. 350-361, April 2014.

S.A. Bowyer, and F. Rodriguez y Baena, “Dynamic frictional constraints for robot assisted surgery”, in Proceedings of the IEEE World Haptics Conference, pp. 319-324, April 2013.

S.A. Bowyer, and F. Rodriguez y Baena, “Dynamic frictional constraints in Translation and Rotation”, in Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2685-2692, June 2014.

1.5 Thesis Structure

In chapter 2, a thorough review of research published regarding active constraints, in both clinical and non-clinical applications, is presented. In addition to formally defining the common attributes shared by the diverse active constraint methods, a generalised framework for active constraint systems is proposed. This proposed framework is then used to present the research within the literature in a structured way.

In chapter 3, the challenge of evaluating the relative configuration between a surgical tool and a deforming constraint geometry is investigated. A novel bounding volume hierarchy is presented which can be

used to perform proximity queries between a mesh constraint surface and the geometrical description of a surgical instrument. Experimental validation of the proposed method is made against widely used methods from the literature.

In chapter 4, the haptic interaction between a surgeon and a dynamic active constraint is considered. In response to the associated challenges, a dynamic active constraint formulation based on friction is described, which guarantees that the respective competencies of humans and robots are combined. This formulation is then compared, in a dynamic path following task, to a conventional constraint which makes no consideration of these aspects, and their comparative attributes are analysed.

In chapter 5, the active constraint formulation devised in the previous chapter is formalised in three dimensional space (translation), it is extended for effective operation in six dimensional space (translation and rotation) and it is proven to be energetically dissipative, ensuring the favourable haptic interaction characteristics. A full experimental evaluation of the various forms of the control method is finally made in this chapter using a dynamic task, based on the steady-hand game, in which the user is required to simultaneously control both the position and orientation of a tool.

In chapter 6, a method for defining the target pose of a complex surgical tool, based on the surrounding constraint geometries, is presented. The method uses the simulated dynamic proxy approach from within the literature; however, it is modified to operate with deforming constraint surfaces and overcome the computationally demanding requirement for continuous collision detection between the tool and constraint geometries.

In chapter 7, a clinical scenario in which dynamic active constraints can provide a novel and potentially effective form of assistance to surgeons is proposed. Based on this prospective surgical procedure, a complete surgical simulation is assembled with a physical soft tissue mode, two surgical instruments, a 3-D visualisation and an active constraint controller based on the research presented in the previous chapters. A set of experiments upon this surgical procedure is then introduced to analyse the efficacy of the proposed active constraint formulation.

In chapter 8, several overarching conclusions are drawn from the research with regards to the initial hypotheses. Additionally, some of the limitations of the research are identified as well as directions and opportunities for future research within the field.

Chapter 2

Active Constraints/Virtual Fixtures

In this chapter, an extensive survey of the published literature in the field of active constraints is given. The research presented is an edited version of research previously published in:

S. A. Bowyer, B. L. Davies and F. Rodriguez y Baena, “Active Constraints/Virtual Fixtures: A Survey”, in IEEE Transactions on Robotics, vol. 30, no. 1, pp 138-157, February 2014. ©2014 IEEE.

2.1 Introduction

The field of active constraints and virtual fixtures has been the focus of a significant amount of research. ‘Virtual fixtures’, as conceived by Rosenberg, are the first published disclosure of this control concept, whereby “abstract sensory information [is] overlayed on top of reflected sensory feedback from remote environments” [Rosenberg, 1992, 1993]. As the name implies, virtual fixtures are loosely based on mechanical fixtures which anisotropically limit tool motion, but, as they are enforced in software, they are considerably more flexible with respect to positioning, real-time modification and feel. Rosenberg’s method includes both auditory and haptic virtual fixtures, but the fundamental concept is that by augmenting feedback from the slave to the master in a teleoperation system (see section 2.2.2), virtual fixtures can reduce mental workload, task time and errors.

Shortly after Rosenberg’s initial publication, in the field of robotic surgery there was little consensus on how best to ensure system safety [Davies, 1996; Taylor et al., 1991b], which had previously been approached using fully autonomous robotic manipulators [Kwoh et al., 1988]. It was in this climate

of exploring possible requirements for a more widely acceptable and safer form of control that Davies *et al.* devised the parallel concept they termed ‘active constraints’ for robot assisted knee replacement surgery [Davies et al., 2006; Ho et al., 1994]. Under this concept, surgical tools continue to be manually operated by the surgeon; however, they are attached to a back-drivable robotic manipulator. When the surgeon is cutting tissue which requires removal, the robot complies; however, as the surgeon moves towards material which should not be removed, the stiffness in that direction is increased to prevent over-cutting.

Since the conception of this form of control¹, many research institutions have developed and diversified it greatly. Some of the methods researched are for medical applications, but many are not, and there is currently no definitive concept which unifies the field. Previous survey papers of medical robotics (for example [Hager et al., 2008]) provide brief discussions of the area; however, they do not approach a complete and structured representation of the broad range of research. This chapter aims to achieve such a depiction for the research carried out in these areas under the general title of ‘active constraints’. The articles cited represent all currently relevant works, which contain a significant conceptual advancement, found using search terms associated to ‘virtual fixture’, ‘active constraint’ and ‘motion constraint’ in scientific publication databases such as IEEE Xplore, ScienceDirect, CiteSeer and Google Scholar.

2.2 Definitions of Terms

There are several terms used to describe active constraint algorithms within the literature. These terms are introduced here so they can be referred to freely in the following descriptions and explanations.

2.2.1 Active Constraints

We formally define ‘active constraints’, and synonymously ‘virtual fixtures’, as collaborative control strategies which can be used in human manipulation tasks to improve or assist by anisotropically regulating motion. Motion regulation is achieved by attaching tools to a robotic arm which is primarily controlled by a human user, under either hands-on or teleoperation control. Throughout operation, the robot controller monitors tool motion, and analyses it with respect to preplanned trajectories and known

¹Independently conceived as ‘synthetic fixtures’ [Sayers and Paul, 1993], ‘position assist shared control’ [Unruh et al., 1993], ‘task oriented virtual tools’ [Kosuge et al., 1995] and ‘motion constraints’ [Joly and Andriot, 1995].

restricted regions. The active constraint controller then attenuates or nullifies any user command which will cause the manipulator to digress from a plan, or enter a restricted region.

2.2.2 Teleoperated vs. Hands-on Devices

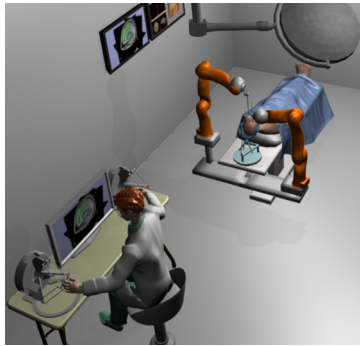
Devices implementing active constraints fall into two categories depending on where the human user interacts with the system. ‘Teleoperated’ devices are those where the human user is physically removed from the tool-carrying slave manipulator and controls it via a separate ‘master’ device [Sheridan, 1989] (see figure 2.1a). Telemanipulation offers benefits such as operation in restricted and unsafe environments and motion scaling. However, a key challenge with telemanipulation is that any environmental information required by the human user has to be interpreted at the slave device, transmitted to the master and then presented in some way.

‘Hands-on’ devices are those where the human user interacts directly with the tool-carrying manipulator [Jakopec et al., 2003a], that is to say that there is a rigid link between the user’s hand and the tool tip (see figure 2.1b). As the human user is holding the actual tool, the benefits such as improved access and motion scaling cannot be provided; however, the user directly feels the tool’s interaction with the environment. Coupling this direct feedback with the fact that the user is manipulating a ‘real’ tool, means that they will feel well integrated into the procedure, and will likely have a more intuitive ability to perform tasks. For applications where erroneous or uncontrolled tool positions present a significant danger, a hands-on robot is likely to be safer than a teleoperated one because the user can regulate it and the actuation system can be manufactured to be more compliant.

If the actuators of a hands-on device are used to regulate the motion of a tool, then this will constrain both the user and the tool with perfect correspondence between them. However, for a telemanipulation system, the constraint could potentially be applied to the master and/or slave devices, and depending on the teleoperation control architecture, these can have considerably different properties. An analysis of these aspects in teleoperation is given in [Abbott and Okamura, 2003a].

2.2.3 Impedance vs. Admittance Device Control

Controllers which implement active constraints in master, slave or hands-on devices generally implement a form of either impedance or admittance control. These are used because they both provide natural



(a) Teleoperated robot



(b) Hands-on robot

Figure 2.1: Example teleoperated and hands-on robots: prototypical surgical suite in the FP7 ACTIVE project [ACT, 2012] (a) and Sculptor Robotic Guidance Arm (b). Image courtesy of Stanmore Implants Worldwide Ltd. [Stanmore Implants Worldwide Ltd., 2012]. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

frameworks for responding to changes in the device’s environment as caused by tool interactions or the human user. As their names suggest, they are both based on the reciprocal mechanical properties of impedance and admittance/compliance. ‘Impedance’ control maps the motion of the device caused by environmental interactions to actuator forces/torques [Hogan, 1985], whereas ‘admittance’ control maps forces/torques being applied to the device by the environment to manipulator motion.

An important distinction between impedance and admittance controlled devices is that those which are impedance controlled must be highly backdriveable so that the environment (or human user) is able to move them freely when there is no controller force; however, admittance controlled devices are highly stiff in response to environmental forces and only move when commanded by the controller.

Though the intended effect of an active constraint controller is the same, an impedance constraint conceptually acts quite differently to an admittance constraint. In an impedance constraint the controller seeks to apply a force to the user which nullifies, to some degree, the motion which violates a constraint. Conversely, an admittance constraint acts like a filter on the user’s intended motion, which only allows the component of motion which does not violate a constraint to be passed to the robot’s actuators.

2.2.4 Regional vs. Guidance Active Constraints

Most of the active constraints considered within this survey have one of two purposes. ‘Regional’ constraints (commonly called ‘forbidden region virtual fixtures’ [Abbott et al., 2007]) bound a manipulator’s tool to certain regions within its task or joint space. That is to say that the constraints prevent the tool

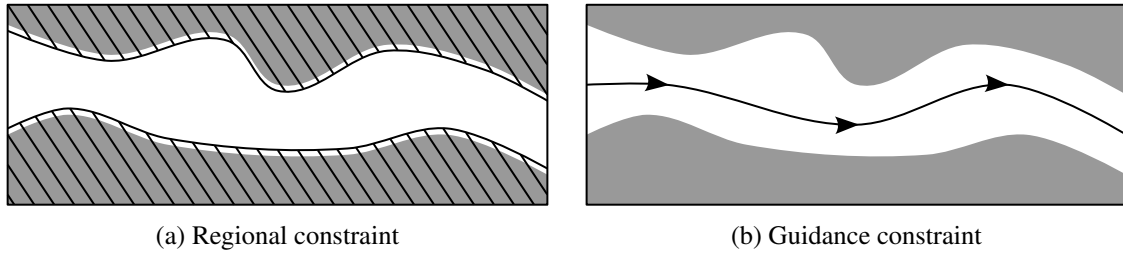


Figure 2.2: Example regional (a) and guidance (b) constraints. Obstacles are shown in grey and the constraints in black. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

from entering certain poses and configurations. Regional active constraints offer benefits such as task simplification [Payandeh and Stanisic, 2002; Rosenberg, 1993], reduced risk of tools damaging protected regions/obstacles [Li et al., 2005] and avoidance of kinematic singularities [Turro et al., 2001]. ‘Guidance’ constraints (commonly called ‘guidance virtual fixtures’ [Abbott et al., 2007]) encourage the user to move the tool along a specific pathway or towards a specific target [Bettini et al., 2001; Burghart et al., 1999]. By their nature, guidance constraints are more intrusive upon the user than regional constraints in normal operation, nevertheless, the ability to constrain positions precisely is invaluable in many applications. Examples of these two constraint types are shown in figure 2.2.

There is an extension to standard guidance constraints which sometimes appears within the literature, commonly in relation to potential fields. We refer to these as ‘bound guidance’ constraints and they can be thought of as guidance constraints limited to only a specific region of the robot’s task space. They are primarily used in assembling curved guidance constraints from multiple linear guidance sections. Here the limited bounds on each section allow the user to pass from one to another [Prada and Payandeh, 2009].

The distinction between regional and guidance constraints is often vague with respect to their control implementations as, for example, very restrictive regional constraints could be considered as guidance constraints [Bettini et al., 2004]. The division is; however, important from an application perspective.

There are certain active constraint implementations where a guidance constraint actively ‘pushes’ the user towards a specific pose as a form of trajectory control [Castillo-Cruces and Wahrburg, 2010; Pezzementi et al., 2007; Sayers et al., 1992]. In general, this type of control is rare within the active constraint literature; however, it is a fundamental feature of the so-called ‘impedance-based assistance’ which is commonly used within rehabilitation robotics. The extensive rehabilitation robotics literature, such as [Marchal-Crespo and Reinkensmeyer, 2009], can provide more information if it is required.

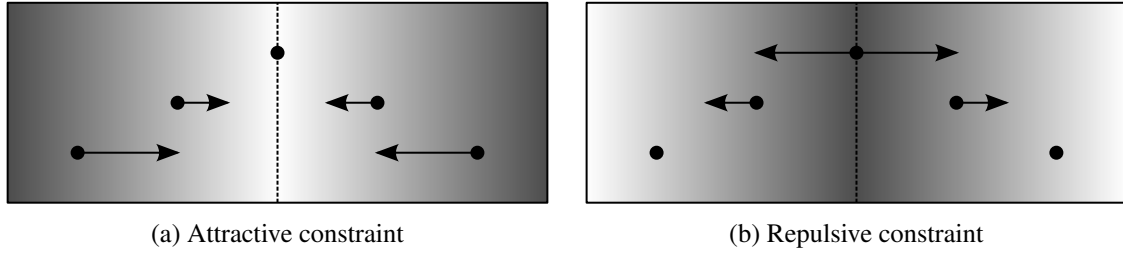


Figure 2.3: Example attractive (a) and repulsive (b) constraints. The constraint acts at the dashed line, with permitted regions shown in lighter colours and restricted regions shown in darker colours. Arrows show the direction of ‘encouraged’ motion. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

2.2.5 Attractive vs. Repulsive Active Constraints

The way in which active constraints act is considered in the literature as either: ‘attractive’ (encourages motion towards the constraint) or ‘repulsive’ (encourages motion away from the constraint) [Kuang et al., 2004]. Examples of these two constraint types are shown in figure 2.3. The difference between these two types of constraint is best exemplified by considering a unilateral (one-sided), regional active constraint. If an attractive constraint were used then the user’s motion would only be modulated once the tool had already passed into the restricted region, whereas for a repulsive constraint the modulation would come into effect before the tool reached the constraint. As with regional/guidance constraints, the difference between attractive and repulsive constraints is largely superficial rather than inherent in the control. This similarity is seen when considering an attractive constraint that encourages the user toward a linear pathway. This can, in fact, act identically to a large repulsive cylindrical constraint which has been centred on the pathway.

2.2.6 Unilateral vs. Bilateral Active Constraints

Though only relevant to some constraint geometries, the directionality of an active constraint is referred to in the literature as either ‘unilateral’ (acting on only one side) or ‘bilateral’ (acting on both sides) [Kikuuwe et al., 2006]. Conventionally, a regional constraint would be constructed from unilateral surfaces which prevent tool motion into the restricted space, whereas a guidance constraint would typically be bilateral, always encouraging tool motion towards the guidance path or point.

One important consideration with respect to unilateral constraints is that in some configurations they become discontinuous such that, if they are applied carelessly, they can cause unsmooth, unstable or erratic robot motion. This problem arises when the constraint enforcement function acts particularly

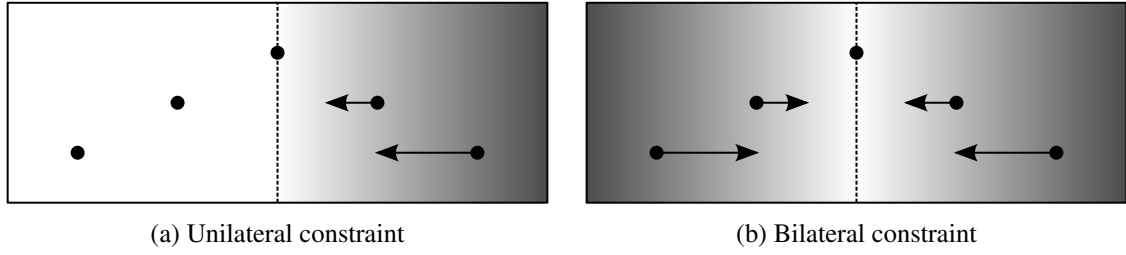


Figure 2.4: Example unilateral (a) and bilateral (b) constraints. The constraint acts at the dashed line, with permitted regions shown in lighter colours and restricted regions shown in darker colours. Arrows show the direction of ‘encouraged’ motion. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

strongly close to the constraint geometry so that, as the tool passes from the unconstrained to constrained sides, the constraint algorithm’s influence increases abruptly. A common method to overcome these problems is to introduce a transition region between the two sides which gradually increases the effect of the constraint as the user approaches the boundary [Burghart et al., 1999; Ho et al., 1995; Xia et al., 2008]. Temporal functions have also been used for this purpose, where the effect of the constraint is gradually increased over time when the tool is within the constrained region [Prada and Payandeh, 2005, 2009].

2.2.7 Static vs. Dynamic Active Constraints

A final distinction between active constraint implementations is as to whether they consider changing constraint geometries during operation. Of the literature surveyed, very few papers make a significant consideration of ‘dynamic active constraints’ where the constraint geometry continuously moves as a result of environmental changes [Bowyer and Rodriguez y Baena, 2013; Gibo et al., 2009; Kwok et al., 2013; Navkar et al., 2012; Ren et al., 2008; Rydén and Chizeck, 2012]. The overwhelming majority of the literature assumes that the environment is stable enough for the use of static active constraints. It is important to constrain deforming or moving geometries in some applications; however, the sensing and computational demands of dynamic constraints are a significant challenge.

2.3 Generalised Active Constraint Framework

From the initial concept, active constraints have been developed and modified for use in a wide range of robotic devices. To present the entirety of the field in a systematic way, a generalised active constraint

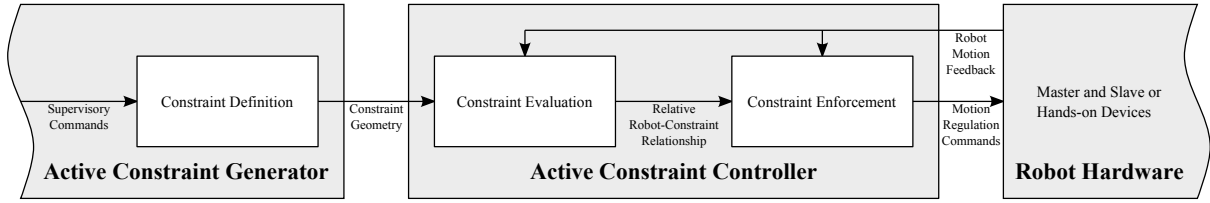


Figure 2.5: Generalised active constraint implementation framework. This illustrates the structuring of a conventional active constraint implementation with three primary components: constraint definition; constraint evaluation and constraint enforcement. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

framework is first introduced. This framework is shown in figure 2.5, and is based on three primary processes which are required in all practical active constraint implementations.

The first stage in applying an active constraint is the input of the geometry which will constitute the constraint itself. In the generalised active constraint framework this task is carried out by the ‘constraint definition’ module. The input to this module can be via human user supervision or autonomous sensing and the output will be some computational representation of the geometry.

Before any constraint can be enforced, the relative configuration of the constraint and the robot must first be evaluated so that the anisotropy in the motion regulation can be established. This relative configuration can take a variety of forms; however, it is typically a pair of colliding or closest points, with one on the constraint and one on the robot. This stage of the active constraint computation is carried out by the ‘constraint evaluation’ module.

Once the relative configuration of the constraint geometry and the robot is known, the active constraint can be applied. This is undertaken by the ‘constraint enforcement’ module which converts the relative robot-constraint configuration into hardware specific commands which will regulate the motion of the human user.

Methods in the literature for these three stages are described in the following sections. The three modules will be considered separately for clarity; however, there are some methodological interdependencies which will be identified as necessary.

2.4 Constraint Definition Methods

The vast majority of the literature assumes that the geometries used as active constraints are defined *a priori* by a separate process, which often involves the supervision of a human operator. There are some methods described within the literature for autonomously generating constraint geometries based on medical image processing [Jang et al., 2012; Kumar et al., 2003; Li and Taylor, 2003; Navkar et al., 2012; Park et al., 2011]; however, these are not deemed to be general enough for detailed discussion in this survey.

The only widely applicable constraint definition methods within the active constraint literature are those described in [Lopez et al., 2013; Tang et al., 2013]. In these publications, it is shown how a user can intra-operatively select points within a robot's workspace and then build a constraint from them. In [Lopez et al., 2013] a simple planar constraint is fitted to the selected points and in [Tang et al., 2013] a convex polyhedron is fitted to the points. Though these methods are intuitive for the user and avoid the need for imaging and registration, they give very simplified and discrete constraint geometries which will not necessarily fit the underlying anatomy very tightly.

Though the remaining methods for computing constraint geometries are too specific for inclusion in this survey, the methods for computationally representing them are widely applicable. Additionally, the chosen constraint representation directly affects the geometrical form that constraints can take and the subsequent choice of constraint evaluation method.

Descriptions of various methods for computationally representing active constraints are presented in this section. Simple methods using points, lines and planes are first introduced and then increasingly flexible surface representations are described. Finally, volumetric and explicit representations are explained. Simple examples of each type of constraint are illustrated in figure 2.6.

2.4.1 Point Constraints

Although they are simple, points in the robot's task space have been widely used in the literature for the definition of guidance constraints to assist with tool positioning [Bettini et al., 2001; Kapoor et al., 2005; Prada and Payandeh, 2005]. Proximities to points can be computed using simple closed form methods and they can be represented efficiently in memory. Point constraints are not restricted to purely Cartesian space and can be used for rotation and configuration control; however, their potential applications are

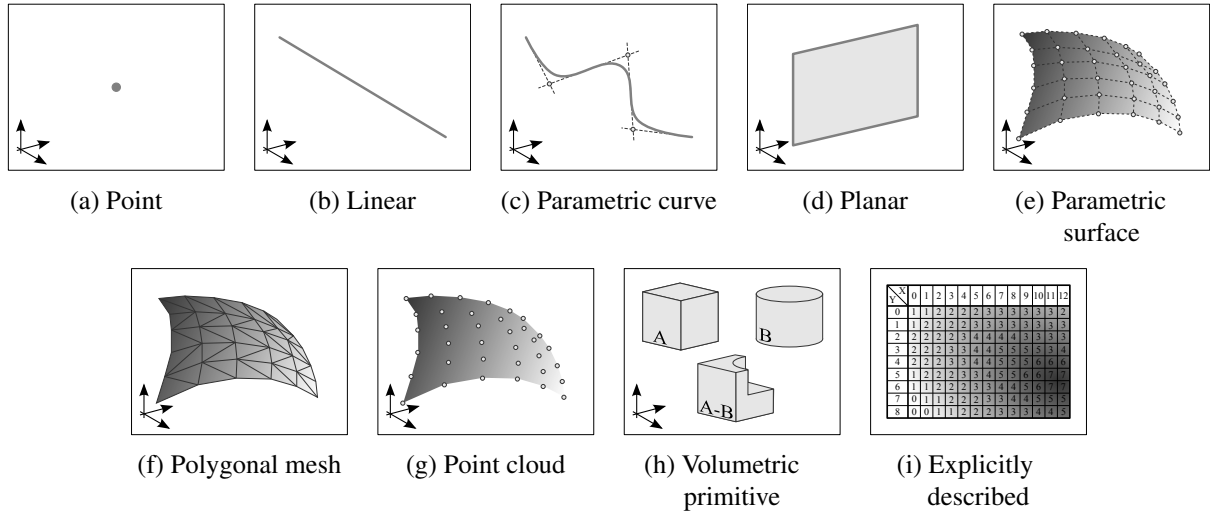


Figure 2.6: Example illustrations of the constraint representations described within the literature. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

limited to simple positioning-like tasks.

2.4.2 Linear Constraints

Linear constraints formed from vectors within a robot’s task space can be used for straightforward guidance constraints [Burghart et al., 1999; Srimathveeravalli et al., 2007]. As with point constraints, they offer closed form constraint evaluation methods and small memory demands, but the number of path following tasks which can be represented as straight lines is limited. It has been shown however, that combinations of multiple short, linear, constraints can be combined for more complex tasks [Aarno et al., 2005]. An important consideration with linear (and indeed point and parametric curve) constraints is that close to the line itself, small changes in position lead to large changes in constraint direction. This can be an issue for stability and therefore adding a radius to the line to form a tubular constraint has also been investigated [Bettini et al., 2004].

2.4.3 Parametric Curve Constraints

Guidance constraints are commonly represented in the literature as parametric curves ranging in complexity from sinusoidal functions to splines [Bettini et al., 2004; Kwok et al., 2013; Li and Taylor, 2005; Marayong and Okamura, 2004]. Parametric curves can be geometrically flexible and are able to describe complex tool paths, but the cost of this is that they do not typically have closed form constraint evalu-

ation methods. It is shown in [Navkar et al., 2012] how guidance constraints in the form of parametric curves can be generated in real-time from cardiac MR images. By ‘extruding’ 2-D curves into 3-D space (so-called 2.5-D), volumetric constraints can be represented [Ho et al., 1995]. This is a computationally efficient method for creating 3-D constraints, as any geometry can be projected onto the plane of the 2-D curve for constraint evaluation.

2.4.4 Hyperplanar Constraints

Hyperplanes can be used to divide a robot’s task space into separate subspaces, and are an effective method for defining both regional and guidance constraints [Rosenberg, 1993; Seung et al., 2010]. Hyperplanes are compact to represent in memory and relative robot-constraint configurations can be evaluated using closed form techniques. Hyperplanes are geometrically rigid, making it difficult to apply them to complex environments; however, in [Park et al., 2001] it is shown that they can be used to assist in the task of artery dissection. In this work, Park, Howe and Torchiana used medical images to align a single plane between an artery requiring dissection and the chest wall to which it was attached. By constraining the surgical tool so that it could only move within this plane, the surgeon was assisted in accurate resection. By combining multiple hyperplanes, constraints can be built to represent semi-complex geometries. However, it has been shown that once a certain level of complexity is reached, the efficiency degrades and more complex methods become faster to evaluate [Kapoor et al., 2006].

2.4.5 Parametric Surface Constraints

Parametric surfaces are the extension of parametric expressions from curves into 3-D surfaces. Little research has been found where parametric surfaces have been used for active constraints [Burschka et al., 2005]; however, non-uniform rational B-splines (NURBS) and other parametric surfaces have been used successfully within the field of haptic rendering [Ikits et al., 2003]. By varying the unilateral/bilateral and attractive/repulsive properties, a parametric surface can be used to define both regional and guidance constraints.

2.4.6 Polygonal Mesh Constraints

Polygonal meshes provide great flexibility for representing constraint geometries, but they are considerably more complex to construct, evaluate and store than the more simple representations. For some applications, the degree of surface conformity offered by meshes is not worth the increased complexity; however, for applications where the constraint profiles are extracted from ‘real world’ surfaces, they are a useful way to ensure tight regional conformance [Aleotti et al., 2005; Li et al., 2007; Ren et al., 2008]. Uniquely, in [Turro et al., 2001], coarse polygonal meshes were instead used to represent robot link geometries for checking against workspace obstacle constraints, although analytical primitives could also be used for this purpose. Methods for performing proximity queries with mesh based geometries are shown in section 2.5.

2.4.7 Point Cloud Constraints

By sampling a set of Cartesian points on the surface of a geometry, a representative point cloud can be constructed. Point clouds are commonly produced from 3-D scanners, range cameras or fiducial tracking systems, and can be produced and maintained effectively in real-time. Within the literature, point clouds have been used for both static [Yamamoto et al., 2012] and dynamic [Rydén and Chizeck, 2012] active constraints.

2.4.8 Volumetric Primitive Constraints

Analytical geometric primitives, such as spheres, cylinders, cones and cuboids (and their higher dimensional equivalents), can be defined using simple mathematical expressions, allowing them to be evaluated in a computationally efficient way. They can be used individually for representing regional constraints, but some researchers have shown that small primitives can be combined such that they form guiding constraint pathways [Bettini et al., 2004]. As combining volumetric primitives is an effective way of representing more complex regional and guidance constraints, several groups have proposed standardised and systematic methods for assembling them [Arata et al., 2010; Kuang et al., 2004; Prada and Payandeh, 2009].

2.4.9 Explicitly Described Constraints

Though less common in modern research, some implementations in the literature use explicit descriptions of constraint configurations. These representations can be considered as analogous to lookup tables which describe the constraint's proximity, or effect, at every pose within the robot's workspace. During operation, the robot controller 'looks up' the current pose within this table, and retrieves a value which is used to decide if, and how, constraints should be applied. Rather than storing a large look-up table covering the whole workspace, the field is often described using a numerical function which can be efficiently evaluated during operation [Aigner and McCarragher, 1997; Turro et al., 2001]. As many constraint methods only depend upon the current pose within the task space, the constraint enforcement can also be precomputed across the workspace and included in the lookup table. This is the case with force fields, where the controller looks up a force vector for the current pose, and applies it without any 'in-loop' computation.

It has been shown in the literature that explicitly represented tabular constraints can be tailored for specific applications in order to reduce memory requirements. The regional constraint profiles required for knee surgery were found by Ho *et al.* to have a loosely circular profile, and this allowed them to be represented as a 1-D lookup table of polar coordinates, necessary due to the slow processing speeds of the time [Ho et al., 1995].

A common limitation to all explicit methods is that, as they are defined before operation, they are generally inflexible with respect to unpredictable variation in their environment. Rigid transformations can be applied to keep explicitly defined constraints in the correct location relative to environmental features, but to date, this has not been demonstrated for complex, non-rigid, applications.

2.4.10 Artificial Neural Network Constraints

Cretu, Petriu and Patry researched a method for representing 3-D geometries in a memory efficient manner by training a neural network to output surface proximity values for any supplied Cartesian location [Cretu et al., 2003]. Their neural network is taught the object that it is required to represent using backpropagation with an explicit lookup table of the proximity values in the Cartesian volume. The main advantage of this method over the explicit representation methods is that it is comparatively memory efficient as the neural network is actually equivalent to an implicit surface function. Only the

set of interconnection weights must be stored and relatively few calculations are required at runtime.

This technique has been used for the representation of a dynamic active constraint based on the surface of a periodically beating heart [Ren et al., 2008]. A method for morphing geometries represented in this way was investigated by Cretu *et al.* [Cretu et al., 2003]; however, their method was based on linear interpolation between known start and finish network representations. Neural networks are typically ‘black boxes’ and the effect of changing weighting values is extremely difficult to predict. This means that using their method for non-periodic deformation, which cannot be predicted and preprocessed, is likely to be challenging.

2.5 Constraint Evaluation Methods

To decide on the necessity, and direction of anisotropy, of an active constraint, the relative configuration of the robot and the constraint must be evaluated. Conventionally, the geometric relationship of interest is the proximity between the robot’s tool tip and the active constraint geometry, or simply whether the two have collided. There are also some implementations in the literature where proximities to constraint features such as apexes and centres are required [Kuang et al., 2004; Prada and Payandeh, 2009]. In addition to the tool tip, there are a small number of methods in the literature where tool bodies or robot links [Kwok et al., 2013; Li et al., 2007; Turro et al., 2001] are constrained and therefore the relative configuration of the constraint and these parts is also required. Within this survey the term constrained tool geometry (CTG) will be used to abstractly refer to the tool geometry that is to be constrained, whether this is tool tip, tool body or robot.

The majority of the constraint evaluation methods within the literature are based on simple constraint representations and, subsequently, closed form expressions from linear algebra are sufficient. In most of the remaining literature, either explicit constraint representations are used, making constraint evaluation trivial, or the method used is not discussed. In the few other cases, the two key methods used, namely spatial partitioning/bounding volume hierarchies and feature tracking, are described below. This is in no way a complete review of proximity querying, nearest neighbour searches or collision detection; only methods currently used within the active constraint literature are discussed. These areas are well covered within fields such as robot motion planning and computer graphics, and therefore the reader is guided to [Clarkson, 2005; Ericson, 2004; Jimenez et al., 2000; Larsen et al., 1999] if more detail is required.

2.5.1 Spatial Partitioning and Bounding Volume Hierarchies

Space partitioning trees and bounding volume hierarchies are widely used to allow efficient searching of geometric objects for intersecting and closest points. These structures are constructed so that during traversal, large sections of the objects can quickly be eliminated without needing to compare every low-level surface primitive [Ericson, 2004].

Covariance trees are a variation on spatial partitioning k -d trees [Bentley, 1975], where the splitting planes are oriented based on an object's covariance so that the splitting of the surface is more efficient, minimising recursions during the search [Williams et al., 1997]. These covariance trees were used in [Li et al., 2007] for potential collision detection in regional constraint assisted endoscopy. In this implementation, potential collisions between line segments defining the tools and a mesh representing the patient's anatomy are checked using a covariance tree. To predict, rather than just identify collisions, each node's bounding volume was inflated by a proximity threshold. This expansion means that, when tree collisions are detected, the tool is actually only close to the constrained tissue, and the constraint can be used to prevent it from moving any closer. Bounding volume hierarchies are efficient to search, but their initial generation can be time consuming and therefore their application to dynamic constraints would require the investigation of real-time modification such as is discussed in [Larsson and Akenine-Möller, 2001; Spillmann et al., 2007; van den Bergen, 1998].

2.5.2 Feature Tracking Algorithms

Sequential constraint evaluations within an active constraint controller will tend to be reasonably similar if the control frequency is high with respect to the speed of robot movement, as is generally the case. The temporal coherence between evaluations means that incremental feature based algorithms are well suited. Methods such as Lin-Canny use the closest points from the preceding computation step to identify a small neighbourhood where the closest points at the current step should be found [Lin and Canny, 1991]. This small neighbourhood means that searches at each stage are considerably reduced from searching the complete objects. Voronoi-clip (V-Clip) [Mirtich, 1998], a modified version of Lin-Canny able to process concave objects, was used in [Aleotti et al., 2005] to evaluate the proximity between a user's hand and a cylindrical guidance constraint for peg placement tasks.

2.6 Constraint Enforcement Methods

There is a wide variety of methods described within the literature for enforcing active constraints. In general, the input to these methods is the current relative configuration of the constrained tool geometry and the constraint geometry. Based on this relative configuration, the active constraint enforcement module decides if, and how, anisotropic motion regulation should be applied. Seven primary approaches for enforcing active constraints were found within the literature and each of these is presented in this section. The constraint enforcement methods have significant differences between them and each have attributes which lend themselves to certain applications over others.

2.6.1 Simple Functions of Constraint Proximity

The most common constraint enforcement methods described in the literature are those where the effect of a constraint on a device is computed using a simple function of the proximity between the constrained tool geometry and the constraint.

Linear functions, generally based on minimum constraint proximity, have been widely used to enforce both regional [Gibo et al., 2009; Lopez et al., 2013; Seung et al., 2010] and guidance [Prada and Payandeh, 2005, 2009] constraints. The mechanical analogy of an elastic spring between the constrained tool geometry and constraint (with stiffness k_p) is often used when describing linear functions, as is the classic control concept of a proportional position controller. A linear function for a Cartesian, pure spring, impedance controller takes the form:

$$\mathbf{f}_p = k_p(\mathbf{p}_{const} - \mathbf{p}_{ctg}), \quad (2.1)$$

where $\mathbf{f}_p$ is the constraint force vector, k_p is a linear gain, $\mathbf{p}_{const}$ is the closest point on the constraint geometry to the constrained tool geometry and $\mathbf{p}_{ctg}$ is the closest point on the constrained tool geometry to the constraint. Alternating the value of k_p between positive and negative changes the constraint between attractive and repulsive respectively, changing its magnitude affects how strongly the constraint acts upon the robot and a piecewise function can be incorporated to implement unilateral constraints. Using the Jacobian and the principle of virtual work [Craig, 2005], the constraining force $\mathbf{f}_c$ can be applied to a device via its joint actuators.

The stability of linear constraints for teleoperation was investigated at static-force equilibrium by Ab-

bott and Okamura [Abbott and Okamura, 2003b]. They initially found that using Hannaford’s two-port model with Llewelyn’s criteria for unconditional stability [Adams and Hannaford, 1999] was unsuitable for predicting constraint stability and it led to unstable systems being misclassified. Using the eigenvalues of a system in its discrete state space form was more successful, giving a “good rule-of-thumb” for predicting stability. In [Abbott and Okamura, 2006b] they developed their discrete formalisation and showed that analysis of the system’s eigenvalues was sufficient for stability investigations, even with a unilateral (i.e. discontinuous) constraint. Based on their discrete state space representation of this system, and a linear time invariant model of the user, they found that they were able to predict experimental stability bounds for k_p with a reasonable consistency. The numerical stability of linear constraints is also discussed in [Joli et al., 2007]. Here, Joli *et al.* define an implicit constraint formulation based on the augmented Lagrangian method which, it is claimed, avoids numerical instabilities.

Derivative terms have been used to extend linear enforcement functions and improve their properties for various applications. If a user moves against a constraint at a higher velocity differential, then it will take a larger force from the controller in order to slow them down and minimise any non-conformance. By introducing a derivative (or viscous) term to the enforcement function, the controller is able to respond to this situation, for example:

$$\mathbf{f}_{pd} = k_p(\mathbf{p}_{const} - \mathbf{p}_{ctg}) + k_d(\dot{\mathbf{p}}_{const} - \dot{\mathbf{p}}_{ctg}), \quad (2.2)$$

where $\mathbf{f}_{pd}$ is the constraint force vector and k_d is a derivative gain. Clearly, if the constraint geometry in this formulation is static (i.e. $\dot{\mathbf{p}}_{const} = 0$), the derivative term will be isotropic so that it opposes motion conforming to the constraint as much as it opposes motion that does not. This may be suitable for some applications [Ammi and Ferreira, 2007; Chong et al., 2001; Ortmaier et al., 2006]; however, if $\mathbf{p}_{const}$ is set to move, along a pathway for example, the robot will be constrained to follow its motion. By considering the rotational penetration of a constraint using ‘eigenscrews’, viscoelastic constraints such as these have been extended into six degrees of freedom (DOF) control [Zhang et al., 2012].

2.6.2 Proxy and Linkage Simulation

Proxy techniques were originally conceived for haptic rendering of surfaces in virtual environments on impedance controlled devices [Ruspini et al., 1997; Zilles and Salisbury, 1995]. At the time, most techniques for surface rendering were proximity based and therefore disregarded the path taken by a

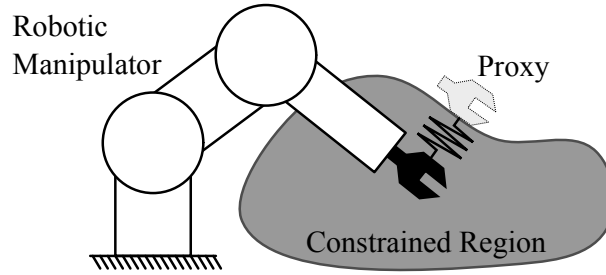


Figure 2.7: Illustration of a proxy-based constraint enforcement where an elastic linkage between a proxy and a robot’s end effector is simulated. The dynamics of the proxy are defined so that it follows the pose of the robot’s end effector while not penetrating the constrained region. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

constrained tool geometry. Often, the path is irrelevant, but when a regional constraint is thin, for example, and the user forces the constrained tool geometry to reach the region’s midpoint, the closest edge will switch to the opposite side and the constraint will act in the opposite direction. There are two potentially detrimental effects arising from this behaviour: it can cause harmful control discontinuities and it will encourage the user to pass completely through a constrained region, rather than restricting them.

In its simplest form, a proxy is an entirely simulated virtual object which can have any dynamic properties required by the controller and constraints. In the simulation, the proxy is virtually attached to the constrained tool geometry of a force reflecting device by some form of virtual linkage (as shown in figure 2.7). The linkage model is typically elastic or viscoelastic, though it can take any form to give the constraint properties desired. If the constrained tool geometry violates a constraint, the proxy will remain on the surface causing them to separate, and the virtual linkage to generate a force. Though proxies are often mentioned within the literature, there have been few significant attempts at utilising them for active constraints in this standard form. Proxies are; however, the foundation of the more complex methods described within this section.

Implicit-force and modified-damping control are two proxy-based algorithms that were formulated by Ho *et al.* for the application of regional constraints to hands-on knee replacement surgery [Ho et al., 1995]. In these approaches, a proportional-derivative position controller (i.e. viscoelastic linkage) is used to drive the constrained tool geometry towards a desired position (i.e. a proxy) which is controlled so as to avoid constraint penetration. Two parallel controllers are used to implement this architecture: one defines the motion of the proxy and the control parameters, and the other computes and applies constraint forces based on the viscoelastic linkage.

The proxy motion controller in this constraint formulation generally acts in the conventional way, whereby the proxy is free to follow the constrained tool geometry in the unconstrained region, but is prevented from following it into the constrained region. Uniquely, this method also considers an additional configuration close to the constraint boundary. In this configuration, the proxy is moved up onto the constraint boundary ensuring that smooth surfaces are produced during contouring cuts. The difference between the ‘implicit-force’ and ‘modified-damping’ variants of this method lies in how the velocity of the proxy is computed. In the ‘implicit-force’ method, the proxy’s motion is quasi-static, such that the derivative term within the control law always opposes constrained tool geometry motion away from the proxy, irrespective of whether the constrained tool geometry is violating a constraint. In the ‘modified-damping’ method, the velocity for the proxy is computed from the force applied by the user to the end effector, in a way similar to that in reference direction fixtures (section 2.6.5). The velocity consideration in this method means that the derivative term is anisotropic and treats motion which either conforms to or violates a constraint differently.

Virtual mechanisms were conceived by Joly and Andriot as a novel method for constraining the motion of a proxy used in teleoperation [Joly and Andriot, 1995]. Their control algorithm operates by simulating the kinematics of an ideal (i.e. inertia free) mechanism, which is attached, via simulated linkages, to both the master and slave device. By selecting an appropriate kinematic structure for the virtual mechanism, the motion of its end effector is restricted to a limited task space, thus constraining the master and slave’s motions via the linkages. Joly and Andriot showed how the kinematics for various functional virtual mechanisms can be constructed; however, the requirement for these kinematic descriptions of constraints can be complex or restrictive. Compound versions of virtual mechanisms were later used to decouple constrained and unconstrained DOFs with greater control [Micaelli et al., 1998]. In this method, secondary virtual mechanisms are used to remove the isotropy of the conventional spring-damper linkages in Joly and Andriot’s method. These secondary mechanisms mean that the dynamics and behaviour of master and slave motion’s in both constrained and unconstrained directions are fully controllable.

Pseudo-admittance control is a method for applying constraints developed for admittance controlled robots (specifically reference direction fixtures, as in section 2.6.5) on impedance controlled teleoperation systems, thus giving quasi-static transparency and hand tremor attenuation [Abbott and Okamura, 2006a]. To achieve this, Abbott and Okamura conceived a proxy based method which uses three control laws to define the forces which should be applied to master and slave devices, and the simulated proxy.

The master forces are based on its separation from the slave device, the slave forces are based on its separation from the proxy and the proxy forces are based on the error between the master and slave. The proxy will respect any admittance constraints applied and as the slave follows the proxy, it will also be constrained. By controlling the velocity of the proxy, Pezzementi, Okamura and Hager were able to use this method for applying active constraints to a robot's velocity rather than just position [Pezzementi et al., 2007].

2.6.3 Non-Energy Storing Constraints

A challenge with many enforcement methods is that the constraints have the effect of storing potential energy when the constrained tool geometry is forced to move against them. Once the energy has been stored, releasing the device sharply can cause unwanted and potentially dangerous motion if not controlled correctly. Another limitation in common proximity based approaches is that they require at least a small amount of constraint violation before any motion regulation is applied.

Simulated plasticity was proposed by Kikuuwe, Takesue and Fujimoto to overcome both of these difficulties with constraints [Kikuuwe et al., 2008]. The property of plasticity is advantageous for active constraints because it involves a stiff initial collision and then, if penetration does occur, the constraint only dissipates, and never stores, energy. A challenge with implementing active constraints which exhibit this behaviour is that it is highly discontinuous when the tool crosses the constraint boundary and this can cause instabilities or tactile inconsistencies in discrete time implementations.

Kikuuwe, Takesue and Fujimoto's solution to the discontinuity problem was to apply the plasticity (modelled as coulomb friction) to a proxy, as discussed in their previous work [Kikuuwe et al., 2006]. They claim that their discretised proxy model can apply non-energy storing static boundary constraints that allow the user to sense boundaries without crossing them. The aim of their research was to show that simulated plasticity constraints could be implemented on both impedance and admittance devices. This means that they did not compare their results with other conventional techniques and quantification of the improvement from plasticity was not given.

Continuous impulsive forces represent a similar approach to the implementation of constraints which imitate hard, inelastic collisions in impedance devices [Hennekens et al., 2008]. The continuous impulsive forces technique is based on the previous use of impulse forces in haptic rendering [Constantinescu et al., 2005]. Impulse forces are applied for a single controller time-step with the aim of eliminating the

constraint-violating component of a constrained tool geometry's motion. The continuous nature of this method comes from the fact that, for each time-step in which the tool is violating a constraint, the controller applies a new impulse force $\mathbf{f}_{CIF}$ to prevent any further incursion. The impulse force is calculated from the equivalent constrained tool geometry mass violating the constraint M_{vc} , the constrained tool geometry velocity component violating the constraint $\dot{\mathbf{p}}_{vc}$ and time-step Δt as follows:

$$\mathbf{f}_{CIF} = \frac{-M_{vc}\dot{\mathbf{p}}_{vc}}{\Delta t}. \quad (2.3)$$

Though it has the effect of storing energy, the implementation in [Hennekens et al., 2008] combines the impulse force with the output from a PD controller to prevent slow 'drift' which impulse forces will not respond to. The authors state that by using this combination of impulse force and PD control, they provide high initial stiffness, reduce incursion magnitude and prevent PD bouncing.

2.6.4 Potential Fields

Potential fields were devised as a method for real-time collision avoidance in robotic systems [Khatib, 1986]. In the initial formulation, each point within a robot's discretised Cartesian workspace is allocated a potential value based on the distribution of obstacles and targets surrounding it (as illustrated in figure 2.8). Areas where the robot is actively encouraged to be (such as targets) are allocated low potential values, and areas which the robot is restricted from (such as obstacles) are allocated high potential values. By calculating the negative gradient of the field, a force vector can be extracted for any robot position, which will point away from obstacles and towards targets. By applying it at the robot's end effector, this force will actively move the robot as required by the obstacles and targets. It is not only workspace obstacles which can be considered by this method, it is also able to incorporate joint limits if a potential field is implemented in joint rather than task space.

It should be noted that the terms 'force fields' and 'repulsion fields' are sometimes used in relation to potential fields. These can be considered as negative gradient representations of potential fields and can be computed directly using the Cartesian partial derivatives. In publications such as [Chong et al., 2001; Kuang et al., 2004], the terms force and repulsion fields are also used to refer to any impedance constraint method where the output is a force vector, irrespective of how the force is computed. To ensure clarity, this convention will be avoided in this survey.

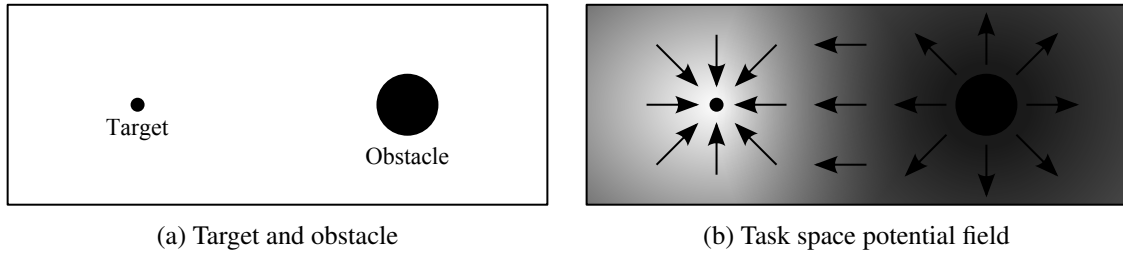


Figure 2.8: Illustration of a task space with a target position and obstacle (a). Also shown is a potential field which could be constructed to constrain this task space (b). In the field, dark colours represent high potential values and light colours represent low potential values. The arrows show the direction of ‘encouraged’ motion. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

Using potential fields for obstacle avoidance in autonomous robots is a well researched area. Aigner and McCarragher extended autonomous, potential field controllers so that a human user was able to guide the manipulator’s motion between obstacles using a non-force reflecting master device [Aigner and McCarragher, 1997]. They proposed that a standard task space potential field should be built for the robot, with attractive ‘wells’ and repulsive ‘peaks’, as necessary for the robot’s environment and task. During operation, the gradient of the potential field was used to compute a robot velocity command which was sent to the slave device. Two methods for combining the user commanded motion with the potential field were proposed. In one approach, the velocity of the master device is added to the field velocity before it is sent to the slave robot. In the other, the motion of the master device is used to translate and scale obstacles within the potential field.

Later research by Turro, Khatib and Coste-Maniere showed how potential field active constraints could be used in teleoperated systems with force reflecting master devices [Turro et al., 2001]. Their method is conceptually similar to the conventional application of potential fields to autonomous robots with the difference being that forces from the potential field are applied on the master device, to regulate the motion of the human user. A position controller on the slave follows the position of the master and therefore the effect of the potential field is on both master and slave. An important development in [Turro et al., 2001] is that not only the end effector of the robot is controlled by the constraint. Each link within the robot is considered in the potential field and the obstacle avoidance forces for each of them are transformed and combined with the force at the end effector before application to the master device. This method ensures that, while the master respects guidance constraints, it does not cause the body of the slave robot to collide with obstacles within its environment. It is this constraint formulation which has subsequently been used in practical constraint applications such as micro-scale telemanipulation

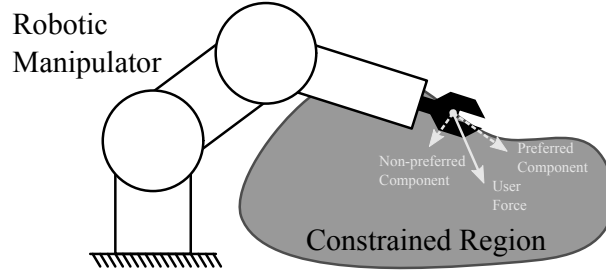


Figure 2.9: Illustration of a reference direction fixture where a user’s force is decomposed into preferred (permitted by the constraint) and non-preferred (resisted by the constraint) components. The admittance controlled motion of the end effector is computed from a weighted sum of the preferred and non-preferred components so that violation of the constrained region can be prevented or minimised. Figure reprinted from [Bowyer et al., 2014]. ©2014 IEEE.

[Amami and Ferreira, 2007] and intracellular injection [Ghanbari et al., 2010].

2.6.5 Reference Direction Fixtures

Active constraints on admittance controlled robots require that the robot is more compliant when moved in a direction which respects a constraint and less compliant when not. Burghart *et al.* proposed an initial approach to generating this compliance anisotropy by geometrically scaling the velocity vectors produced from a linear admittance function [Burghart et al., 1999]. However, the more recent ‘reference direction fixtures’ described by Hager, Bettini and Marayong *et al.* [Bettini et al., 2001, 2004; Hager, 2002; Marayong et al., 2003] offer a more systematic formalisation and are widely used within the literature. It is this approach that will be focused upon in this section.

In reference direction fixtures, the scalar admittance gain of Burghart *et al.* is replaced with a compliance matrix which accedes to the user applied force in an anisotropic way. The key component of this approach is the method employed for specifying the compliance matrix is based on the concept of ‘reference directions’. For a given time and position, the entire space of possible directions of motion is divided up into two complementary subspaces: ‘preferred’ (i.e. those which correspond to a direction permitted by the active constraints) and ‘non-preferred’ (i.e. those which correspond to a direction restricted by the active constraints). An example of the subspace decomposition used in reference direction fixtures is shown in figure 2.9.

The preferred directions can be computed in a variety of different ways, allowing different constraints to be represented. In their general form they are represented as a $6 \times n$ matrix $\mathbf{D}$, defining n linearly

independent directions in which motion is preferred. Taking the span $[\mathbf{D}]$ and the kernel $\langle \mathbf{D} \rangle$ projections of $\mathbf{D}$, any user applied force vector $\mathbf{f}$ can be decomposed into independent components in preferred and non-preferred directions. The span and kernel can be weighted and combined to construct an anisotropic compliance matrix that can convert user forces $\mathbf{f}$ into constrained velocities $\dot{\mathbf{p}}_{rdf}$:

$$\dot{\mathbf{p}}_{rdf} = c([\mathbf{D}] + c_\tau \langle \mathbf{D} \rangle) \mathbf{f}, \quad (2.4)$$

where c is the overall system compliance and c_τ is the compliance in non-preferred directions.

By varying c_τ between 0 and 1, the compliance matrix can be tailored so that non-preferred directions are non-compliant ($c_\tau = 0$), fully compliant ($c_\tau = 1$) or partially compliant to any degree in-between. The partial compliance for non-preferred motion means that admittance active constraints can be constructed which can be overpowered if the user requires.

Guidance constraints are implemented on reference direction fixtures by functionally generating the preferred direction matrix $\mathbf{D}$ based on the relative constraint-robot configuration. For positioning constraints, a single task space vector can be used which points from the current to the desired positions. This method has been used for both 3 DOF [Bettini et al., 2001] and 6 DOF [Marayong et al., 2003] manipulators; however, some research has shown that when the robot is close to a target, it can become erratic or unstable due to discretisation and noise causing large discontinuities in $\mathbf{D}$. Several solutions were proposed in [Bettini et al., 2004], but autonomous fine positioning and fixing the constraint direction were found to be the most successful. In [Prada and Payandeh, 2009], this problem was addressed for impedance devices, where it was proposed that a damping factor could be introduced when the tool becomes close to the target. This could equally be extended to admittance controllers by scaling the compliance matrix as a function of the constrained tool geometry velocity.

Path following constraints can be implemented in this framework by setting $\mathbf{D}$ as a single vector which is tangential to the closest point on the path. However, if the constrained tool geometry does move away from the path, the controller will make no distinction and the motion will be constrained to a parallel pathway [Bettini et al., 2001]. To overcome this, positional feedback can be incorporated into $\mathbf{D}$, so that rather than only directing tangentially to the path, it includes the vector between the constrained tool geometry and the closest point on the pathway $\mathbf{u}$ as a correction component [Marayong et al., 2003]:

$$\mathbf{D}_c = (1 - k_d)[\mathbf{D}]\mathbf{f} + k_d\|\mathbf{f}\|\langle \mathbf{D} \rangle \mathbf{u}, \quad (2.5)$$

where $\mathbf{D}_c$ is the modified preferred direction matrix and k_d is a constant which controls the degree of offset compensation.

For applications where the path does not need to be followed precisely, the constant motion regulation can be intrusive. Bettini *et al.* created a method whereby a 3-D ‘tube’ is created around the path, inside which the end effector’s motion is unregulated [Bettini et al., 2004]. Here, conical constraints are also proposed for relaxing the approach on point positioning constraints.

Regional constraints based on reference direction fixtures have not been explicitly discussed within the literature; however, computing $\mathbf{D}$ based on surface collisions is not believed to be overly complex. The initial paper by Burghart *et al.* [Burghart et al., 1999] and subsequent papers such as [Matinfar et al., 2007; Xia et al., 2008] all enforce regional admittance constraints by selectively scaling the robot velocity based on its direction and proximity with respect to regional boundaries. The effect of these methods is the same, but their formulation is less general.

Autonomous error compensation was used by Castillo-Cruces and Wahrburg to overcome human difficulties in simultaneously controlling position and orientation of a 6 DOF robot under reference direction fixtures [Castillo-Cruces and Wahrburg, 2010]. They found that, in a system with both position and orientation reference direction fixtures, only position or orientation would be effectively constrained at any one time. Castillo-Cruces and Wahrburg stated that this is due to the translational and rotational components of motion being decoupled from each other in such a way that the user, focusing on moving one, will not notice an error in the other.

To ensure that errors are always corrected, they modified equation (2.4) to include a proportional controller with gain k_v which autonomously compensates for error $\mathbf{e}$:

$$\dot{\mathbf{p}}_{aec} = c([\mathbf{D}] + c_\tau \langle \mathbf{D} \rangle) \mathbf{f} + k_v \mathbf{e}, \quad (2.6)$$

where $\dot{\mathbf{p}}_{aec}$ is the constrained and autonomous error compensated velocity. This formulation was found to reduce off-motion-axis error, though the cost of this increased accuracy is that a component of the robot’s motion is no longer fully controlled by the human user, which is somewhat inconsistent with the conventional definition of active constraints.

2.6.6 Constrained Joint Optimisation

Constrained quadratic optimisation of robot joint velocities was one of the earliest active constraint enforcement strategies proposed in the literature. Unlike the majority, this method considers the robot in joint space and can therefore efficiently constrain the motion of individual links as well as the end effector. Funda *et al.* [Funda et al., 1996] demonstrated the application of this method for the control of a teleoperated robotic endoscope where: the end effector position was constrained to prevent tissue damage; the endoscope focal position was constrained to a location controlled by the user and the robot joints were constrained to be within their mechanical limits. Expressing these three constraints as algebraic inequalities and combining them into a single matrix expression allows them to be solved numerically for joint displacements.

There can potentially be many joint displacements which are considered acceptable by the set of constraints above, some of which may be ‘more suitable’ than others. Funda *et al.* establish the best solution by minimising: the rotational error of the endoscope focal position; the end effector displacement and the joint displacements. This is equivalent to the equation shown below:

$$\Delta \mathbf{q}^* = \arg \min_{\Delta \mathbf{q} \in Q(\mathbf{q})} \sum_{i=1}^N C_i(\mathbf{q}, \Delta \mathbf{q}), \quad (2.7)$$

where $\Delta \mathbf{q}^*$ is a vector containing the optimised displacement at each joint; N is the number of controllable joint frames; $\mathbf{q}$ is a vector containing the current joint positions; $\Delta \mathbf{q}$ is a vector containing joint displacements; $C_i(\cdot)$ is a weighted objective cost function for joint i based on the three criteria above and $Q(\mathbf{q})$ is the set of constraint conforming joint displacements in the current position. By representing equation (2.7) as a linear least squares problem, the value of $\mathbf{q}$ is iteratively optimised using Lawson and Hanson’s method [Lawson and Hanson, 1995].

Enforcing constraints with this technique ensures that commands are never sent to the robot which will cause it to move into a constrained configuration. This is a problem which will reduce with developments in computing power; however, the limitation of this method is that the iterative optimisation algorithm is computationally demanding and inconsistent for some constraint and cost functions. In addition, the master device which commands the focal position is non-force reflecting, and therefore only visual feedback can be relied upon to inform the user about the constraint interactions.

Haptic constraint feedback can be incorporated into a constrained joint optimisation controller by

implementing it on a hands-on, as opposed to teleoperated, robot. Under hands-on control, when the constrained joint optimisation prevents the manipulator from moving in a certain direction, the user holding the robot will directly feel this. As the robot under constrained joint optimisation must be admittance controlled, a force sensor is required on a handle at the end effector to interpret the user's desired motion. Li *et al.* used such haptic constraints with a 7 DOF, admittance controlled, robot for simulated endoscopic sinus surgery [Li and Taylor, 2003; Li et al., 2007].

The sinus geometry considered by Li *et al.* is complex and therefore a covariance tree (as described in section 2.5.1) was preoperatively constructed from it, to allow efficient identification of when tool-tissue collisions were impending, for the application of constraints. Intraoperatively, a linear admittance control law was used to compute the required constrained tool geometry velocity and the covariance tree was used to identify when and where the constrained tool geometry is closer than permitted to the tissue. Based on these pieces of information, the constrained optimisation controller utilised terms to prevent tissue penetration and minimise the discrepancy between the user desired and the actual constrained tool geometry velocity. By using their iTaSC constrained optimisation controller, Borghesan, Willaert and De Schutter applied similar optimisation based active constraints to a PR2 robot [Borghesan et al., 2012].

In [Li and Taylor, 2005], a comparison was made between applications of constrained joint optimisation using teleoperation (i.e. a form of shared control without haptic feedback to the user) and hands-on (i.e. the conventional form of active constraints with haptic feedback to augment the user's performance) control. It was found that with haptic feedback the tasks were faster and more intuitive than without.

Guidance constraints can be incorporated into the constrained joint optimisation by including additional constraint inequalities [Kapoor et al., 2005; Li et al., 2005], or by modifying the admittance control law to incorporate reference direction fixtures, as described previously [Li et al., 2007].

2.6.7 Passive Constraint Enforcing Mechanisms

Constraints on active robots have been the main focus of this survey; however, several groups have researched passive devices which use physical mechanisms to constrain the motion of an end effector. The core concept in these devices is that they are hands-on and their primary motive force comes directly from a human user. In each case, the control architecture is only able to limit or redirect the motion. The main advantage cited for passive mechanisms is that they are safer because errors or malfunctions are

much less likely to cause harmful movements.

Mechanical brakes were used by Taylor *et al.* to apply ‘passive manipulation aids’ as a method for improving the placement accuracy of bone segments during craniofacial osteotomies [Taylor et al., 1991a, 1992]. Passive manipulation aids were implemented on a mechanical device consisting of six orthogonally decoupled, passive axes with a brake attached to each. By actuating each brake when the associated manipulator axis was correctly aligned with the target, the task of manual placement could be considered one axis at a time by the surgeon. Sub-millimetre/degree accuracy was achieved in this way.

Mechanical clutches have also been used to passively enforce active constraints. ‘Passive Arm with Dynamic Constraints’ (PADyC) is a ‘mechanical guiding system’ which can enforce both regional and guidance constraints [Troccaz and Delnondedieu, 1996]. It consists of modular mechanisms constructed from clutched, contra-rotating freewheels which allow the controller to place bounds on the angular velocity of each joint. Based on the relative configuration of the end effector position and the constraint geometry in joint space, PADyC’s controller identifies what angle each joint can move through before it will violate a constraint or pass a target point. Using these joint displacement limits, the angular velocity bounds can be set for each joint so that during the following time step the end effector will respect all constraints. Rendering constraint profiles which are not parallel to the axes within PADyC’s joint space is a significant challenge, as it is with most brake actuated mechanisms. When moving along constraints that are not parallel to joint space axes it is not the absolute change in each joint which must be limited, rather it is the ratio between joint velocities which must be controlled. As a result, surfaces can be rough, frictional or possibly even unenforceable by PADyC.

Both brakes and clutches are combined to apply guidance constraints in the ‘Passive Trajectory Enhancing Robot’ (P-TER) [Davis and Book, 1997]. P-TER was designed with brakes on each joint within the mechanism and clutches between the joints. The intention was that clutches can enforce relative velocity constraints between joints so that smooth guidance pathways could be enforced.

Continuously variable transmissions (CVT) were used within ‘Cobots’ to constrain the relative joint velocities in a passive mechanism [Gillespie et al., 2001; Peshkin et al., 2001]. Mechanisms based on brakes and clutches all dissipate the energy that users introduce to the system and therefore constraints cannot be applied which are both smooth and stiff. CVTs are used within a Cobot to link the angular velocity of all joints to the angular velocity of all others such that at any one moment, the chain of joints and CVTs limits the end effector to a single DOF. By varying the gear ratios of the CVTs, a Cobot

controller can orient the single free DOF so that the user follows a smooth pathway. Cobots approach the problem of constraining motion from the opposite perspective to most methods covered in this survey. Rather than having a free device, which when interacting with a constraint, has some DOFs restricted, a Cobot is a constrained device, which when not interacting with a constraint, has some DOFs simulated. The DOF simulation is achieved using a force sensor mounted on the end effector. The controller uses the force sensor to interpret the intended direction of motion and, if this direction is permitted by the constraints, the CVTs are reconfigured to align the single free DOF with this direction. By attaching a motor to the CVT chain, so-called ‘Intelligent Assist Devices’ can be created, though these are no longer passive [Colgate et al., 2003]. These devices actively assist a user working with heavy payloads, while still allowing for the enforcement of constraints.

2.7 Advanced Constraint Extensions

The three previous sections describe the core concepts behind the most common active constraint approaches within the literature. Several research groups have built upon these fundamental elements to create advanced active constraints which overcome specific limitations, or assist in more complex tasks. An overview of such ‘advanced constraint extensions’ found within the literature is given in this section. In each case considered, the research does not involve an entirely new control method, rather, each is based upon one or more of the methods described in the preceding sections. These advanced constraint extensions have been separated from the methods upon which they are based, because they can be considered as higher level concepts which have the potential to be applied across a wider range of methods and applications.

2.7.1 Adaptive Constraints

Various methods have been proposed in the literature for applying active constraints in an adaptive manner. In each of these methods, the constraint is enforced using one of the previously discussed methods; however, the controller decides when and how to apply the constraints based on some knowledge of the task, hardware or users.

Selective application of active constraints has been achieved by several groups using hidden Markov models (HMM). By using HMMs to classify and interpret user commands, these groups are able to

adaptively apply active constraints to situations which have not been explicitly programmed. In [Aarno et al., 2005], complex high level tasks are broken down into simple subtasks, each of which has a straight line guidance constraint of its own. Here, the current state is identified using a HMM, so that the controller can apply a guidance constraint appropriate to the subtask associated with it. Yu *et al.* applied a similar approach based on subtasks, but rather than just straight lines, they also considered curved constraints for positioning and obstacle avoidance [Yu et al., 2005]. Procedure specific adaptive constraints are shown in [Weede et al., 2011] where a HMM based controller is trained on expert procedures from which it learns tool positions which should be taken during the procedure. Guidance constraints are then implemented by clustering the tool positions into discrete states and constructing a HMM for transitions between them. Based on the possible state transitions at any moment, the constraint guides the user towards the next state which was learnt from the expert. In addition the literature using HMMs, in [Chowriappa et al., 2013], a trocar insertion pathway is encoded within a Gaussian mixture model built from expert insertions. Similarly to the HMM methods, the encoded pathway is then retrieved and used enforce constraints which can guide novices.

When the user needs to depart from a constraint due to unpredicted obstacles or unplanned targets, it can be problematic with many constraint implementations. If, for example, a particularly stiff guidance constraint is in use, and a physical obstacle blocks the robot's path, the robot can become immobilised. In [Marayong and Okamura, 2004] a process for selecting the optimal constraint compliance value with respect to this problem is reported. The method described is predicated on finding a compromise between the level of permissible, accidental violation and the need for freedom of intentional violation. For some applications it might not be possible to make concessions on either of these aspects. In [Nolin et al., 2003], a solution is proposed. In this work, three methods for scaling constraint effects ('toggle', 'fade' and 'hold') and three methods for deciding when to apply the scaling ('explicit', 'implicit' and 'autonomous') were studied. In [Kragic et al., 2005; Li and Okamura, 2003], HMMs are used to identify when a user was consciously intending to leave a guidance constraint as a result of an obstacle blocking the path. Once a user's motion was classified as intentionally violating the constraint, the anisotropy on the reference direction fixture was removed and user motion was smoother and faster as a result. Passenberg *et al.* used 'interaction forces' between the user and robot to achieve a similar autonomous response [Passenberg et al., 2011]. By monitoring how much 'agreement' there is between the user and robot they classify intentional violation cases and react accordingly.

2.7.2 Compliance in Tools and Robot Mechanisms

Compliance within a robot can reduce the positional accuracy of active constraints because the constrained tool geometry will not be in the exact position reported by the robot joint sensors. Beasley and Howe compared the effect of tool tip tracking and flexion modelling for improving the accuracy of long, thin, surgical instruments under regional active constraints [Beasley and Howe, 2009]. They found that incorporating the Euler-Bernoulli beam bending model could significantly reduce maximum constraint penetration over a rigid tool assumption. It was also found that feeding back the tracked tool tip could significantly reduce the maximum constraint penetration further.

Marayong *et al.* investigated potential methods which could be used to accommodate for admittance controlled compliant robot mechanisms. They initially proposed an open-loop control architecture based on a model of human hand dynamics under ‘equilibrium-point control’ [Marayong et al., 2006]. Using a unified mass-spring-damper model of the hand, robot and constraint, they were able to adaptively position the constraint so that the user’s hand would be stopped before it penetrated a restricted region, irrespective of compliance between the robot actuators and the hand. The method was successful at preventing region penetrations; however, it was conservative and generally stopped the user’s hand well before the region. They later implemented a closed loop controller using a camera to measure and feedback the deviation between the rigid kinematic and actual end effector positions [Marayong et al., 2008]. By interpolating between low frequency images with a Kalman filter, Marayong *et al.* were able to incorporate a linear dynamic model of the manipulator compliance in their controller so that the actual end effector position could be accurately constrained.

2.7.3 Dynamic Active Constraints

Dynamic active constraints are those where the constraint geometry moves continuously, as a result of changes in the physical environment or task being undertaken.

An abstract investigation of dynamic active constraints was carried out using simple proximity based constraints (as discussed in section 2.6.1) by Gibo *et al.* [Gibo et al., 2009]. They constructed an experiment where a linear actuator was used to move a soft tissue phantom in 1 DOF while a human user attempted to depress the phantom by a fixed amount using a teleoperated robot. By constructing a regional dynamic active constraint which followed the tissue phantom at the required tissue depth, the user

was assisted while the tissue was moved periodically and randomly. Gibo *et al.* used two methods for computing the necessary position of the dynamic constraint; one based on the current tissue position and one based on its predicted position. They found that the two methods were comparable and both were significant improvements on unconstrained teleoperation and static constraints.

Much of the remaining dynamic active constraint research has focused on applications for surgery on a beating heart. Navkar *et al.* considered the heart's left ventricle and generated multi-dimensional dynamic active constraints based on a simple proximity function [Navkar et al., 2013, 2012]. They constrained the end effector of a simulated surgical tool to follow a 'dynamic guidance curve' between the inner walls of a beating heart which were generated in real-time. They used a haptic master device to render a constraint force to the user. The results of experiments with this system show that off-path error was reduced compared with cases with only visual guidance and cases without any guidance.

Potential fields were used by Ren *et al.* to constrain a simulated surgical tool to the heart's external surface [Ren et al., 2008]. In their method, a series of preoperative images are taken of the heart throughout its pulsatile cycle. Each 3-D time-slice is segmented to extract the geometry of the surface of the heart, and image dilation is used to create an explicit distance map from the heart's surface. The distance map is then used to train a B-spline-based artificial neural network, as described previously, allowing for highly efficient real-time evaluation. Intraoperatively, the time-slices are synchronised with the patient using electrocardiography and the controller uses the neural network to efficiently establish the distance of the constrained tool geometry from the closest point on the heart's surface. Values for the potential are computed from the proximity values using either a generalised sigmoid function (for regional constraints) or a Gaussian function (for guidance constraints), allowing the constraint forces to be computed in the standard way.

The preoperative acquisition and processing of medical images within the above method means that it is unable to adapt to intraoperative changes in the heart's motion. By acquiring a real-time model of a beating heart using a range camera, Rydén and Chizeck are able to overcome this limitation [Rydén and Chizeck, 2012, 2013]. By considering each point within a point cloud as an individual spherical constraint, Rydén and Chizeck's method creates implicit surfaces which the constrained tool cannot penetrate.

A hyper-redundant 'snake' robot was dynamically constrained using joint optimisation for camera positioning by Kwok *et al.* in cardiac surgery [Kwok et al., 2010] and colonoscopy [Kwok et al., 2013],

and by Vitiello *et al.* in liver surgery [Vitiello et al., 2011]. They defined a method for creating tubular dynamic active constraint pathways by placing cubic Hermite spline control points, which move relative to a tissue's deforming surface. By creating an exponential cost function based on the robot body's infractions outside of the tubular constraint, joint optimisation was used to position the body of the snake, while the camera was focused on the user defined target. They also showed how GPU acceleration could allow for complex proximity queries between dynamic surfaces in real-time.

Research into robotic motion compensation [Hagn et al., 2010] is closely related to the research in dynamic active constraints for teleoperation. In this field, robotic control is used which automatically tracks a deforming surface such that the human user at the master console interacts with what appears to be a static environment. In [Moustris et al., 2013] it is shown how a form of dynamic active constraint can be enforced by implementing static active constraints in the motion compensated space of a deforming environment. The authors did not make a direct comparison with a conventional dynamic active constraint; however, they did show that the static constraint in the motion compensated environment improved the users accuracy over motion compensation on its own.

2.7.4 Multi-Handed Active Constraints

When multiple manipulators are operating within the same workspace, multi-handed active constraints can be used [Kapoor and Taylor, 2008]. The challenge for multi-handed tasks is in enforcing temporally and spatially dependent relationships between the manipulators while still ensuring that each one conforms to its individual environmental and physical constraints. The method in [Kapoor and Taylor, 2008] uses constrained joint optimisation with a supervisory optimisation function which enforces inter-robot constraints while seeking to minimise the weighted sum of the cost functions for each individual robot. This method was used for a simplified knot positioning task, where two robots held opposite ends of the thread. This application is a good demonstration of multi-robot constraints because the relative angle and velocity with which the robots pull on their ends of the thread determines how close the knot is to the desired position. An interesting effect of these constraints is that by removing the user guidance from one of the robots, it essentially becomes autonomous and follows the constrained optimum position based on the position of the other. In [Xia et al., 2011], it is shown how this method could be implemented on a commercial teleoperation system for surgery.

2.7.5 Active Constraints on Handheld Manipulators

It is shown by Becker *et al.* that constraints can be applied via handheld manipulators which are not physically attached to the ground in the conventional way [Becker et al., 2011]. In this research, a handheld micromanipulator is used, where actuators are located between the user's hand and the tool-tip, such that the controller can compensate for unintentional motion like hand tremor. Visually tracking the micromanipulator, Becker *et al.* computed the relative position of the tool-tip to a constraint geometry. Using this relationship with an admittance control law similar to the reference direction fixtures above, they could use the handheld actuators to minimise the error from pre-planned positions and pathways. Subsequent research by the same group demonstrates how a vision system can be used to estimate the surface geometry of a retinal membrane allowing dynamic active constraints which prevent over penetration during a surgical procedure [Becker et al., 2012, 2013].

A limitation of the handheld manipulator of Becker *et al.* is that when the constraints are being enforced, the user does not receive any haptic information to inform them of the nature of the constraint. In [Payne et al., 2013], a 1 DOF handheld manipulator was constructed which included a linear force-feedback display which pushes upon the user when they interact with a constraint. In a user study, it was shown that this simple force feedback successfully indicated to the user the nature of the constraint interaction.

2.8 Discussion

2.8.1 Literature Overview

Active constraints and virtual fixtures are robot control strategies that, through physical human-robot interaction, assist users performing manipulation tasks. The assistance is achieved by anisotropically regulating the user's motion, such that deviations from a plan are attenuated and conformance to a plan is encouraged. This chapter has presented the current state of published literature regarding these synonymous methods.

To portray the wide range of research in a systematic way, a generalised active constraint framework has been proposed which consists of three primary components. Initially, in a process called 'constraint definition', the geometries which constitute the constraints are input to the system and are represented computationally. At each time step during operation, the relative configuration of the constraint geom-

entry and the robot's current location is computed. This stage is termed 'constraint evaluation'. Finally, 'constraint enforcement' is a process whereby the relative robot-constraint configuration is converted into commands that can be sent to the robot hardware to enact the constraint. Though all of these elements are required for a complete implementation, each publication tends to focus on certain elements. A breakdown of the contribution to methods for each of these stages from the literature is given in table 2.1.

The procedures used to define constraints are specific to each application, and therefore this aspect has not been discussed in detail. However, the methods available to computationally represent constraint geometries are universally applicable and should be considered prior to constructing an active constraint system. Ten constraint representation methods have been found within the literature, ranging from simple points and lines, up to meshes and point clouds. At present, the simple methods are far more widespread within the literature than the complex ones (approximately 85% of references in table 2.1). By applying them creatively, it has been shown how simple constraint representations can be used for a range of practical tasks. However, they are inevitably less flexible than those which are more complex, and this can lead to difficulties in unstructured environments. It is likely that the pervasiveness of simple constraint representations is due to the focus within the literature on methods for constraint enforcement. Using simple constraint representations for abstract tasks facilitates the unbiased validation of constraint enforcement methods. Additionally, when complex constraint representations are used, the subsequent step of constraint evaluation also becomes more demanding.

For constraint evaluation, the process of taking a robot pose and a constraint geometry and computing their relative configuration, methods involve a similar range of complexity. Simple methods which rely on linear algebra, look-up tables or exhaustive searches are also much more common within the literature than those which involve geometric algorithms for collision detection and closest point searches (approximately 90% of references in table 2.1). The choice of constraint evaluation method is based upon the geometric properties of the chosen constraint representation and methods available from within the computer graphics literature.

The published approaches to enforcing constraints are largely independent of the methods selected for the preceding stages of constraint representation and evaluation. In general, constraint enforcement algorithms consider the relative robot-constraint configuration as an input parameter, and are indifferent to how it was produced. Nonetheless, constraint enforcement is intrinsically linked to the robot hardware through which constraints will be applied, and therefore this aspect requires consideration. The

Reference	Points	Representation								Evaluation		Enforcement																	
		Lines	Parametric Curves	Planar	Parametric Surface	Polygonal Mesh	Point Cloud	Volumetric Primitives	Explicitly Described	Artificial Neural Network	Closed Form	Explicit	Hierarchical Representations	Feature Tracking	Simple Functions of Proximity	Proxy and Linkage	Non-Energy Storing	Potential Fields	Reference Direction Fixtures	Constrained Joint Optimisation	Passive Mechanisms	Admittance	Impedance	Hands-on	Teleoperated	Regional	Guidance	Static	Dynamic
Aarno et al. [2005]	•										•										•								
Abbott and Okamura [2003a]					•						•				•							•							
Abbott and Okamura [2006b]					•						•				•							•							
Aigner and McCarragher [1997]								•	•																				
Aleotti et al. [2005]						•								•										•					
Ammi and Ferreira [2007]				•							•												•						
Arata et al. [2010]								•			•												•						
Beasley and Howe [2009]				•							•											•							
Becker et al. [2011]	•		•		•						•											•							
Becker et al. [2013]			•								•												•						
Bettini et al. [2001]	•		•								•											•							
Bettini et al. [2004]	•		•					•			•											•							
Bowyer and Rodríguez y Baena [2013]			•								•						•					•						•	
Burschka et al. [2005]					•						•												•						
Davis and Book [1997]			•								•												•						
Borghesan et al. [2012]				•							•								•				•						
Burghart et al. [1999]		•						•			•												•						
Castillo-Cruces and Wahrburg [2010]			•								•				•							•							
Chong et al. [2001]	•										•				•								•						
Funda et al. [1996]				•							•												•						
Ghanbari et al. [2010]					•			•			•						•						•						
Gibo et al. [2009]				•							•												•						
Gillespie et al. [2001]; Peshkin et al. [2001]				•							•												•						
Hager [2002]	•										•												•						
Hennekens et al. [2008]				•							•												•						
Ho et al. [1994, 1995]			•					•			•												•						
Jang et al. [2012]					•						•												•						
Joli et al. [2007]	•	•									•												•						
Joly and Andriot [1995]; Micaelli et al. [1998]	•	•		•							•												•						
Kapoor et al. [2005]	•	•									•												•						
Kapoor et al. [2006]	•	•		•							•												•						
Kapoor and Taylor [2008]	•	•		•							•												•						
Kikuuwe et al. [2008]	•	•		•				•			•												•						
Kosuge et al. [1995]	•										•												•						
Kragic et al. [2005]	•			•							•												•						
Kuang et al. [2004]	•										•												•						
Kumar et al. [2003]	•							•			•												•						
Kwok et al. [2010, 2013]; Vitiello et al. [2011]				•				•			•												•						
Li and Taylor [2003]; Li et al. [2007]					•						•												•						
Li and Okamura [2003]					•						•												•						
Li et al. [2005]	•		•								•												•						
Li and Taylor [2005]			•		•						•												•						
Lopez et al. [2013]				•							•												•						
Marayong and Okamura [2004]				•							•												•						
Marayong et al. [2003]				•							•												•						
Marayong et al. [2006]				•							•												•						
Marayong et al. [2008]				•							•												•						
Matinfar et al. [2007]				•							•												•						
Moustris et al. [2013]				•							•												•						
Navkar et al. [2013]				•							•												•						
Nolin et al. [2003]	•	•		•							•												•						
Ortmaier et al. [2006]				•							•												•						
Park et al. [2001]				•							•												•						
Park et al. [2011]				•							•				•								•						
Passenberg et al. [2011]		•									•												•						
Payandeh and Stanisic [2002]								•			•												•						
Payne et al. [2013]				•	•						•												•						
Pezementi et al. [2007]				•							•												•						
Prada and Payandeh [2005, 2009]	•							•			•												•						
Ren et al. [2008]			•							•													•						
Rosenberg [1992, 1993]				•							•												•						
Rydén and Chizeck [2012, 2013]				•				•			•												•						
Sayers et al. [1992]	•	•		•							•												•						
Seung et al. [2010]				•							•												•						
Srimathveeravalli et al. [2007]		•									•												•						
Tang et al. [2013]								•			•												•						
Taylor et al. [1991a]	•							•			•							•	•				•						
Troccaz and Delnondedieu [1996]	•		•		•			•			•						•	•					•						
Troccaz et al. [1998]											•							•	•				•					•	
Turro et al. [2001]								•			•												•						
Unruh et al. [1993]	•							•			•												•						
Weede et al. [2011]				•		•					•											•							
Xia et al. [2008]		•			•						•							•					•						
Xia et al. [2011]								•			•									•			•						
Yamamoto et al. [2012]								•			•												•					•	
Yu et al. [2005]								•			•												•					•	
Zhang et al. [2012]			•								•												•						
Total occurrences (only •)	17	10	29	26	4	6	3	19	4	1	58	4	7	1	28	7	3	8	26	14	6	28	39	37	36	58	53	75	11

Table 2.1: Tabulated summary of research contributions from the key literature within this survey (86 papers). For a given paper, the symbol ‘•’ identifies constraint methods which are an appreciable component of the work, and the symbol ‘◦’ identifies methods which are used or discussed, but are of less significance. Table adapted from [Bowyer et al., 2014]. ©2014 IEEE.

main factor dictating potential enforcement methods is whether the constrained device is impedance controlled, admittance controlled or constructed with some other mechanism. Some methods within the literature have embodiments for more than one of these options; however, most are specialised. Though constraints on all hardware have the same overall goal, the haptic properties of each are different, and they interact with users in a different way. Admittance controlled devices, for example, can implement very stiff constraints which never permit access to constrained regions; however, the mechanism's rigidity can stifle any tactile feedback from the environment.

In addition to the generalised active constraint framework, this survey has detailed several advanced extensions that expand the core active constraint concepts to overcome limitations or increase functionality.

2.8.2 Research Trends and Directions

Active constraints, virtual fixtures and several other similar robot control strategies were first developed during the early 1990s. Rather than replace a human operator with an autonomous robot, these strategies sought to augment the human by anisotropically regulating their motion, so as to prevent access to certain regions within a robot's environment, or encourage conformance to pre-produced plans.

The primary benefit of collaborative robot control strategies, such as active constraints, is that as a human remains within the control loop, they can safely cope with more complex and unstructured tasks than would be possible under full automation. Nevertheless, in the early embodiments of the concept, conveniently well structured tasks were considered. Indeed, in the first research on virtual fixtures, the forces were not generated through software; rather they were the result of the master device physically touching a 'fixture board'. At the initiation of a concept, comparatively simple applications will be considered; however, there were several limitations at the time making wider application more difficult. For example, custom robotic hardware was used in most of the early research and complete controllers had to be created. Additionally, certain components of an active constraint controller, specifically constraint evaluation, can involve computationally non-trivial algorithms (often $O(n^2)$) which, at the time, would have been very unlikely to run sufficiently fast for online use.

Since this early research, many of the barriers to the wider active constraint use have been overcome. Robots suitable for physical human robot interaction, that can operate under Cartesian control 'out of the box', can now be purchased. Computational power has also drastically increased since the 1990s

and most algorithms pertinent to active constraints can readily be run at control rates over 1,000 Hz.

Due to these, and other, background developments, active constraint research has grown significantly since its conception. There are currently well over 200 publications directly discussing active constraints and virtual fixtures, with a diverse range of methods and applications considered. It is shown within the literature that constraint geometries can be computationally represented in a variety of ways and they are no longer tied to simple, inflexible shapes. The use of flexible structures such as meshes and point clouds allows the constraint generator, human or otherwise, to be much more expressive, making constraints of greater use during a task.

The fast computation of relative robot-constraint configurations is an important factor for robust and stable control of a constrained robot. In practice, the choice of a method for constraint representation is dependent on the availability of fast computational methods for evaluating it, and it is likely to have been this problem that limited early research. Modern publications have shown that collision detection algorithms for the most complex of freeform constraint surfaces can operate fast enough for stable control; however, this problem has generally been disregarded. The field of computer graphics contains numerous methods for collision detection and proximity queries, therefore, if required for an active constraint implementation, this field could provide a solution.

The key component of an active constraint controller takes the relative robot-constraint configuration and computes a set of robot commands that enforce the required constraint. This aspect has been the primary focus of publications over the years, and seven core methods for enforcing active constraints have been proposed. These methods, along with their extensions and adaptations, provide a range of functional and haptic properties for constraint implementations.

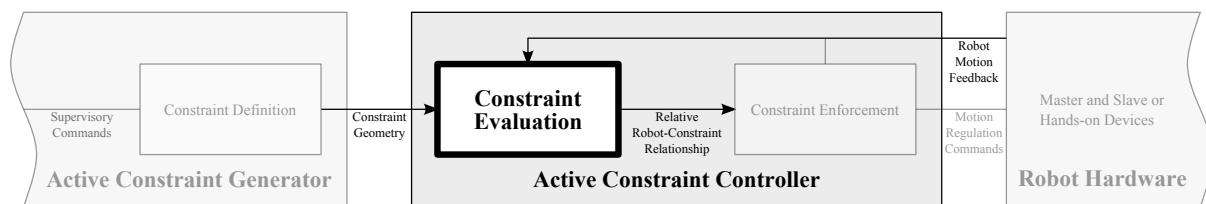
By combining these options for constraint enforcement with those for representation and evaluation, modern active constraints can incorporate many assistive properties, making them useful in a variety of practical tasks. One area where active constraints have seen significant development is in computer-assisted surgery, notably with the commercial development of the hands-on Sculptor Robotic Guidance Arm (formerly of Stanmore Implants Worldwide Ltd.) and the RIO Robotic Arm Interactive Orthopedic System (MAKO Surgical Corp.). In these embodiments, not only do active constraints supply the ability to constrain cuts for accurate sculpted surfaces, but they can also physically constrain surgeons to avoid cutting into critical areas that need protecting. In this way, active constraints provide additional safety in complex surgery.

2.9 Summary

Within this chapter, research into active constraints, across a wide range of clinical and non-clinical applications, has been surveyed. The key terminology used within the literature has been defined, and a generalised active constraint framework has been identified and used to present the published research. Within the subsequent chapters of this thesis, several research areas pertinent to the application of active constraints to soft tissue surgery are pursued, based on the opportunities identified in this chapter. In all of these chapters, the generalised active constraint framework proposed here will be used to introduce the individual contributions. In chapter 3, the specific challenges associated with constraint evaluation in a dynamic active constraint controller is investigated. The enforcement of dynamic active constraints is considered in three and six dimensions in chapters 4 and 5 respectively. In chapter 7, a complete active constraint controller is then validated which contains both elements.

Chapter 3

Deformation Invariant Bounding Sphere Hierarchies



In this chapter, a computationally efficient method for establishing the relative configuration between a constrained tool and a constraint geometry, constraint enforcement, has been proposed. The newly conceived bounding volume can be used within a conventional bounding volume hierarchy and exploits the coherence of soft tissue deformation to preclude the necessity for computationally demanding hierarchy updates, allowing the application of high resolution active constraints to maximise surgeon assistance. The research presented within this chapter is an edited version of research previously published in:

S. A. Bowyer, and F. Rodriguez y Baena, “Deformation invariant bounding spheres for dynamic active constraints in surgery”, in Proceedings of the Institute of Mechanical Engineers, Part H: Journal of Engineering in Medicine, vol. 228, no. 4, pp. 350-361, April 2014.

3.1 Introduction

The geometries which are used to represent the anatomical pathways or regions within an active constraint must be defined in a discrete computational structure. As shown in section 2.4, the simplest, and most widely used, constraint representations are simple geometries such as lines, planes and 3-D geometric primitives. These constraint representations can be used to approximate various anatomical structures, with the important benefit that computing the relative configuration between the tool and the structure can generally be achieved using simple, fast, closed form, numerical methods. There are, however, times when these simple structures are too inflexible to represent an anatomical structure effectively. For instance, the coarseness of a simple constraint representation could force the constraint designer to choose between either under constraining delicate anatomical structures (risking them being damaged), or over constraining the free region (risking interfering or occluding safe movements).

When over or under constraining is unacceptable, it is often necessary to use constraint geometries which conform to the patient's anatomy tightly and hence more complex structures such as parametric surfaces, point clouds and meshes can be used. Of the three primary free-form constraint representations, meshes were selected to be the foundation of this research for complex anatomical structures. Meshes are the most well established and widely used method within the wider literature, and therefore provided a mature basis on which to build. The main advantage of meshes over parametric surfaces is that, as required for dynamic active constraints, they can be deformed in a controlled and predictable way using displacement maps. Deforming a mesh only requires displacing the selected vertices with the underlying anatomy. There is rarely such a simple relationship between control points and anatomy for parametric surfaces and more complex modification methods are required [Terzopoulos and Qin, 1994]. The main advantage of meshes over point clouds, within this context, is that they explicitly describe a surface. This means that the relationship between a tool and a surface can be established without first having to compute the geometry of the underlying surface from the discrete points. There has been research into using implicit constraint surfaces from point clouds, e.g. [Petersen and Rodriguez y Baena, 2013; Rydén and Chizeck, 2012]; however, these implicit surfaces are typically offset from the points within the cloud and therefore the surfaces are not defined precisely, and the motion of the implicit surface under deformation will not be guaranteed to follow the underlying anatomy. Another advantage of mesh based constraint surfaces is that if, as is often the case, they need to be visualised, this can be achieved simply using conventional 3-D visualisation techniques.

Though they have the advantages described above, meshes are a complex and memory demanding way of representing a geometry. The cost of the increased geometric flexibility is subsequently that the relative configuration between the tool and constraint cannot be computed simply. Explicitly searching through all elements within a mesh can be prohibitively slow when there are many, and therefore there has been a large amount of research into methods for reducing their computational complexity and accelerating them. In addition to the few mesh based constraints described in section 2.4 and for general use in robotics, searching meshes for collisions, proximities and penetrations has been a focus of extensive research within the field of 3-D computer graphics due to their importance in simulating and visualising virtual environments, e.g. [Ericson, 2004; Hughs et al., 2013]. Though many methods in the literature can be used for computing collisions, proximities and intersections, constraints are used specifically to prevent a tool and constraint from making contact, therefore it is primarily the process of proximity querying which is of interest.

Proximity querying methods are often considered in two stages: ‘broad phase’ and ‘narrow phase’ [Hubbard, 1995]. During the broad phase, pairs of objects that are close enough to be of interest are identified and those which are large distances apart can be eliminated from further consideration. The identified object pairs are then each processed in the narrow phase to compute their exact proximity and details such as closest point pairs. Within an active constraint application, there are typically few tools and few disjoint constraint regions and therefore broad phase proximity queries are less pertinent. Narrow phase proximity queries are of primary importance however, as the detailed information regarding the interaction between tool and constraint are necessary to enforce the required constraint behaviour.

There are several methods within the literature for narrow phase proximity queries and they are summarised in the following section. Bounding volume hierarchies are hierarchical structures which represent a geometry in a way which is well suited to performing accurate and fast proximity queries in non-convex static environments. However, the use of bounding volume hierarchies within a dynamic active constraint is limited due to the need to refit or rebuild the hierarchies to the anatomical geometries after deformation. Though traversing the hierarchies to compute the proximities are fast, even the most efficient rebuilds are comparatively slow. The relative slowness of the refits and rebuilds can be prohibitive when the deformation rate (i.e. the rate at which the mesh, and constrained anatomy, deforms) approaches the traversal rate (i.e. the rate at which the hierarchy is traversed to compute a proximity). That is to say that, as long as the time taken to refit or rebuild the hierarchy can be shared across many traversals, then searching a hierarchy is fast overall. However, if the long refit or rebuild time is only

split across a few traversals, then searching the hierarchy can become slow.

Within haptic applications, such as active constraints, the time taken for robot control commands to be updated based on new inputs has a direct impact on the stability and feel to the user. In line with the human tactile bandwidth, 1,000 Hz is generally considered to be the control rate required for well perceived haptic interactions and assists with ensuring stability [Srinivasan and Basdogan, 1997; Zhuang and Canny, 2000]. At present, the deformation rate and geometric resolution of a dynamic active constraint which uses bounding volume hierarchies for proximity queries is limited by the time taken to refit or rebuild.

It was hypothesised that a new bounding volume hierarchy could be formulated, which takes advantage of the limited amount of stretch that anatomical tissue can undergo, to increase the range of constraint deformation rates and mesh complexities at which the 1,000 Hz control rate can be maintained. The conceived bounding volume hierarchies are constructed and used so that some traversal efficiency is sacrificed in return for making the hierarchy invariant to deformation such that refits and rebuilds are unnecessary. The formulation and validation of these ‘deformation invariant bounding sphere’ (DIBS) hierarchies, as they are referred to, is described in section 3.3, after some key related work has first been introduced.

3.2 Related Work

A proximity query, also referred to as a closest point search or distance query, is defined by Lin and Manocha as a computation of the “length of the shortest line segment joining two sets of points, A and B ” [Lin and Manocha, 2004] as shown in the following equation:

$$\text{dist}(A, B) = \min_{a \in A, b \in B} \|a - b\|, \quad (3.1)$$

where a is a point in the set A and b is a point in the set B . This can be seen as an extension of the ‘nearest neighbour search’ problem [Knuth, 1973], where the point sets represent the entire surfaces of the mesh or meshes being searched.

The simple approach to performing proximity queries involves exhaustively checking every possible pair of primitives from the objects under consideration and storing the minimum separation found. As complex geometries often contain many thousands of polygons, the inefficiency of this approach tends

to be prohibitive. In general, the methods used to improve the speed of narrow phase proximity queries within the literature fall into four categories, these are: ‘space partitioning hierarchies’; ‘bounding volume hierarchies’; ‘convexity based methods’ and ‘hardware acceleration’ [Ericson, 2004]. A summary of these four approaches to proximity query methods follows.

3.2.1 Space Partitioning Hierarchies

Space partitioning hierarchies allow a polygonal structure, such as a mesh, to be represented hierarchically, by recursively partitioning the Euclidean space in which the object exists [Lin and Manocha, 2004]. The intention with space partitioning hierarchies is that, rather than having to check the proximities of all polygons within a mesh (the leaf nodes within the hierarchy), large numbers of polygons can be excluded from detailed consideration by checking against a the subset of the full Euclidean space in which they reside (the branch nodes within the hierarchy). By replacing many leaf node proximity queries with a single branch node proximity query, the number of calculations required, and hence time taken, for a full proximity query can be significantly reduced.

The simplest form of space partitioning hierarchies are 2-D ‘quadtrees’ [de Berg et al., 1997; Finkel and Bentley, 1974]. Although they cannot be used for proximity queries within 3-D environments, they clearly illustrate the concept of a space partitioning hierarchy and are therefore presented as an introduction. A quadtree can be constructed from a 2-D space of points by recursively subdividing the points into four equally sized square or rectangular quadrants, until each point is isolated from any others. The construction process for a quadtree is top-down and starts by considering all points to be a part of the root node. The set of points is then split into four subsets by two dividing lines, one horizontal line at the vertical midpoint, and one vertical line at the horizontal midpoint. Next, any of the subsets which contain more than one point are further split into four at the new vertical and horizontal midpoints. This process is recursively repeated until none of the sets contain more than one point.

A method for performing a range query using a quadtree is described in [Samet, 1989]. In a range query a search is made for all points within a specified distance of a given centre point. Starting from the root of the hierarchy, the search can immediately eliminate any of the four child quadrants which are further from the centre point than the required search distance. The child quadrants which are not eliminated then each have their own child quadrants inspected until the leaf nodes are reached. At this point, all points which belong to nodes which have not been eliminated must be within the required distance of

the centre point and the search is complete. This search can be modified for a conventional proximity query by systematically updating the range requirement based on the closest point found so far during the traversal.

The direct 3-D equivalent of a quadtree is an ‘octree’ [Meagher, 1982]. Rather than splitting a 2-D set of points into four quadrants, an octree recursively splits a 3-D set of points into eight octants. As octrees are 3-D, they can be used for proximity queries within conventional virtual environments [Dyllong and Luther, 2007]. Even though the structure of an octree allows significant pruning of points within a search, when the spacing between points is uneven the rigid midpoint splitting can lead to unbalanced hierarchies, with some leaf nodes much deeper in the hierarchy than others.

‘ k -d trees’ allow for more balanced, and therefore on average more efficient, hierarchies by subdividing points at their median [Bentley, 1975]. The k within a k -d tree is the number of dimensions in the search space, as they were conceived to search data in any dimension required. Another difference between k -d trees and octrees is that k -d trees are binary trees, with the set of points being split into two equal sized child subsets at each branch node. The splitting still takes place along the axis of the space’s coordinate frame, except the axis used to split the points is alternated through at each level of the hierarchy.

If the hyperplanes used to split the points between sibling nodes do not have to be aligned with the base coordinate frame, then the structure becomes a more generalised ‘binary space partitioning tree’ [Fuchs et al., 1980]. Unlike the previous space partitioning hierarchies, the flexibility of alignment of dividing hyperplanes means that binary space partitioning trees can represent polygonal volumes in a variety of efficient ways [Ericson, 2004].

Space partitioning hierarchies are efficient choices for collision detection because they allow simple exclusions of large half-spaces; however, they can be less effective for proximity queries because the large half-spaces mean that the minimum proximity does not reduce very rapidly until the leaf nodes are reached.

3.2.2 Bounding Volume Hierarchies

Bounding volume hierarchies are a well established method for reducing the number of geometric tests required when assessing the spatial relationship between complex geometric objects [Ericson, 2004; Lin and Gottschalk, 1998]. They operate in much the same way as space partitioning hierarchies, with the

primary difference being that, rather than splitting the polygons between half-spaces, the polygons are split between bounding volumes. As bounding volumes tend to fit the polygons tighter than the half-space intersections within space partitioning hierarchies, they allow the minimum proximity within a proximity query to reduce more quickly, eliminating potential polygons more efficiently. The cost of this increased discriminatory power is that the proximity queries against the bounding volumes themselves are generally more complicated and therefore more time consuming.

The general structure of bounding volume hierarchies are like those of space partitioning hierarchies, with the complete polygonal model contained within the root node and single polygons contained within the leaf nodes. They can have varying branching factors, can be constructed bottom-up or top-down and are generally traversed top-down.

Various volumes have been proposed within the literature to bound the polygons within nodes of the hierarchy, and there is often a trade-off required between the properties of each. ‘Bounding spheres’ [O’Sullivan and Dingliana, 1999; Quinlan, 1994] are the simplest to represent computationally and they are very efficient to examine during traversal; however, they do not generally fit the polygons which they contain particularly tightly and are therefore considered less efficient. ‘Axis aligned bounding boxes’ (AABB) [Cohen et al., 1995] are rectangular cuboids where each edge is parallel to the base coordinate frame axes. AABBs tend to fit the underlying geometry tighter; however, each individual volume is less efficient to query. ‘Oriented bounding boxes’ (OBB) [Gottschalk et al., 1996] are similar to AABBs except they do not require that cuboid edges are parallel to the base axes. This freedom allows for even tighter bounding but individually they are slower to query. ‘Discrete oriented polytopes’ (k -DOP) [Klosowski et al., 1998; Zachmann, 1998] are polytopes with a different number of faces allowing them to be more flexible still.

A common feature of all of these bounding volumes is that they are rigidly constructed around a model, and therefore, if the underlying model deforms, they require updating or reconstruction to ensure that they continue to represent the entire model accurately. Rebuilding a complete bounding volume hierarchy every time the model deforms is often too time consuming for practical application, therefore, several update methods specifically for deforming geometries have been created [Teschner et al., 2005].

In [van den Bergen, 1998], an efficient AABB hierarchy update method is described, in which the leaf nodes are individually refitted and then a bottom-up algorithm refits each parent based on the boxes of their children. A bottom-up method for bounding spheres is described in [Brown et al., 2001], where

the locality of some deformations is exploited to minimise node updates. It was stated in [Larsson and Akenine-Möller, 2001] that, if only a few of the deep nodes are reached during traversal, then the exhaustive AABB update approach in [van den Bergen, 1998] can be inefficient. As an alternative they proposed a ‘hybrid’ approach whereby the top of the tree is updated bottom-up before traversal, and the bottom of the tree is updated when the information is requested during traversal.

The majority of the other approaches used for applying bounding volume hierarchies to deforming objects involve some kind of restriction upon the form of the deformation which is permitted. ‘Bounded deformation trees’ (BD-trees) [James and Pai, 2004] can be efficiently updated when the object’s deformation can be described as an aggregation of displacement fields. In this case the displacement fields themselves are used to update the bounding volumes rather than explicitly considering the surface points. Similarly, [Spillmann et al., 2007] showed how linear and quadratic geometric deformations can be applied directly to update bounding sphere hierarchies.

3.2.3 Convexity Based Methods

When performing a proximity query between convex objects it can be guaranteed that the minimum separation points will be found at a single local minimum which is therefore also the global minimum [Ericson, 2004]. The fact that the only local minimum is the global minimum within a convex proximity query means that the search can be performed by means of a simple gradient descent.

The ‘Lin-Canny’ algorithm exploits the concept of global minimisation to identify the closest features (vertices, edges or faces) between two convex objects using precomputed Voronoi regions [Lin and Canny, 1991]. The method is predicated on the knowledge that two features $a \in A$, $b \in B$ in objects A and B are the globally closest pair, if and only if a is in the Voronoi region of b and b is in the Voronoi region of a . Once complete Voronoi regions have been precomputed for each object, the closest feature pair can be found by iteratively searching from an initial feature pair. At each step of the iteration, if the Voronoi region comparisons fail, then the next feature pair is selected from their neighbours, based on which edges of the Voronoi region were not aligned, so that the method converges towards the global minimum.

An advantage of iterative methods such as Lin-Canny is that in an application where the objects are moving continuously, there will be a degree of consistency between closest features, and therefore the final pair from one step can be used as the initial estimate for the subsequent step, improving expected

convergence time.

The major disadvantage of the Lin-Canny algorithm is that it can fall into infinite loops when the objects intersect or the solution is geometrically degenerate, such as occurs with parallel features. The ‘Voronoi-Clip’, or ‘V-Clip’, algorithm was proposed as a modification on the Lin-Canny algorithm which overcomes these issues [Mirtich, 1998]. By defining the feature neighbourhood in a different way to Lin-Canny, the search transitions within the V-Clip algorithm are acyclic if the minimum separation is not increasing, and therefore finite convergence in all cases is guaranteed.

The separation between two convex, polyhedral objects is equal to the separation of their Minkowski difference from the origin, and therefore once computed, the simplex method [Dantzig, 1963] can easily be used to find their minimum proximity. This attribute is efficiently exploited by the ‘Gilbert-Johnson-Keerthi’ algorithm [Gilbert et al., 1988]. Rather than computing the entire, computationally expensive, Minkowski difference between the two objects, the Gilbert-Johnson-Keerthi algorithm uses a support mapping which allows a direction on a polyhedral object to be mapped into a direction on the Minkowski difference, via a pair of support points [Ericson, 2004]. This mapping means that, without preprocessing, the Gilbert-Johnson-Keerthi algorithm iteratively moves along the vertices and edges of the Minkowski difference such that it monotonically converges to the minimum proximity.

Each of the methods described in this section require that the objects being queried are convex. Some of the methods, particularly V-Clip, can be used for non-convex objects; however, it is necessary to divide the non-convex object into convex pieces. For a complex and potentially deforming object, this requirement can make the methods less attractive.

3.2.4 Hardware Acceleration

With modern technology, many computing problems which require repeated applications of simple functions to large amounts of data can be significantly sped up by applying the functions, in parallel, on dedicated hardware such as ‘graphics processing units’ (GPU) and ‘field programmable gate arrays’ (FPGA). This, so called ‘general purpose computing on graphics processor units’ (GPGPU), is well suited to both simple, exhaustive proximity queries and some of the structural methods described previously.

The primary challenge with hardware acceleration of proximity queries is formulating the functions and structures in such a way that they can be parallelised and run with the limited resources of the

individual hardware cores. In [Chau et al., 2013], it is shown how an exhaustive searching, non-convex, mesh proximity query can be formulated and accelerated on a FPGA. It was also shown how hardware acceleration can be applied to various bounding volume hierarchies for proximity querying and collision detection in [Benallegue et al., 2009; Lauterbach et al., 2010; Zhang and Kim, 2007].

Even with the wide range of literature available, performing proximity queries between deforming, non-convex objects still represents a significant computational challenge. Within the field of dynamic active constraints, and indeed all surgical simulation, the specific deformation properties of the soft tissue environment allow for an optimisation to be made. A new formulation, deformation invariant bounding sphere hierarchies, are described below, which take advantage of the deformation properties to achieve a significant performance improvement.

3.3 Deformation Invariant Bounding Spheres

This section describes how a new form of bounding volume can be formulated to increase the computational efficiency of proximity queries for deforming surfaces, over methods currently available within the literature. The proposed features of these deformation invariant bounding spheres mean that, when they are assembled within a conventional bounding volume hierarchy, they capitalise upon the deformation properties of soft tissue so that they do not require refitting or rebuilding when the geometry which they represent deforms.

3.3.1 Surface ‘Pinned’ Bounding Volumes

A key aspect of the proposed method for constructing a deformation invariant bounding volume hierarchy is the way that the individual bounding volumes are ‘pinned’ onto the surface of the deforming object. This means that the bounding volumes follow the surface of the model as it deforms, as though part of the bounding volume was physically pinned onto the surface.

The idea of having a topologically fixed hierarchy, where each bounding volume moves over time so that it tracks a certain portion of the model’s surface is not a novel concept. This is the aim of many other deformable techniques such as in [Larsson and Akenine-Möller, 2001]; however, these methods rely on manual repositioning and resizing of each bounding volume. Using ‘pinned’ bounding volumes, the repositioning is carried out automatically as the mesh changes, without any special consideration.

Pinned bounding volumes can be implemented when the surface of the deforming model is represented using a set of Cartesian points, such as the vertices of a polygonal mesh. Each one of these vertices provides a possible pinning location for one or more of the bounding volumes within the hierarchy. To pin a bounding volume to a specific vertex, the bounding volume is linked to an entry in a dynamic vertex table, rather than storing an explicit Cartesian coordinate, for its position. During traversal of the hierarchy, when the position of the bounding volume is required, the vertex is simply retrieved from the table and its current position is used.

There are several limitations on the geometries and properties of bounding volumes which can be pinned to a deforming surface in this way. For the purposes of clarity, only deformations where the external surface area of a model is constant will be considered here, cases where this simplification does not hold will be discussed in the following section.

An important consideration is that the surfaces of a model are only guaranteed to be flat at the lowest level of a hierarchy, where they contain a single polygon. The surface sections bounded by higher branch nodes can be curved and therefore deformation can cause flattening and, as a result, gaps in the surface coverage when pinning is used. An example of surface deformation causing this type of failure is shown in figures 3.1a - 3.1c.

A possible solution to this problem is to use oversized spheres, based on the surface topology of the contained surface, for the bounding volumes. The radius of a conventional bounding sphere is the maximum Euclidean separation between the centre and any of the vertices within the mesh section. Instead of this, the maximum geodesic length between the centre and each vertex is used for the sphere's radius (see figure 3.1d). The geodesic length is the shortest distance between two surface points while following the surface profile. For a vertex v in the set of surface polygons P , the geodesic bounding sphere radius r_g is a function of the sphere centre vertex v_c and a geodesic separation function $d_g(\cdot)$ as follows:

$$r_g = \max_{v \in P} d_g(v_c, v). \quad (3.2)$$

In a practical implementation of this method, a shortest path tree can be constructed using the edges and vertices of a mesh to approximate the geodesic separations to each vertex. A crucial property of spheres is that they are rotationally invariant with respect to the centre, and therefore, if the geodesic separation radius is used, then it will be guaranteed to cover the surface under all configurations.

As this method uses spheres, and as the sphere centres have to be located on the surface of a model,

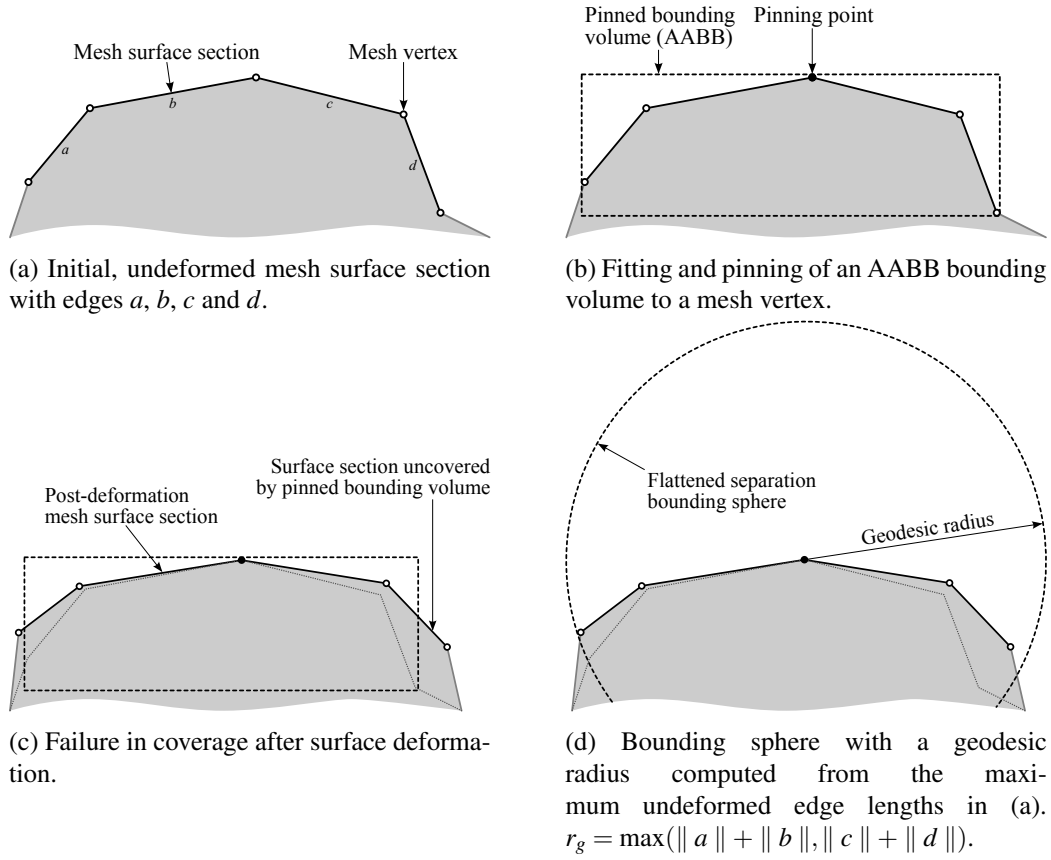


Figure 3.1: An example of how surface deformation can cause tightly fitting pinned bounding volumes to fail (a-c) and how a conservative bounding sphere can overcome this (d).

each bounding volume will not fit the surface section as tightly as using other methods. It was shown in [Gottschalk et al., 1996] that tightness of bounding was important, and that bounding spheres do not perform as well as tighter fitting bounding volumes such as OBBs. However, this method is not solely concerned with tightly bounding the current geometry. Instead, it aims to bound a volume in space, which the model’s surface section will always remain inside of as it deforms. An additional advantage of using bounding spheres is that they are considerably more efficient to query during hierarchy traversal than others, as they are primarily considered as a single Cartesian point.

3.3.2 Assumption of Limited Surface Stretch

In the previous section, pinned bounding spheres were described where the surface area of a model remained constant. There may be some practical applications where this holds, but for the majority of cases the deformation will involve some stretching of the model’s surface. To allow the extension

of pinned bounding volumes to these cases, an assumption is made regarding the amount of stretch a surface will sustain, and this expected limit is used to inflate the pinned bounding spheres. It is these pinned and inflated bounding spheres which are referred to as Deformation Invariant Bounding Spheres (DIBS).

The limit on a model's surface deformation might appear to be restrictive; however, soft tissue structures typically only stretch by a relatively small amount before being damaged¹ and therefore DIBS can be effectively used in a range of clinical applications. In this work, the limit on a model's surface deformation is described by a 'limiting surface stretch factor' σ . The value of σ states that, after deformation, no pair of points on the surface can be further apart than σ times their initial geodesic separation. Formally, for a deformation D , a surface is said to obey a limiting surface stretch if and only if:

$$\sigma d_g(p, q) \geq d_g(D(p), D(q)) \quad \forall p, q \in P, \quad (3.3)$$

where p and q are vertices on the surface of the model, $D(p)$ and $D(q)$ are corresponding vertices on the surface of the deformed model.

Once a suitable value for σ has been established for a model, the final radius of the bounding sphere r_σ can be computed based on equation (3.2):

$$r_\sigma = \sigma \max_{v \in P} d_g(v_c, v). \quad (3.4)$$

If a conventional, static, bounding volume hierarchy was simply inflated in this way, it would not achieve large deformation invariance because each individual bounding volume's position would still be fixed as it was set during initial construction. Oversized bounding volumes have been used to allow for small amounts of deformation between slow hierarchy refits [Mezger et al., 2003], but these methods rely on temporary limits to the absolute deformation of a model such that each point in the model only moves by a limited amount from its initial, pre-deformation, position. This is a much more restricting concept than using them in combination with surface pinning and many cases would require extremely large σ values, making traversal inefficient.

To summarise the previous two sections, the specific properties of DIBS are as follows:

¹Fung gives the approximate distensibility of certain soft tissue structures as: mesentery 100-200 %, ureter 60 %, heart muscle 15 %, arteries and veins 60 %, skin 40 % and tendons 2-5 % [Fung, 1993]. Additionally, Deck and Willinger state that 40 % first principal strain in brain tissue causes a 50 % probability of severe diffuse axonal injury [Deck and Willinger, 2008].

- The centre of each bounding sphere is centred on the surface of the model at a pinning point.
- The unstretched radius of each bounding sphere is computed from the geodesic separation of the contained vertices using equation (3.2).
- The stretched radius of each bounding sphere is computed from the limiting surface stretch factor using equation (3.4).

The following section describes how a hierarchy of DIBS can be constructed with these properties.

3.3.3 Hierarchical construction

A top-down method was selected for the construction of DIBS hierarchies as this has previously been used for bounding spheres in [O’Sullivan and Dingliana, 1999]. In this top-down approach, the entire model is initially used as the root node. The model’s surface is then recursively subdivided into smaller and smaller surface sections, with new descendent nodes created for each, until single polygons remain at the leaf nodes. The choices of top-down construction and a binary structure were only based upon trends within the literature and further analysis would be required to ensure this is optimal.

A bounding sphere is fitted to every node in the hierarchy during construction. There are two parameters required for each bounding sphere, the pinning vertex v_c (i.e. a vertex on the surface section to which the sphere will be pinned) and a stretched geodesic radius r_σ . The computation of these parameters is described below for branch and leaf nodes.

3.3.3.1 Fitting bounding spheres to branch nodes

To ascertain the pinning vertex (i.e. the centre) for a branch node’s bounding sphere, the Cartesian mid-range ϕ of the surface polygons P is first computed in the x , y and z Cartesian axes. The centre vertex is then taken as the vertex in P which is closest to ϕ . Numerically, v_c is computed from:

$$\phi_x = (\max X(P) + \min X(P))/2 \quad (3.5)$$

$$\phi_y = (\max Y(P) + \min Y(P))/2 \quad (3.6)$$

$$\phi_z = (\max Z(P) + \min Z(P))/2 \quad (3.7)$$

$$v_c = \arg \min_{v \in P} \|v - \phi\| \quad (3.8)$$

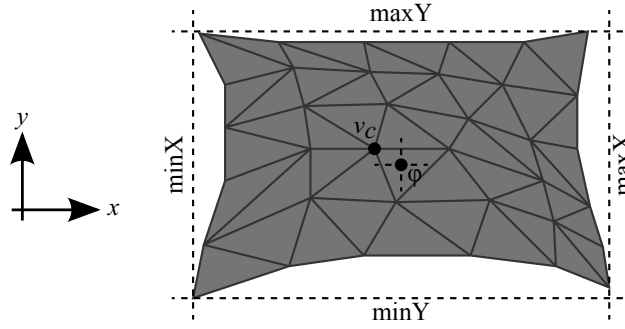


Figure 3.2: Two dimensional illustration of the method for defining the pinning vertex, v_c . The Cartesian mid-range, ϕ , of the surface polygons is computed from the axis limits in x and y and then the pinning vertex is selected from those in the surface polygons as the one which has the shortest Cartesian distance from ϕ .

where $\max X$ and $\min X$ are functions returning the maximum and minimum vertex x values from the supplied polygon set. This is similarly defined for y and z . The full computation of the pinning vertex based on these equations is illustrated in figure 3.2. Alternative definitions for ϕ , such as the mean and median vertex positions, were investigated, however, the mid-range was found to allow for the fastest traversal.

To compute the stretched radius of the bounding sphere, the geodesic radius is required. The geodesic radius of a surface section is approximately equivalent to the distal node separation in a shortest path tree formed from its vertices and edges. Dijkstra's algorithm [Dijkstra, 1959] is used to compute the shortest path tree for each surface section. The algorithm is initialised so that the vertices and edges in the surface section are used as vertices and edges in the graph, edge costs are the Euclidean separations between vertices and the centre vertex v_c is used as the starting point. Once Dijkstra's algorithm has returned the distal node separation from the polygon set, this can be used with equation (3.4) to produce the bounding sphere's radius r_σ .

3.3.3.2 Fitting bounding spheres to leaf nodes

Due to the comparatively high computational expense of polygon operations, it was decided that the leaf bounding spheres (i.e. those which contain a single polygon and have no child nodes) should be tight fitting, to prevent as many unnecessary polygon operations as possible. Tightened leaf bounding spheres were achieved using so-called 'elastic' bounding spheres.

The pinning vertex of an elastic bounding sphere is found in the same way as for branch nodes; however,

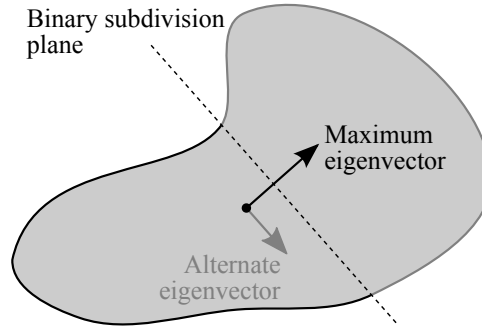


Figure 3.3: Example of orienting a dividing plane so that it is orthogonal to the eigenvector associated with the largest eigenvalue of surface vertices' covariance matrix. By splitting a set of polygons at this oriented dividing plane the sizes of the bounding spheres fitted to the child nodes can be minimised, making the subsequent hierarchy traversal more efficient.

no radius is stored during construction. Instead, if a leaf node is reached during hierarchy traversal, the algorithm computes the maximum Cartesian separation between the pinning vertex and the other vertices in the polygon and uses this value (without any stretching) as the radius of the bounding sphere.

Establishing the maximum vertex separation for complex polygons during traversal could be costly; however, this is compensated for by the reduction in number of polygon operations. This elastic method of refitting leaf bounding spheres when required is similar to top-down hierarchy updates discussed in [Larsson and Akenine-Möller, 2001].

3.3.3.3 Binary subdivision of surface sections

If the node is not a leaf then, once a bounding sphere has been fitted, the recursive construction algorithm subdivides the surface section into two parts for the creation of child nodes. The subdivision is performed using a single, oriented, dividing plane. Taking the concept from that used in OBB [Gottschalk et al., 1996], the dividing plane is oriented so that it is orthogonal to the eigenvector associated with the largest eigenvalue of the surface vertices' covariance matrix, as illustrated in figure 3.3. By making the dividing plane orthogonal to the maximum eigenvector, the construction process minimises the aspect ratio of the polygon set in the child bounding spheres, allowing them to be bound by spheres more tightly, and hence more efficiently.

The dividing plane is positioned along the maximum eigenvector so that it is exactly midway between the maximum and minimum vertex positions when projected onto the maximum eigenvector, again, minimising the child polygon sets' aspect ratios. The implementation of this division process iterates

through the list of surface polygons and allocates each to one of the two child lists based on their scalar projection onto the maximum eigenvector. Formally the polygon classification is performed as follows:

$$P_{child-a} = \{p \in P_{parent} : \tilde{p} \geq (\tilde{p}_{\max} + \tilde{p}_{\min})/2\} \quad (3.9)$$

$$P_{child-b} = \{p \in P_{parent} : \tilde{p} < (\tilde{p}_{\max} + \tilde{p}_{\min})/2\} \quad (3.10)$$

where p is a polygon on the parent surface section P_{parent} , $\tilde{p}$ is the scalar projection of the centre of polygon p onto the maximum eigenvector and $\tilde{p}_{\max}$ and $\tilde{p}_{\min}$ are the maximum and minimum scalar projections respectively.

A result of using geodesic surface separations is that the surfaces considered must be contiguous. The geodesic separation between two disconnected points in a non-contiguous surface is undefined and therefore the above description for fitting bounding spheres would fail. To overcome this problem, a check of contiguity after each subdivision is performed, and orphaned sections are swapped between siblings. Additionally, the necessity of geodesic surface separations means that all raw meshes to be searched with a DIBS hierarchy must be fully contiguous to start with.

3.3.4 Real-time proximity queries

To perform a proximity query using a DIBS hierarchy, algorithm 1 can be used. This algorithm is a recursive, depth first, search and is similar to that for conventional bounding sphere hierarchies in [Quinlan, 1994]. The initial parameters for this algorithm are the DIBS hierarchy D and the current tool geometry T , and it returns the minimum separation sep and the closest point pair $\langle cp_D, cp_T \rangle$.

It should be noted that the traversal algorithm presented here is specifically for cases where the representation of the tool (potentially as a proxy, as described in section 7.3.2) and the constraint surface should not collide, and their relative configuration can be described based solely on minimum proximities between them. If the tool and mesh do collide, the algorithm will identify that the proximity is zero and it will return one of the, potentially infinite, locations at which their surfaces intersect. If full contact information is required then the DIBS hierarchy can also be used for collision detection, similarly to the method described in [van den Bergen, 1998]; however, this is not the focus of the research presented here.

Algorithm 1 Recursive proximity query function

procedure DIBS-PROXIMITY-QUERY(D, T) $m \leftarrow \infty$

▷ Initialise minimum separation

 $n \leftarrow \text{root}(D)$

▷ Initialise search node structure

return RECURSIVE-PQ(m, n, T)**end procedure****procedure** RECURSIVE-PQ(m, n, T) $sep \leftarrow \infty$

▷ Initialise separation

 $\langle cp_D, cp_T \rangle \leftarrow \text{NULL}$

▷ Initialise closest points

if $\text{leaf}(n)$ **then**

▷ Leaf node

 $poly \leftarrow \text{getAssociatedPolygon}(n)$ $\langle cp_D, cp_T \rangle \leftarrow \text{findClosestPts}(poly, T)$ $sep \leftarrow \text{separation}(cp_D, cp_T)$ **else**

▷ Branch node

 $i \leftarrow 0$ **for all** $c \in \text{children}(n)$ **do**

▷ Compute child DIBS separations

 $pp \leftarrow \text{getPinningPt}(c)$ $rad \leftarrow \text{getRadius}(c)$ $\text{dibsseps}(i) \leftarrow \text{separation}(pp, T) - rad$ $i \leftarrow i + 1$ **end for** $\text{dibsseps} \leftarrow \text{sortAscending}(\text{dibsseps})$ **while** $\text{front}(\text{dibsseps}) < m$ **do**

▷ Recursively search ordered children

 $c \leftarrow \text{associatedChild}(\text{front}(\text{dibsseps}))$ $\langle cs, c1, c2 \rangle \leftarrow \text{RECURSIVE-PQ}(m, c, T)$ **if** $cs < m$ **then** $m \leftarrow cs$ $\langle sep, cp_D, cp_T \rangle \leftarrow \langle cs, c1, c2 \rangle$ **end if** $\text{dibsseps} \leftarrow \text{popFront}(\text{dibsseps})$ **end while****end if****return** $\langle sep, cp_D, cp_T \rangle$ **end procedure**

3.4 Experiments

To validate the effectiveness of using DIBS hierarchies for proximity queries, two experiments were carried out. Standard refittable AABB hierarchies [van den Bergen, 1998], hybrid refittable AABB hierarchies [Larsson and Akenine-Möller, 2001] and exhaustive searches were implemented for comparison, as these are widely used and do not rely on hardware acceleration.

The first experiment was entirely simulated and involved randomly deforming a surface mesh while randomly positioning a tool geometry relative to it, and then performing a proximity query between them. In the second experiment, the proximity between a time-dependent beating heart model and a human controlled tool was continuously computed.

These experiments were performed using single 3-D geometric primitives, namely spheres and capsules (cylinders with rounded ends), for the tool geometries and manifold, triangular, surface meshes for the deforming objects. The proximity queries were implemented in a single C++ thread and were run on a 3.0 GHz Core i7 CPU (Intel, Corp.). An abstract base class was used to implement the hierarchy traversal algorithm shown in algorithm 1. This class was inherited by each of the three bounding volume hierarchies under investigation, thus allowing for an unbiased comparison of their effectiveness.

3.4.1 Randomised Positioning and Deformation Experiment

For this experiment, four surface meshes of a human brain were used to represent constraint geometries. The four meshes were constructed from the same initial surface, but they were each sampled at different resolutions to create mesh sizes across four orders of magnitude. These meshes ranged in size from 500 to 500,000 polygons and are shown in figure 3.4.

Prior to the experiment, the DIBS and AABB hierarchies were constructed from the meshes in their initial, undeformed state. Based on the brain tissue strain limit in [Deck and Willinger, 2008], the DIBS hierarchy was constructed with a conservative limiting stretch factor of 2 (i.e. allowing for a tissue stretch of up to 100 %). Experiments were used to ascertain the optimum level for splitting bottom-up and top-down updates of the hybrid hierarchy for each mesh size.

Before each proximity query was performed, the mesh in use was modified from this initial state so as to induce both local and global random surface deformation. The deformation was produced using a combination of random, independent, sinusoidal deformations and linear scalings in the x , y and z axes.

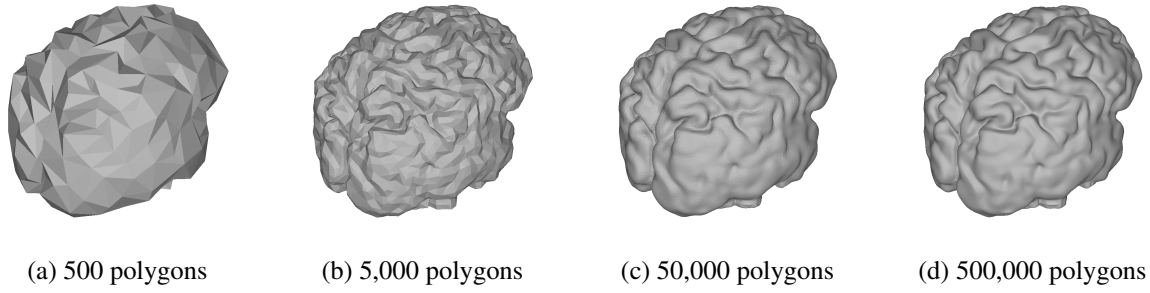


Figure 3.4: Multi-resolution brain mesh used within the randomised validation of the deformation invariant bounding sphere method. The meshes were produced with sizes across four orders of magnitude by decimating a single high resolution surface.

The rigid tool was positioned and oriented randomly within close proximity of the deforming mesh such that there was a mean separation of approximately $40 \text{ mm} \pm 30 \text{ mm}$. After each random deformation and tool positioning, a proximity query was made between the tool and surface using DIBS, AABB, hybrid AABB and exhaustive searches.

During the experiment, the deformation rate was equal to the traversal rate, such that the total search time was calculated from the sum of the refit time and the traversal time. Each of the four meshes was randomly deformed and searched 10,000 times with each of the four methods.

3.4.2 Beating Heart Experiment

To assess the DIBS hierarchies in a scenario closer to conventional active constraints, an experiment was performed using a simulation of a beating heart (taken from [Kwok et al., 2010]). The heart simulation was a surface mesh of 10,272 triangles, with each vertex's position defined based on a sinusoidal interpolation between three control frames. In practice, a periodic motion, such as that of a beating heart, presents efficient opportunities for precomputed bounding volume hierarchies or distance fields for each time step (as in [Ren et al., 2008]); however, for the purposes of this experiment, it was posited that the deformation could not be predicted and therefore real-time updates were required.

In this experiment, the simplified tool geometry was positioned relative to the beating heart by a human operator with the use of a Phantom Omni (Sensable Technologies, Inc.) haptic device, without force feedback. The single human operator moved the capsule both around and through the beating heart randomly, while the bounding volume hierarchies were used to compute the closest point pairs. Based on the estimated 15 % distensibility of heart muscle [Fung, 1993], a conservative limiting stretch factor

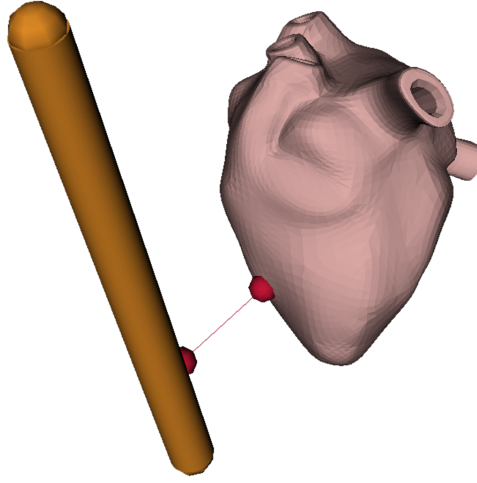


Figure 3.5: Image from the proximity query experiment between a beating heart mesh and a human positioned capsule. The user manually moved the orange tool around the heart, using a Phantom Omni as an input device, until over 20,000 proximity queries had been performed against the surface using all four of the methods within the experiment. The red line identifies the pair of closest points.

of 1.5 was sufficient for this experiment.

An image from this experiment is shown in figure 3.5. The tool motion was continued until over 20,000 proximity queries had been performed using each method. To ensure the integrity of the experiment, all four search methods were evaluated at each time step, and therefore they all operated upon the same deformation and position configuration.

3.5 Results

3.5.1 Randomised Positioning and Deformation Experiment

The results of the randomised positioning and deformation tests are shown in table 3.1 for the sphere primitive and table 3.2 for the capsule primitive. The graphs in figure 3.6 show the trends for total search times across the four meshes. A one-way analysis of variance test was applied to the total search times for each mesh size and all were found to be significantly affected by the search method ($p < 10^{-6}$). Tukey's honest significant difference test was then used to perform pairwise comparisons between each search method, and all were found to be significantly different from all others at the 99.9 % confidence interval.

Mesh size	Method	Refit time (ms)	Traversal time (ms)	Polygon PQs	BV PQs	Total search time (ms)
500 polygons	DIBS	0.00 ± 0.00	0.02 ± 0.02	52 ± 20	419 ± 117	0.02 ± 0.02
	AABB	0.04 ± 0.02	0.00 ± 0.01	14 ± 7	80 ± 30	0.05 ± 0.02
	Hybrid AABB	0.01 ± 0.01	0.02 ± 0.01	14 ± 7	80 ± 30	0.03 ± 0.01
	Exhaustive	0.00 ± 0.00	0.02 ± 0.01	500 ± 0	0 ± 0	0.02 ± 0.01
5,000 polygons	DIBS	0.00 ± 0.00	0.07 ± 0.03	125 ± 47	1,318 ± 432	0.07 ± 0.03
	AABB	0.52 ± 0.06	0.01 ± 0.01	26 ± 15	174 ± 80	0.53 ± 0.06
	Hybrid AABB	0.12 ± 0.02	0.06 ± 0.04	26 ± 15	174 ± 80	0.18 ± 0.04
	Exhaustive	0.00 ± 0.00	0.21 ± 0.04	5,000 ± 0	0 ± 0	0.21 ± 0.04
50,000 polygons	DIBS	0.00 ± 0.00	0.24 ± 0.10	254 ± 121	2,488 ± 991	0.24 ± 0.10
	AABB	5.81 ± 0.57	0.02 ± 0.01	68 ± 46	398 ± 221	5.83 ± 0.57
	Hybrid AABB	1.61 ± 0.21	0.16 ± 0.09	68 ± 46	398 ± 221	1.77 ± 0.23
	Exhaustive	0.00 ± 0.00	1.97 ± 0.14	50,000 ± 0	0 ± 0	1.97 ± 0.14
500,000 polygons	DIBS	0.00 ± 0.00	0.64 ± 0.26	638 ± 325	5,917 ± 2,510	0.64 ± 0.26
	AABB	74.72 ± 2.51	0.09 ± 0.06	189 ± 137	1,010 ± 643	74.81 ± 2.51
	Hybrid AABB	20.04 ± 0.78	0.61 ± 0.37	189 ± 137	1,010 ± 643	20.65 ± 0.88
	Exhaustive	0.00 ± 0.00	18.41 ± 0.82	500,000 ± 0	0 ± 0	18.41 ± 0.82

Table 3.1: Comparative performance for proximity queries between randomly positioned spheres and deforming meshes of differing sizes. The total search time for each test was calculated from the sum of the refit time and the traversal time. ‘Polygon PQs’ identifies the mean number of proximity queries performed against the primitives in the meshes and ‘BV PQs’ identifies the mean number of proximity queries performed against the bounding volumes in the hierarchies. Deformation invariant bounding sphere hierarchies are denoted DIBS and axis aligned bounding box hierarchies are denoted AABB. The results shown are the means and standard deviations of 10,000 tests.

Mesh size	Method	Refit time (ms)	Traversal time (ms)	Polygon PQs	BV PQs	Total search time (ms)
500 polygons	DIBS	0.00 $\pm$ 0.00	0.04 $\pm$ 0.02	55 $\pm$ 23	434 $\pm$ 130	0.04 $\pm$ 0.02
	AABB	0.04 $\pm$ 0.02	0.04 $\pm$ 0.02	12 $\pm$ 7	75 $\pm$ 30	0.08 $\pm$ 0.03
	Hybrid AABB Exhaustive	0.01 $\pm$ 0.01 0.00 $\pm$ 0.00	0.05 $\pm$ 0.02 0.11 $\pm$ 0.02	12 $\pm$ 7 500 $\pm$ 0	75 $\pm$ 30 0 $\pm$ 0	0.06 $\pm$ 0.02 0.11 $\pm$ 0.02
5,000 polygons	DIBS	0.00 $\pm$ 0.00	0.12 $\pm$ 0.05	131 $\pm$ 55	1,392 $\pm$ 488	0.12 $\pm$ 0.05
	AABB	0.52 $\pm$ 0.06	0.08 $\pm$ 0.05	23 $\pm$ 15	163 $\pm$ 86	0.60 $\pm$ 0.08
	Hybrid AABB Exhaustive	0.12 $\pm$ 0.02 0.00 $\pm$ 0.00	0.13 $\pm$ 0.07 1.05 $\pm$ 0.09	23 $\pm$ 15 5,000 $\pm$ 0	163 $\pm$ 86 0 $\pm$ 0	0.25 $\pm$ 0.08 1.05 $\pm$ 0.09
50,000 polygons	DIBS	0.00 $\pm$ 0.00	0.33 $\pm$ 0.16	266 $\pm$ 166	2,651 $\pm$ 1,217	0.33 $\pm$ 0.16
	AABB	5.81 $\pm$ 0.57	0.18 $\pm$ 0.12	59 $\pm$ 49	351 $\pm$ 234	5.99 $\pm$ 0.58
	Hybrid AABB Exhaustive	1.61 $\pm$ 0.21 0.00 $\pm$ 0.00	0.30 $\pm$ 0.20 9.73 $\pm$ 0.46	59 $\pm$ 49 50,000 $\pm$ 0	351 $\pm$ 234 0 $\pm$ 0	1.91 $\pm$ 0.30 9.73 $\pm$ 0.46
500,000 polygons	DIBS	0.00 $\pm$ 0.00	0.90 $\pm$ 0.42	653 $\pm$ 399	6,000 $\pm$ 2,756	0.90 $\pm$ 0.42
	AABB	74.72 $\pm$ 2.51	0.45 $\pm$ 0.31	162 $\pm$ 129	863 $\pm$ 590	75.16 $\pm$ 2.54
	Hybrid AABB Exhaustive	20.04 $\pm$ 0.78 0.00 $\pm$ 0.00	0.91 $\pm$ 0.59 87.55 $\pm$ 3.36	162 $\pm$ 129 50,000 $\pm$ 0	863 $\pm$ 590 0 $\pm$ 0	20.95 $\pm$ 1.00 87.55 $\pm$ 3.36

Table 3.2: Comparative performance for proximity queries between randomly positioned capsules and deforming meshes of differing sizes. The total search time for each test was calculated from the sum of the refit time and the traversal time. ‘Polygon PQs’ identifies the mean number of proximity queries performed against the primitives in the meshes and ‘BV PQs’ identifies the mean number of proximity queries performed against the bounding volumes in the hierarchies. Deformation invariant bounding sphere hierarchies are denoted DIBS and axis aligned bounding box hierarchies are denoted AABB. The results shown are the means and standard deviations of 10,000 tests.

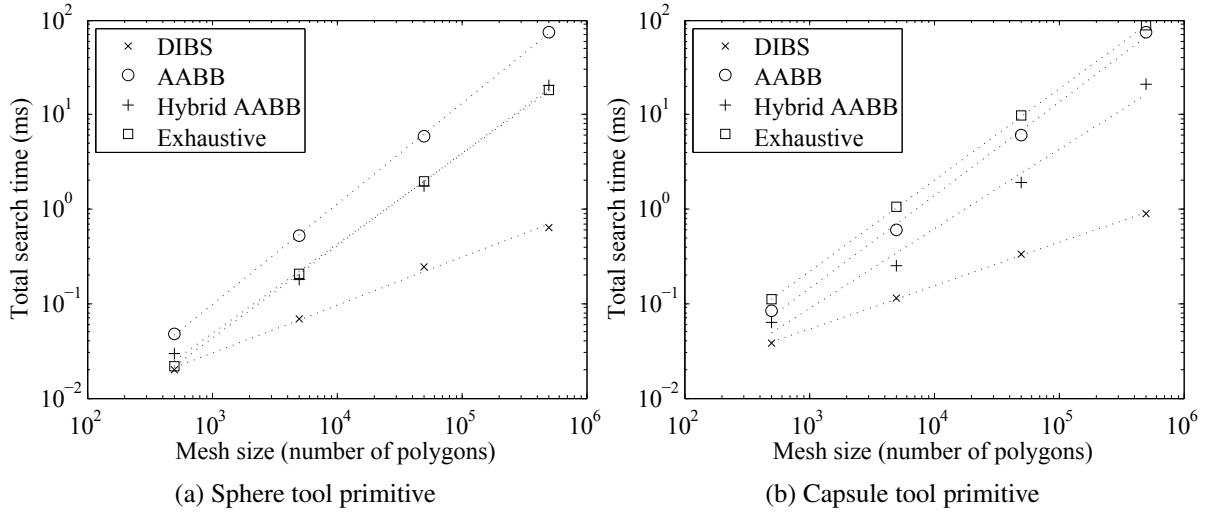


Figure 3.6: Log-log plots of mean total search time against mesh size for the four proximity query methods used in the experiments with sphere and capsule tool primitives. The graphs show that the total search times with the deformation invariant bounding spheres (DIBS) were lower than for the axis aligned bounding box (AABB) methods and the exhaustive searches across all four mesh sizes. Moreover, the trend line for the DIBS method has a lower gradient than those for the other methods, implying that as the mesh sizes increase, the benefits of the method becomes greater.

Proximity query method	Frequency (Hz)
DIBS	7,222
AABB	873
Hybrid AABB	1,819
Exhaustive	503

Table 3.3: Comparative performance for proximity queries between a capsule and a deforming heart mesh. Deformation invariant bounding sphere hierarchies are denoted DIBS and axis aligned bounding box hierarchies are denoted AABB. Results shown are the mean produced from a total of 23,081 queries.

3.5.2 Beating Heart Experiment

For the beating heart experiment, the only metric considered was the mean search frequency across all proximity queries. The results for this experiment are displayed in table 3.3.

3.6 Discussion

The results shown in tables 3.1, 3.2 and 3.3 illustrate that using DIBS hierarchies in proximity queries, between continuously deforming and rigid geometries, can be more efficient than two widely used bounding volume hierarchy methods and exhaustive searches.

In figure 3.6, the total search times for DIBS hierarchies are shown to be lower than those for the other methods across all four mesh sizes. Additionally, the trends for the DIBS hierarchies have a lower gradient than the other methods, such that increasing mesh sizes gives DIBS an increasing benefit when compared to the others. For the two larger meshes, the DIBS hierarchy was the only method of the four which returned from a proximity query, on average, faster than 1,000 Hz and therefore this improvement is significant for allowing an active constraint system to use constraint geometries which accurately represent the full detail of the patient's anatomy.

To create bounding spheres which are invariant to deformation, it is necessary to make them oversized, and therefore looser in bounding the objects which they represent. The consequence of this is shown in tables 3.1 and 3.2, where the traversal times, the number of polygon separation tests and the number of bounding volume separation tests for the DIBS hierarchy are often greater than those for the AABB methods. The reduced efficiency of traversing a DIBS hierarchy, when compared to an already refitted AABB hierarchy, means that its intended application must be considered before it is used.

In the tests shown here, the object deforms and the hierarchy is updated at every search. That is to say that, the ratio between refits and traversals is equal to one. This ratio is referred to as the refit ratio η :

$$\eta = \frac{\text{hierarchy refits per second}}{\text{hierarchy traversals per second}}. \quad (3.11)$$

If the object deforms only intermittently, and the refits are required less frequently, or if multiple tools are considered at each step, then the refit ratio will be less than one. When this happens the cost of each refit can be spread across several traversals, and the average speed of the AABB methods will increase, while the DIBS speed will remain the same. The result of this is that as the refit rate reduces, AABB hierarchies will eventually become faster than DIBS hierarchies.

The point at which the refits are rare enough such that the AABB hierarchy becomes more efficient than DIBS is defined by a crossover refit ratio η_c , which is a function of the DIBS traversal time t_{dt} , the AABB traversal time t_{at} and the AABB refit time t_{ar} :

$$\eta_c = \frac{t_{dt} - t_{at}}{t_{ar}}. \quad (3.12)$$

The experimental values for the crossover refit ratios, based on the mean times in tables 3.1 and 3.2, are shown in figure 3.7.

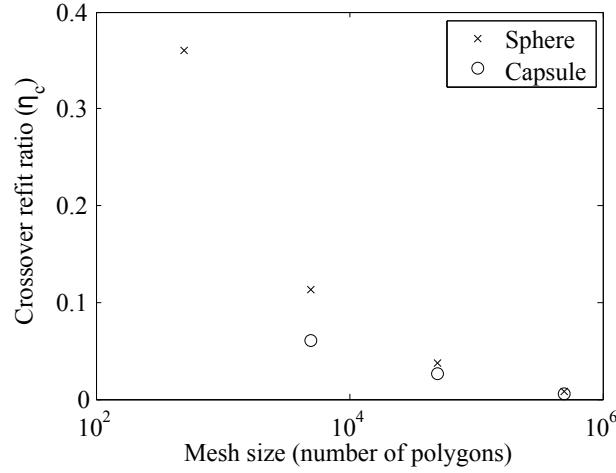


Figure 3.7: Plot of the crossover refit ratios above which deformation invariant bounding sphere (DIBS) hierarchies are more efficient. The computed values are based on the mean traversal and refit times taken from tables 3.1 and 3.2. No value is shown for the capsule with the 500 polygon mesh because the DIBS traversal time was actually equal to that for the axis aligned bounding box traversal time such that DIBS is faster for all refit ratios.

An unexpected result illustrated in figure 3.7 is that, for the small mesh with a capsule tool, the DIBS hierarchy takes the same amount of time to traverse as the AABB hierarchy, meaning that it is always faster to search, irrespective of how often the mesh deforms. The cause of this idiosyncrasy is believed to be the relatively low ratio between bounding volume proximity queries in this configuration. The individual sphere-capsule proximity queries used by the DIBS hierarchy are significantly simpler, and therefore faster, than AABB-capsule proximity queries used by the AABB hierarchy. In this simple configuration the efficiency of the faster individual proximity queries is great enough that it balances the fact that more individual proximity queries are performed and the overall traversals take the same time. For the other configurations, the ratio between bounding volume proximity queries increases and the DIBS hierarchy becomes slower to traverse than the others.

For the smallest mesh, the sphere crossover refit ratio is relatively high (approximately one refit in three traversals) limiting the number of applications where DIBS hierarchies would be advantageous. However, exhaustive searches are almost as efficient for these small meshes and, in practice, a bounding volume hierarchy would be unnecessary. It is for larger meshes where the crossover refit ratio drops and DIBS hierarchies are of most benefit. For the largest mesh, the crossover refit ratio is less than 0.01, meaning that if the mesh deforms more often than once in 100 traversals, then DIBS hierarchies are more useful than AABB hierarchies. For a conventional traversal rate of 1,000 Hz, this translates to a deformation frequency of 10 Hz, which is well within the range of most conventional tracking

and imaging systems. As the rate of imaging and real-time tissue modelling techniques improve over time, the potential benefit from using DIBS hierarchies over the alternatives presented here will increase further.

Although figure 3.7 shows that DIBS hierarchies can be slower to search than conventional methods when the deformation rate is low, DIBS hierarchies are not necessarily unsuitable in these cases. When the refit ratio is less than one, the time for each AABB refit is spread across several searches to guarantee a consistent control rate. The effect of this is that there is a latency between an input deformation and the enforcement of new constraints based on that deformation. If the motion of a constrained region is high, then this delay can cause inaccuracies in the constraint positioning, instabilities or poor surface rendering for the user. All of these problems can be mitigated with the immediate updates achieved using DIBS hierarchies.

It was hoped that the positive results shown here could be translated into more diverse proximity querying applications. Several preliminary experiments were performed with proximity queries between two deforming bodies, both represented with DIBS hierarchies. Though this was not the requirement for a dynamic active constraint, this is the formulation in which many proximity queries are performed within fields such as computer graphics. In these experiments, it was discovered that the computational time saved by obviating the rebuild or refit of the bounding volume hierarchy could not make up for the inefficiency of the loose bounding spheres. This behaviour can be understood by considering the increases in computational time for the refit/rebuild and traversal of a bounding volume hierarchy. During a mesh-mesh proximity query, the rebuild time only doubles, whereas the number of bounding volume proximity queries is approximately squared. As explained previously, DIBS hierarchies are efficient when the time spent refitting/rebuilding is a significant component of the overall search time. During a mesh-mesh proximity query, the refit/rebuild becomes a very small component of the overall search time, and therefore the efficacy of the DIBS hierarchy is lost. The conclusion from this is that DIBS hierarchies are only well suited to applications where the proximity query is required between a large deforming object and a simple rigid object, exactly as it occurs within a dynamic active constraint.

The experimental results have shown that DIBS hierarchies achieve the proposed goal of increasing the sizes of constraint meshes permitted, and the range of deformation rates, across which the target control frequency of 1,000 Hz can be maintained. There are, however, aspects of the proposed DIBS hierarchy which could be exploited to improve the applicability of the method further. A significant improvement in efficiency could likely be achieved by computing the geodesic separations, necessary

for setting the bounding sphere radii, with a more specialised method. Using Dijkstra’s algorithm to compute the geodesic separations based upon edges is likely to generate paths, and hence radii, which are longer than the actual geodesic separation along the surface. If the bounding sphere radii are larger than required, then they will prune fewer individual proximity queries and therefore will lose efficiency. There are various methods within the literature which can follow polygonal faces in computing accurate geodesic distances [Novotni and Klein, 2002; Surazhsky et al., 2005], implementing one of these would be expected to improve the efficiency.

The unique way in which DIBS hierarchies implicitly follow the vertices of the underlying mesh could be used to reduce the computational demand of deforming the mesh itself. Within a large mesh, applying a deformation function to every vertex can be time consuming. If the deformation formulation allows, then ‘lazy evaluation’ of the displaced vertices could be incorporated into the DIBS hierarchy traversal. By only deforming each vertex when its position is called for by the DIBS hierarchy, the number of vertex updates could be greatly reduced. This optimisation could not be applied to most conventional bounding volume hierarchies as all the vertices must be speculatively updated to refit the bounding volumes before the actual tool position is even considered.

A final attribute of DIBS hierarchies which warrants consideration is their application to point clouds. Point clouds are generally well suited to dynamic active constraints because they are easily generated by commercially available range cameras. Though meshes were seen as the ideal method to represent a constraint surface, DIBS hierarchies are also well suited to use with point clouds. The only change that would be required to convert the method described within this chapter from meshes to point clouds is the computation of the bounding sphere radii. Clearly surface or edge based methods, like Dijkstra’s algorithm, could not be used to compute the geodesic distance from a point cloud, therefore an alternative, such as is described in [Mémoli and Sapiro, 2001], would have to be used. However, all other details and methods as described could be translated directly.

3.7 Summary

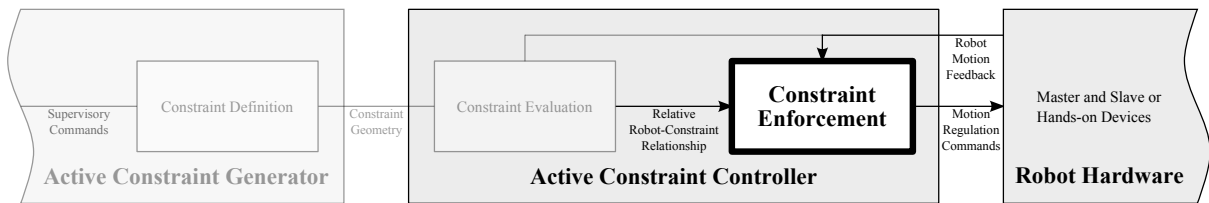
Within this chapter, the challenge of computing the relative configuration between a constrained tool and a constraint geometry has been addressed. When applying a dynamic active constraint, the constraint geometry is liable to deform regularly, causing many conventional methods for computing the necessary relationship to become inefficient. Due to the high demands for control rates in haptic applications,

inefficiency in computing this relative configuration can restrict the resolution with which constraint geometries can be constructed, potentially limiting their effectiveness.

A novel bounding volume hierarchy has been proposed which exploits the limits on the surface deformation of soft tissue structures, so that it implicitly follows a deformation, without requiring the conventionally computationally expensive rebuilding or refitting processes. Having proven that the method is significantly faster at computing the required relationship when the deformation rate in the tissue is high, the next element required of an active constraint controller is the constraint enforcement. A novel constraint enforcement method, based upon a dissipative friction model, is described in the following chapter.

Chapter 4

Dynamic Frictional Constraints



In this chapter, a new haptic control algorithm for the enforcement of dynamic active constraints is discussed. The formulation is based on the application of a dissipative friction model, which ensures that the surgeon retains control of the procedure. The research presented within this chapter is an edited version of research previously published in:

S. A. Bowyer, and F. Rodriguez y Baena, “Dynamic frictional constraints for robot assisted surgery”, in Proceedings of the IEEE World Haptics Conference, pp. 319-324, April 2013. ©2013 IEEE.

4.1 Introduction

It was shown in chapter 3 that it is currently possible to compute the relative configuration between a robot/tool and a continuously deforming constraint geometry sufficiently fast for use in a stable haptic application. Once the relative configuration is known by an active constraint controller, it is necessary to compute, from this relationship, a command which can be sent to the robotic hardware such that it enforces the desired constraint behaviour. Within chapter 2 this component of an active constraint controller was referred to as ‘constraint enforcement’.

While considering the available literature describing methods for constraint enforcement on impedance devices in dynamic environments (as described in section 2.7.3), it was noticed that they all exhibited a common haptic characteristic which is potentially undesirable, or even unacceptable, in delicate environments such as are found during robot assisted surgery. This property is that they are not guaranteed to be energetically passive and have the potential to cause autonomous movements in the robotic hardware.

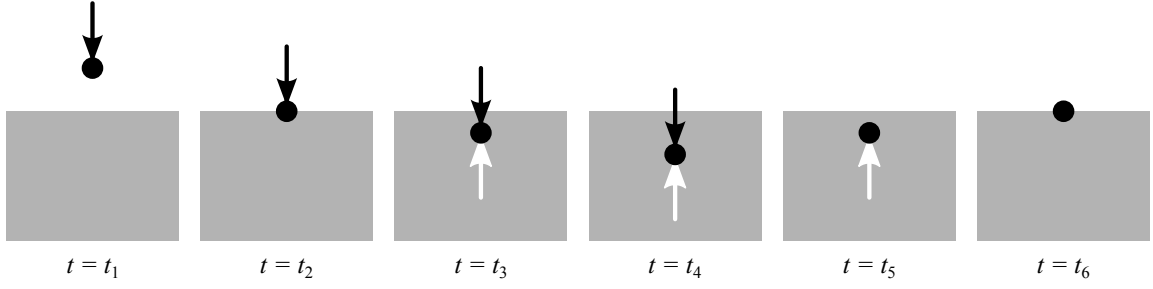
The dynamic active constraint controllers within the literature [Kwok et al., 2013; Navkar et al., 2012; Petersen and Rodriguez y Baena, 2013; Ren et al., 2008; Rydén and Chizeck, 2012] can be considered as variants of position controllers which attempt to prevent a tool from penetrating a constrained region. Within a dynamic active constraint there are two base cases for the form of the interaction between a tool and constraint:

- **Case 1** the user moves the tool into the constrained region, whether intentionally or accidentally.
- **Case 2** the user remains stationary while the constrained region (due to changes in the environment) moves past the tool.

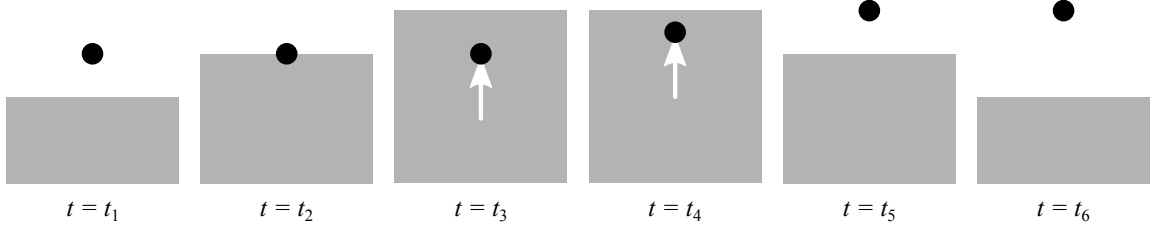
The response of a conventional dynamic active constraint in the two interaction cases is illustrated in figure 4.1. In the first case, the methods respond in the way of a conventional, static active constraint, whereby they anisotropically resist the penetrating component of motion, until the user retracts back into the unconstrained region. In the second case, the methods make no distinction from case 1, and attempt to force the tool to follow the moving constraint surface as though the user had pushed the tool in.

By not distinguishing between the two interaction cases, conventional dynamic active constraints exhibit behaviour which is inconsistent with that of conventional, active constraints as presented within the literature. Specifically, during the second interaction case, conventional dynamic active constraints can be energetically active such that the tools move without intention or action from the human user. This behaviour is henceforth referred to as ‘active forcing’ of the tools. Instinctively, active forcing can be understood by considering the constraint motion in figure 4.1b. In this example, as the constraint surface moves past the tool, the constraint enforcement generates robot commands which push the tool along with the moving surface. During the interaction, the user applies no force, and hence no energy, to the tool, therefore all of the energy to move the tool must have come from the robotic system itself.

Adams and Hannaford showed that a two-port network can be used to characterise a haptic interaction with a virtual environment and define conditions necessary for stability [Adams and Hannaford, 1999]. This representation of the haptic interaction with a dynamic active constraint can give a more



(a) Case 1. The user moves the tool into the constraint between time steps t_1 and t_4 . The constraint resists the user between time steps t_3 and t_5 until the user retract from the constraint at t_6 .



(b) Case 2. The user remains compliant across all time steps while the constraint boundary moves up and down. Once the constraint boundary moves past the tool, the constraint pushes the tool upwards between time steps t_3 and t_5 . Once the constraint boundary recedes between time-steps t_5 and t_6 , the tool is left within the unconstrained region. The net movement of the tool between time steps t_1 and t_6 has not been caused by the intention of the human user and represents energetic activity within the robot itself.

Figure 4.1: Illustrations of the two base interaction cases for conventional dynamic active constraints. The white region represents the unconstrained region, the grey region represents the constrained region and the black circles represent the tool positions at six discrete time steps. The black arrows represent user applied forces and the white arrows represent constraint forces.

formal understanding of the energetic properties of active forcing. In their analysis, Adams and Hanaford considered the haptic interface (the device itself and associated virtual couplings) as a two-port attached to two one-ports representing the human user and the virtual environments. By considering the energy transfer between components in terms of forces (efforts) and velocities (flows), and by applying Llewellyn's stability criteria, they showed that passive and stable virtual couplings can be defined which are independent of the virtual environment. However, it is necessary for the virtual environment to be passive within this analysis. The corollary of this is that a passive haptic interface does not guarantee system passivity if the virtual environment with which it interacts is not itself passive.

A two-port network for a dynamic active constraint system is shown in figure 4.2. In this way, a conventionally defined dynamic active constraint with a passive virtual coupling between the constraint and the robot is not guaranteed to be stable or, of particular interest with regards to active forcing, energetically

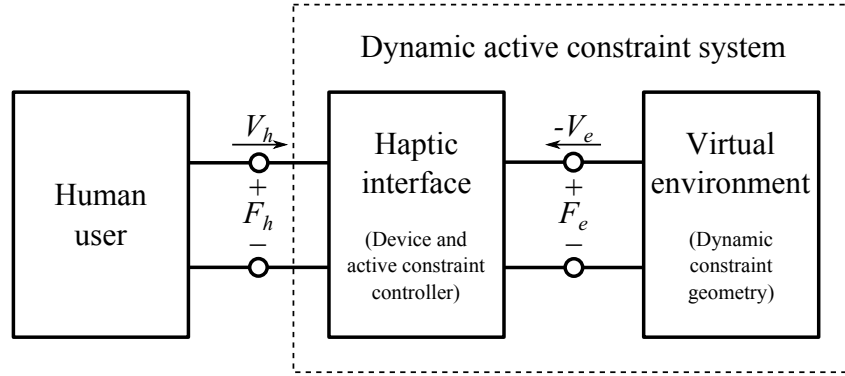


Figure 4.2: Two-port network to illustrate the energetic interaction between a user and a dynamic active constraint system. V_e and V_h are the virtual environment and human velocities, and F_e and F_h are the virtual environment and human forces respectively. Figure adapted from [Bowyer and Rodriguez y Baena, 2013]. ©2013 IEEE.

passive.

It is posited in this thesis that the energetically active behaviour exhibited during active forcing of the tools negates a fundamental property of a typical active constraint system, namely that overall control of the tool changes from the human surgeon to the robot.

As detailed in chapters 1 and 2, the benefit of collaborative robot control strategies, such as active constraints, is that they allow for the combination of human and robot competencies in such a way that it maximises overall performance. The key contribution from a human user in a collaborative control system is the ability to interpret, process and make complex decisions based upon disparate and unstructured information. Within a surgical environment, a human surgeon is able to observe extensive amounts of information about the patient and decide how to continue with a procedure to maximise the clinical outcome. Robots can contribute otherwise unachievable accuracy and precision, and can integrate with other imaging and sensing modalities seamlessly; however, they cannot currently approach the decision making capabilities of a human. All three of the commercially successful surgical robots in widespread clinical use at present (da Vinci Surgical System, Intuitive Surgical Inc.; Sensei X, Hansen Medical Inc. and Robotic Arm Interactive Orthopedic System, MAKO Surgical Corp.) conform very strongly to this non-autonomous, collaborative control concept, whereby the human surgeon has ultimate control of the tool motion and the robots assist with precision, accuracy and the integration of additional information.

Within conventional active constraints in surgery, the surgeon remains in overall control of the tool, deciding how best to perform the procedure, while the robot monitors the scene and attempts to regulate

the tool motion when it diverges from the plan. Active constraints in this form can be considered as ‘reactive’ to the actions of the surgeon such that they identify and nullify surgeon movements which are deemed incorrect.

When a conventional dynamic active constraint causes active forcing it is no longer reactive to the actions of the surgeon, instead, it is actively generating tool motion of its own volition. The hypothesis followed in this work is that, at present, the human surgeon’s understanding of the surgical environment and procedure is that much greater than that of the robot that the active constraint should always remain subservient to the human, offering guidance where necessary but never taking control for itself. Though tracking and imaging systems can be developed with high accuracy and reliability, the potential for a momentary error means that a constraint could suddenly change position in an incorrect and unpredictable way with respect to the true anatomy. If this happens, active forcing of the surgical tools could push the surgical instruments into, and damage, tissue structures without any intention from the surgeon. A similar issue can occur when the human user has identified a delicate anatomical structure which the robot has not, if another constrained surface moves during the procedure, the active forcing could again cause unintended motion towards this unidentified tissue, damaging it.

In addition to general safety concerns, active forcing, and the autonomous tool motion it represents, presents several other issues within surgery. One of these issues is that the potential for autonomous tool motion is likely to make surgeons and patients uneasy and non-accepting. Historically, surgery has been performed by highly skilled human experts who train for many years to develop their skills. The perception, whether unfounded, is that this skilled surgeon is being replaced by an unstable and volatile robot with the same capacity for crashes and failure as a home PC, and this is understandably worrying for the surgeon and patient. Further to this, if it can no longer be proven that the surgical tool motion is entirely controlled by the human surgeon, it is likely to make clinical certification for the robot more complicated. Another issue raised by autonomous tool motion is that legal liability for the patient’s safety is blurred. In a conventional active constraint system, the surgeon always controls tool motion and therefore they are always responsible for the patient. As soon as the robot autonomously controls the tool, the surgeon cannot solely be held fully responsible for the outcome.

There are potentially some situations where a dynamic active constraint with active forcing would be beneficial in ensuring tool accuracy in fully trackable dynamic environments; however, this form of semi-autonomy is, in the authors opinion, closer to that within motion compensation in teleoperation [Hagn et al., 2010] than active constraints. Indeed, the level of confidence within the tracking technology

is not thought to be high enough at present to justify clinical usage.

To ensure that dynamic active constraints offer the control characteristics of static active constraints under all conditions, a new formulation was required which guarantees energetic passivity, and hence overall surgeon control.

Friction has a key characteristic which makes it well suited to enforcing dynamic active constraints with the required properties. This characteristic is that friction forces always oppose the direction of motion and are therefore guaranteed to be passive. In fact, friction is actually dissipative [Willems, 2007], meaning that in addition to being passive, it only stores a bounded amount of energy. What dissipativity means for an active constraint is that it is even less likely to generate motion which is uncontrolled by the surgeon, for example when loosening their grip on the tool inside a constrained region.

This chapter describes the concept of an active constraint enforcement method, based upon friction, which allows accurate constraint enforcement within a dynamic environment, without the undesirable active forcing of the surgical tools. These proposed constraints are termed ‘dynamic frictional constraints’. A major challenge with active constraints based on friction is how to respond once the user has penetrated a constraint. The requirement for dissipativity means that the tool cannot be directly pushed back to the constraint surface. In dynamic frictional constraints, a novel concept called ‘frictional redirection’ is used whereby energy from the human user is redirected to guide them out of the constrained region in a fully dissipative way.

The research presented here is not the first disclosure of a constraint based upon friction. A method called ‘simulated plasticity’ was previously created by Kikuuwe, Takesue and Fujimoto [Kikuuwe et al., 2006, 2008]. This research, described in detail in section 2.6.3, uses proxy control laws to apply a large plastic resistance when the user pushes into a constrained region. While this method prevents constraint penetration effectively, it does nothing to reduce the magnitude of the penetration once it has occurred. The properties of simulated plasticity and dynamic frictional constraints will be contrasted within this chapter as necessary.

Due to the increased complexity of considering and constraining orientation, the application of constraints for three dimensional translations only will be discussed in this chapter. Applying constraints to orientation raises issues regarding avoiding active forcing, efficient energy utilisation and defining rotational constraint penetration. These areas are addressed in chapters 5 and 7, once the fundamental dynamic frictional constraint concept has been described here.

4.2 Formulation of Dynamic Frictional Constraints

4.2.1 Elastoplastic Friction Model

The elastoplastic friction model of Dupont *et al.* [Dupont et al., 2002], and the discrete implementation described in [Hayward et al., 2009], were used as the foundation for the proposed dynamic frictional constraints. This friction formulation maintains the simplicity of single state models such as Dahl [Dahl, 1968] and LuGre [Canudas de Wit et al., 1995], while preventing the problem of ‘non-physical drift’ during plastic presliding. The non-physical drift in these models is caused by the necessity in a simulation for an object to at least move by a small amount before a frictional force can be computed. In a simulation, this effect can mean that an object which keeps stopping and starting can gradually move in any direction with near-zero friction forces. Dupont *et al.* overcame this problem by enforcing an initial, purely elastic, displacement.

The elastoplastic friction model is based around the simulated system shown in figure 4.3. In this system, the displacement of an object under friction $x(t)$ consists of both plastic (i.e. permanent) $w(t)$, and elastic (i.e. reversible) $z(t)$ components. It is the elastic displacement which is used to compute the friction force $f_f(t)$ using the following equation, from [Dupont et al., 2002]:

$$f_f(t) = \sigma_0 z(t) + \sigma_1 \dot{z}(t) + \sigma_2 \dot{x}(t), \quad (4.1)$$

where σ_0 , σ_1 and σ_2 are the friction stiffness, tangential compliance damping and viscous friction coefficients respectively.

In the elastoplastic friction model, the change in the state variable, $z(t)$, is calculated for the one degree of freedom equation below:

$$\dot{z}(t) = \dot{x}(t) \left(1 - \alpha(z, \dot{x}) \frac{z(t)}{z_{ss}(\dot{x})} \right), \quad (4.2)$$

where α defines the type of displacement (elastic, plastic or mixed) and z_{ss} defines the steady state value for the elastic displacement (i.e. the limit to the magnitude of $z(t)$), and hence controls the steady state friction force. The function used to compute α ensures that the elastoplastic friction model has a fully elastic region, and as such, does not suffer from drift.

The discrete implementation of equation (4.2) is given in [Hayward et al., 2009] as two saturation oper-

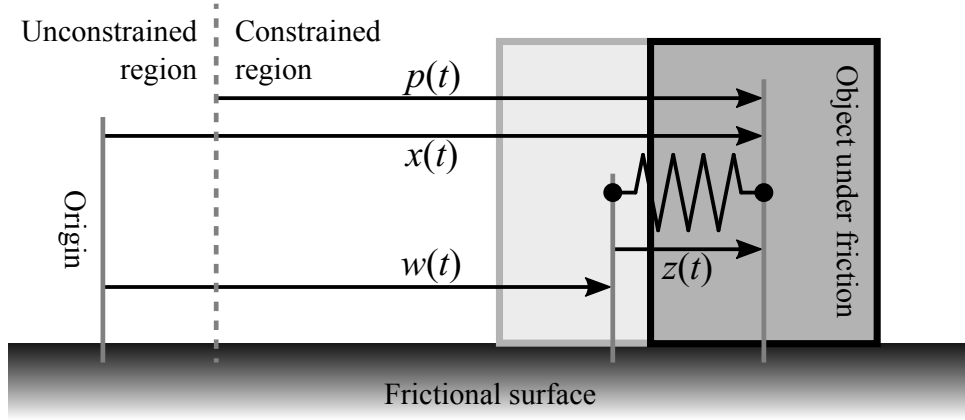


Figure 4.3: Illustration of the elastoplastic friction model created by Dupont *et al.* [Dupont et al., 2002]. The total displacement of the object $x(t)$ is the sum of the plastic $w(t)$ (i.e. permanent) and elastic $z(t)$ (i.e. reversible) components. Also shown is a regional constraint boundary and the constraint penetration $p(t)$. Figure reprinted from [Bowyer and Rodriguez y Baena, 2014]. ©2014 IEEE.

ations:

$$y_k = \begin{cases} \left[1 - \alpha(\hat{z}_{k-1}, \bar{v}_k) \frac{\hat{z}_{k-1}}{z_{ss}(\bar{v}_k)} \right] \Delta \bar{x}_k & \frac{\hat{z}_{k-1}}{z_{ss}(\bar{v}_k)} < 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.3)$$

$$\hat{z}_k = \begin{cases} \hat{z}_{k-1} + y_k & \frac{\hat{z}_{k-1} + y_k}{z_{ss}(\bar{v}_k)} < 1 \\ z_{ss}(\bar{v}_k) & \text{otherwise} \end{cases}, \quad (4.4)$$

where $\hat{z}_k$ is the discrete estimate of $z(t_k)$, $\bar{v}_k$ is the discrete estimate of $\dot{x}(t_k)$ and $\bar{x}_k$ is the discrete displacement $x(t_k)$.

4.2.2 Using Friction for Constraints

By simulating friction in all regions of a device's workspace where the user is not permitted, simple constraints can be enforced. Considering figure 4.3, this means computing the sign of the penetration value $p(t)$ and then, whenever this is positive, computing and applying a friction force to the user. In practice there are several aspects which must be considered for this method to work successfully.

4.2.2.1 Steady State Friction Force

The steady state friction force in the elastoplastic model is controlled by the steady state z value, z_{ss} . In [Hayward et al., 2009], three velocity dependent z_{ss} models are discussed which allow for realistic phenomena such as the Stribeck effect. It is possible that constraints could make use of such frictional

characteristics; however, at this stage only the simple Coulomb model [Malhotra et al., 1994] was necessary to construct a constraint. Making the simplification that the static friction force f_{max} is equal to the Coulomb friction force f_C in equations 6 and 9 from [Hayward et al., 2009], Coulomb friction z_{ss} is calculated from:

$$z_{ss}(\dot{x}) = \begin{cases} \frac{f_C}{\sigma_0} & \dot{x} \geq 0 \\ -\frac{f_C}{\sigma_0} & \text{otherwise} \end{cases} . \quad (4.5)$$

This friction model is, like the majority of physical friction, isotropic. For use in a constraint, this behaviour is not ideal and therefore the model was made anisotropic such that friction opposes the user when moving into a constraint, but not when retracting from a constraint. The convention used is that motion in the $+x$ direction is violating, and motion in the $-x$ direction is retracting from, a constraint.

To allow unopposed retraction from a frictional constraint, it should be ensured that z_{ss} , and hence $\sigma_0 z$, is greater than, or equal to zero at all times. This can be achieved with equation (4.5) by changing the lower subfunction from $-f_C/\sigma_0$ to zero. However, using $z_{ss} = 0$ causes the divisions in the clauses in equations (4.3) and (4.4) to fail as they require that the signs of $\hat{z}_{k-1}$ and $z_{ss}(\bar{v}_k)$ are different when there is a relaxation of the elastic displacement. By using a small, negative, constant value $-\varepsilon$ instead of zero, restructuring the complete elastoplastic model can be avoided. This gives the Coulomb friction z_{ss} expression:

$$z'_{ss}(\dot{x}) = \begin{cases} \frac{f_C}{\sigma_0} & \dot{x} \geq 0 \\ -\varepsilon & \text{otherwise} \end{cases} . \quad (4.6)$$

It is important to note that this is not equivalent to arbitrarily setting $\hat{z}_k = 0$ whenever $\dot{x} < 0$. This would allow a ‘slipping’ penetration of the constraint under oscillatory input forces. The saturation operation means that the user must fully retract from the elastic friction region before moving freely, ensuring the model will not suffer from this problem.

In the one dimensional friction system considered so far, the transition between constrained and unconstrained regions is guaranteed to be continuous. When moving into the constrained region, z , and hence the friction force, gradually increases from zero up to z_{ss} as the user moves through the elastic presliding region. When retracting from a constraint, the anisotropy in the model means that the user must have retracted from the friction before passing back across the constraint boundary unconstrained.

In the multi-dimensional system introduced below, it is not guaranteed that the friction acts orthogonally to a constraint boundary, and subsequently, it is possible for the user to cross the boundary while z is

non-zero. To prevent discontinuities as a result of this, the depth of constraint penetration p was used as a maximum limit to z_{ss} . By transitioning z_{ss} in this way, the smoothness of the controller is maintained at the boundary, while the initial stiffness is maximised. Incorporating this modification into equation (4.6), the completed function for computing z_{ss} in a frictional constraint becomes:

$$z_{ss}''(\dot{x}, p) = \begin{cases} \min\left(\frac{f_c}{\sigma_0}, p\right) & \dot{x} \geq 0 \\ -\varepsilon & \text{otherwise} \end{cases} . \quad (4.7)$$

4.2.2.2 Displacement Regime Function

The presliding regime within the elastoplastic friction model is determined by the function $\alpha(z, \dot{x})$. When α takes the value zero it corresponds to fully elastic presliding, when α takes the value one it corresponds to fully plastic presliding and when between zero and one it corresponds to mixed elastic and plastic presliding.

For frictional constraints, the function for α proposed in equation 10 of [Hayward et al., 2009] was used. As the value of z_{ss} can reach zero in the constraint formulation, a constant breakaway z value z_{ba} , as is used in [Hayward et al., 2009], was not appropriate as the breakaway value could end up being greater than z_{ss} . The breakaway z value defines the point at which the presliding moves from fully elastic to mixed displacement. Instead of a constant value, it was functionally defined as 95 % of z_{ss} . Clearly this no longer represents any physical phenomena, rather it just ensures that there is a smooth transition between plastic and elastic displacements. The high value of 95 % was selected for use in α to minimise non-physical drift, though further analysis would be required to ensure optimality. The updated α function is shown below.

$$\begin{aligned} & \text{if } \text{sgn}(z) \neq \text{sgn}(\dot{x}), \quad \alpha'(z, \dot{x}, q) = 0, \text{ otherwise} \\ & \alpha'(z, \dot{x}, q) = \begin{cases} 0 & \frac{z}{z_{ss}(\dot{x}, q)} \leq 0.95 \\ \sin\left(\pi \frac{|z| - 0.475z_{ss}(\dot{x}, q)}{0.1z_{ss}(\dot{x}, q)}\right) + \frac{1}{2} & 0.95 < \frac{z}{z_{ss}(\dot{x}, q)} < 1 \\ 1 & \text{otherwise} \end{cases} . \end{aligned} \quad (4.8)$$

The final version of the discrete friction model which incorporates the above modification is now given.

$$y_k = \begin{cases} \left[1 - \alpha'(\hat{z}_{k-1}, \bar{v}_k, \bar{q}_k) \frac{\hat{z}_{k-1}}{z''_{ss}(\bar{v}_k, \bar{q}_k)} \right] \Delta \bar{x}_k & \frac{\hat{z}_{k-1}}{z''_{ss}(\bar{v}_k, \bar{q}_k)} < 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.9)$$

$$\hat{z}_k = \begin{cases} \hat{z}_{k-1} + y_k & \frac{\hat{z}_{k-1} + y_k}{z''_{ss}(\bar{v}_k, \bar{q}_k)} < 1 \\ |z''_{ss}(\bar{v}_k, \bar{q}_k)| & \text{otherwise} \end{cases} . \quad (4.10)$$

The modulus in equation (4.10) is required to ensure that ε , rather than $-\varepsilon$, is used for $\hat{z}_{k-1}$ in the following step.

4.2.3 Frictional Redirection

4.2.3.1 Challenges of Parallel Motion

There are several ways in which the model in section 4.2.2 can be used to enforce practical constraints in multiple dimensions. At a constraint boundary, the necessary behaviour is reasonably clear. The friction forces should act orthogonally to the constraint boundary, in the direction of the unconstrained region. This allows the user to slide along the surface of the constraint while preventing movement inside. This can be achieved with the constraint formulation defined above by aligning the one dimensional friction model above to be parallel with the constraint penetration vector.

Once the user has penetrated the constraint, the necessary behaviour is less clear. The obvious approach is to keep the behaviour of the constraint like that at the surface, with the friction continuing to act orthogonally to the constraint surface. The characteristics of this form of frictional constraint are very similar to those exhibited by the simulated plasticity of Kikuuwe, Takesue and Fujimoto [Kikuuwe et al., 2008]. This approach to frictional constraint enforcement leads to unbounded, unopposed motion within the constrained region.

Figure 4.4 shows the constraint anisotropy in the unconstrained, boundary and constrained regions of a workspace. In the unconstrained region, there is no friction force and the user is unopposed in all directions. On the constraint boundary, as stated above, the friction force opposes motion orthogonally into the constraint but the user can slide freely along the surface. In the constrained region the anisotropy of the constraint is the same as that on the boundary, the effect of this is that the user is opposed when pushing further into the constraint, but is free to slide indefinitely along a surface parallel to the constraint

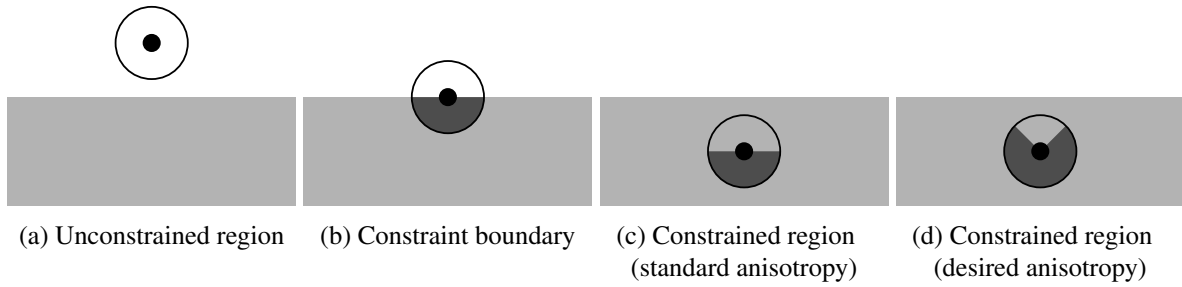


Figure 4.4: Illustrations of the anisotropy characteristics of simple frictional constraints in (a) unconstrained, (b) boundary and (c) constrained regions. The white region represents the unconstrained region, the light grey region represents the constrained region and the black points represent the tool positions. The black circles represent the frictional anisotropy such that dark grey directions are opposed by friction. Also shown is a desired constrained region anisotropy profile, (d), which prevents indefinite parallel movement within the constraint.

boundary but within the constrained region.

The research of Kikuuwe, Takesue and Fujimoto was focused on applying constraints to static environments, therefore, if the initial frictional force was high enough, the user could be prevented from becoming embedded within the constrained region. In this case the unconstrained parallel motion was less likely to be an issue; however, in a dynamic active constraint the problem is particularly acute, as the user will often become embedded when the constraint surface moves past the user.

A similar problem with unconstrained parallel motion was found by Bettini *et al.* during their investigation of active constraints in admittance control [Bettini et al., 2004]. In their development of reference direction fixtures (as described in section 2.6.5), they biased the subspace of preferred directions of motion towards to constraint boundary such that as the user moved sideways, they would be gently guided out of the constrained region.

The general concept for the solution in frictional constraints, adapted from that used in reference direction fixtures, is shown in figure 4.4d. Here, the angle of the directions in which motion is permitted is reduced from 180 degrees. This configuration means that the user cannot move completely parallel to the surface. Instead, some component of the user's motion must be in the direction of reducing penetration. The effect of this is that, assuming the user does not overpower them, the friction forces will monotonically reduce the magnitude of the constraint penetration until the user passes back into the unconstrained region. The angle of the directions in which motion is unopposed by friction controls how much parallel motion is permitted before the user must leave the constrained region. When the angle is small, very little parallel motion is permitted and the user must retract almost orthogonally to

the constraint boundary. When the angle is large, the user will be able to move parallel to the surface a long way before gradually being guided out of the constraint.

Some preliminary experiments were performed with an anisotropic frictional constraint similar to that described above. In this method, the one dimensional friction formulation from section 4.2.2 was aligned parallel to the direction of motion such that the friction force was always opposed to the direction of motion. By defining the Coulomb friction force f_C high (e.g. 5 to 10 N) when moving in the ‘red’ directions and zero when moving in the ‘green’ directions in figure 4.4d an anisotropic constraint was created.

This method was effective at stopping the user from moving into, and parallel to, the constraint, but an immediate issue arose in that it gave the user no haptic information as to which direction they should be moving in. The frictional forces would effectively communicate to the user that they were moving in the wrong direction, but, as the friction forces were always diametrically opposite to motion, they did not communicate in which direction the user should turn to leave the constrained region. This lack of communication range led to a frustrating process in which the user would try to search for the direction in which the constraint would stop opposing them.

Frictional redirection was conceived as a frictional constraint enforcement method which, rather than just stopping the user from moving in the wrong direction, communicates the desired direction of motion to the user by subtly redirecting the user’s motion, much like sliding along a ruler. To achieve this, it is necessary for the friction model to be oblique to the direction of motion. A method for achieving this frictional redirection, with the friction model described above, follows.

4.2.3.2 The Frictional Redirection Method

The ideal properties of a frictional constraint, for three primary penetration states, are defined as follows:

- **Orthogonal violation:** When the tool motion is primarily orthogonal to a constraint boundary, and is in the direction of increasing penetration, the friction acts to slow it and prevent further penetration.
- **Parallel violation:** When tool motion is primarily parallel to a constraint boundary the friction redirects the motion, pushing the user towards the constraint surface and reducing the penetration magnitude.

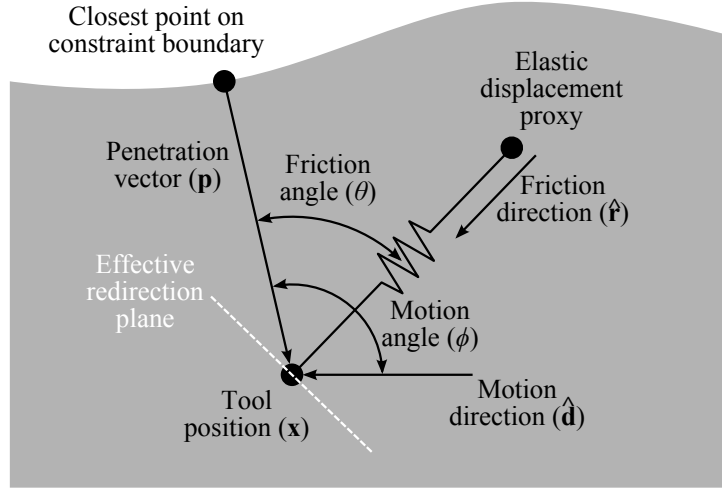


Figure 4.5: Illustration of the parameters used within the frictional redirection method. A tool is shown which has violated a constraint, as shown by the penetration vector $\mathbf{p}$. The frictional proxy and the elastic linkage represent the one dimensional elastoplastic friction model in figure 4.3, which acts upon the tool at $\mathbf{x}$. This effect of the friction model is considered as a redirection plane. Figure modified from [Bowyer and Rodriguez y Baena, 2013]. ©2013 IEEE.

- **Orthogonal retraction:** When tool motion is primarily orthogonal to a constraint boundary, and in the direction of decreasing penetration, the friction does not interfere.

By reorienting the one dimensional friction model in three dimensional space, the proposed constraint method aims to achieve the properties above. Figure 4.5 illustrates the terms used within the method. The Cartesian vector between the closest point on the constraint surface and the tool position $\mathbf{x} \in \mathbb{R}_3$ defines the penetration vector $\mathbf{p} \in \mathbb{R}_3$. The angle between the penetration vector and the normalised motion direction vector $\hat{\mathbf{d}} \in \mathbb{R}_3$ gives the motion angle $\phi \in \mathbb{R}$. The angle between the penetration vector and the normalised friction direction $\hat{\mathbf{f}} \in \mathbb{R}_3$ gives the friction angle $\theta \in \mathbb{R}$.

The influence of the friction model is considered as an ‘effective redirection plane’. This plane, shown in figure 4.5, is centred on the tool point and is orthogonal to the friction model. If the tool motion is orthogonal to the redirection plane (i.e. parallel to the friction direction), then the plane diametrically opposes motion. If, however, the motion is not orthogonal to the plane, then the tool will tend to ‘slide’ sideways. This lateral sliding is used to encourage the user to retract from a constraint.

The general scheme for orienting the redirection plane is shown in figure 4.6. Two constants are used to define the boundaries between the three penetration states defined above. When the motion angle ϕ is less than Φ_{min} , the motion is considered as in orthogonal violation. In this case the friction angle is set to 0 in order to slow, or prevent, further penetration. When $\Phi_{min} \leq \phi \leq \Phi_{max}$, the motion is considered as

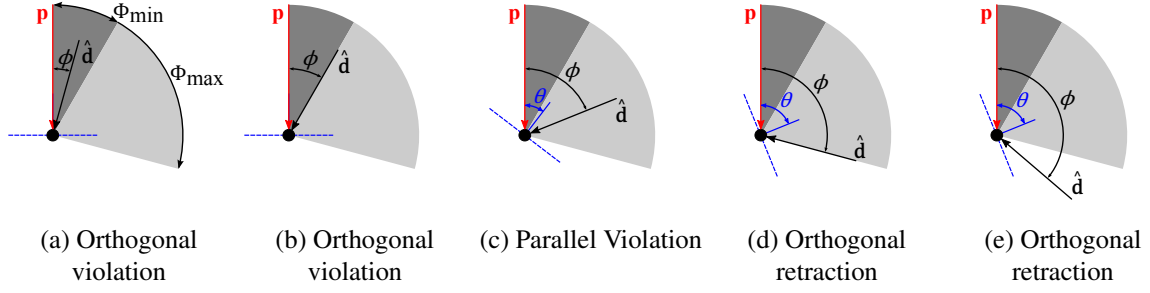


Figure 4.6: Sequence illustrating the three penetration states of the frictional redirection method. In (a) and (b), $\phi \leq \Phi_{min}$. To minimise the magnitude of further constraint violation, $\theta = 0$. In (c), $\Phi_{min} < \phi < \Phi_{max}$. To reduce the magnitude of violation, the user is redirected by setting $\theta = \phi - \Phi_{min}$. In (d) and (e), $\phi \geq \Phi_{max}$. To allow the user to retract freely, the friction angle is limited to $\theta = \Phi_{max} - \Phi_{min}$. Figure reprinted from [Bowyer and Rodriguez y Baena, 2013]. ©2013 IEEE.

in parallel violation. In this condition the redirecting plane begins to rotate, so that the user slides along it, towards the constraint boundary. The rotation continues until the motion angle is more than Φ_{max} . Once this happens, the motion is considered as being in orthogonal retraction and the redirection plane stops rotating, allowing the user to continue retraction freely.

It is intended that, once the user has violated a constraint, they will transition smoothly from orthogonal penetration to parallel penetration to orthogonal retraction and finally out of the constrained region. The numerical definition of the friction angle θ shown in figure 4.6 is as follows:

$$\tilde{\theta} = \begin{cases} 0 & \phi < \Phi_{min} \\ \phi - \Phi_{min} & \Phi_{min} \leq \phi \leq \Phi_{max} \\ \Phi_{max} - \Phi_{min} & \text{otherwise} \end{cases} \quad (4.11)$$

To ensure the continuity of the above redirection scheme at the constraint boundary, a transition region of size P_{trans} is introduced. This ensures that close to the boundary the redirection plane is parallel to the surface so that as the user retracts, they will slide freely along the surface without feeling a sudden removal of force:

$$\theta = \begin{cases} \tilde{\theta} \frac{\|\mathbf{p}\|}{P_{trans}} & \|\mathbf{p}\| < P_{trans} \\ \tilde{\theta} & \text{otherwise} \end{cases} \quad (4.12)$$

By rotating the normalised penetration vector $\hat{\mathbf{p}}$ through the angle θ , the Rodrigues' rotation formula can be used to calculate the required friction direction $\hat{\mathbf{r}}$ as follows:

$$\hat{\mathbf{r}} = \hat{\mathbf{p}} + \sin \theta (\hat{\mathbf{n}} \times \hat{\mathbf{p}}) + (1 - \cos \theta) \hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \hat{\mathbf{p}}), \quad (4.13)$$

where

$$\hat{\mathbf{p}} = \frac{\mathbf{p}}{\|\mathbf{p}\|}, \quad \hat{\mathbf{n}} = \hat{\mathbf{p}} \times \hat{\mathbf{d}}.$$

The scalar input properties for the one dimensional friction model are then computed from the three dimensional tool vectors:

$$\Delta \bar{x}_k = (\mathbf{x}_k - \mathbf{x}_{k-1}) \cdot \hat{\mathbf{r}}_k \quad (4.14)$$

$$\bar{v}_k = \dot{\mathbf{x}}_k \cdot \hat{\mathbf{r}}_k \quad (4.15)$$

$$\bar{q}_k = \|\mathbf{p}_k\|. \quad (4.16)$$

Using the above parameters with the constraint friction model in equations (4.1), (4.9) and (4.10) allows for the computation of the scalar friction force f_f . From this, the constraint force can be computed:

$$\mathbf{f}_k = \hat{\mathbf{r}}_k f_f. \quad (4.17)$$

4.3 Experiments

A dynamic path-following task was used to evaluate the efficacy of the proposed dynamic frictional constraint method. In this task, a simulated three dimensional scene (shown in figure 4.7) was presented to the user which consisted of a horizontal plane, upon which a two dimensional curved pathway was shown. Throughout the test, the plane oscillated vertically at 0.25 Hz, which is approximately the respiratory rate of a human adult, with an amplitude of 20 mm. The user was asked to follow along the pathway using a Phantom Omni (Sensable Technologies Inc.) as accurately as possible while compensating for the vertical motion. A total of 17 subjects took part in the experiment. This consisted of 15 (2) males (females), 15 (2) right (left) handed people and 13 (4) people inexperienced (experienced) with haptic simulations.

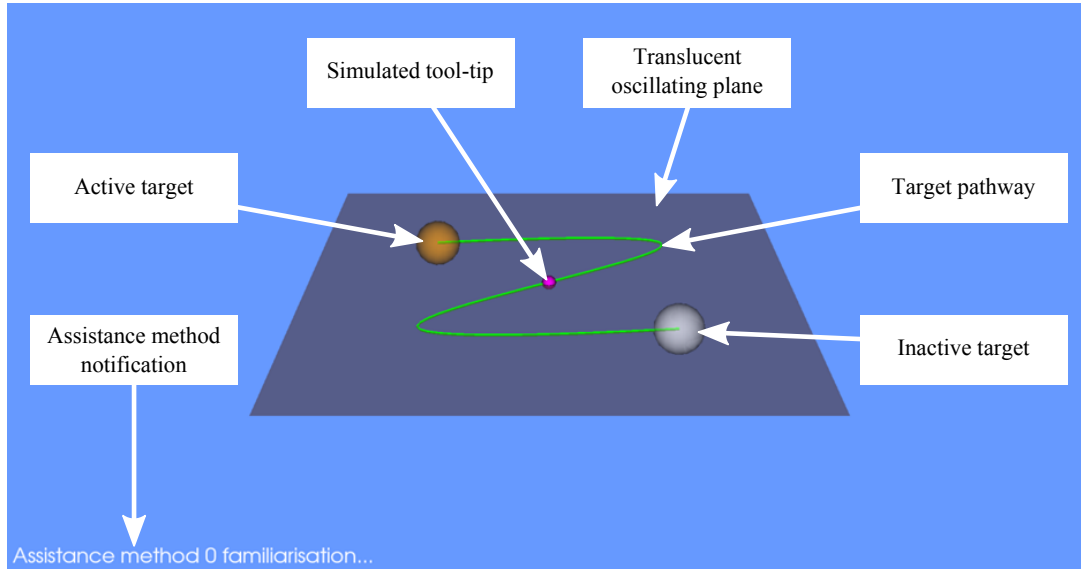


Figure 4.7: Annotated view of the display used within the experimental validation of the constraint strategies. Within the simulation, the user controlled the position of the simulated tool-tip with a Phantom Omni and attempted to follow the target pathway while it oscillated vertically. Figure adapted from [Bowyer and Rodriguez y Baena, 2013]. ©2013 IEEE.

Three assistance method classifications were considered within the experiments. These methods were defined as follows:

- **UC:** Unconstrained, no force feedback was applied and the user was required to follow the pathway using only visual feedback. This method was the experimental control and provided a baseline for the user's ability at the task.
- **VE:** Viscoelastic force feedback, given by $\mathbf{f}_{ve} = k_{ve}\mathbf{p} + c_{ve}\dot{\mathbf{x}}$, was applied, which pushed the user towards the closest point on the pathway. This method was the benchmark for task accuracy as it firmly held the user on the pathway.
- **DFC:** Dynamic frictional constraint force feedback was applied based on the method described in section 4.2. The penetration vector was computed from the tool tip to the closest point on the pathway.

The experimental scenario, control and visualisation were all implemented in a multi-threaded C++ application on a 3 GHz Core 2 Duo CPU (Intel Corp.). The multi-threading of the application meant that the control algorithms ran independently of the visualisation and at a consistent, though not real-time, synchronous rate. The haptic rate for all three assistance methods was 1 kHz and the display rate was approximately 25 Hz. Where it was required, the closest point on the pathway to the tool was

Parameter	Value	Parameter	Value	Parameter	Value
f_C	3.5 N	Φ_{min}	75°	k_{ve}	0.7 N/mm
σ_0	0.7 N/mm	Φ_{max}	105°	c_{ve}	10^{-4} N s/mm
σ_1	0 N s/mm	P_{trans}	0.5 mm		
σ_2	10^{-5} N s/mm				

Table 4.1: Constraint parameters used within the experimental procedure. The Coulomb friction force f_C was the maximum continuous force of the selected haptic device and the remainder of the parameters were determined experimentally. Table adapted from [Bowyer and Rodriguez y Baena, 2013]. ©2013 IEEE.

computed by exhaustively searching a linearised version of the pathway.

The force parameters for methods VE and DFC were tuned experimentally to provide the maximum stable stiffness, and the frictional redirection parameters were empirically selected to reduce constraint penetration while ensuring smooth progression. The final values for methods VE and DFC are shown in table 4.1. Preliminary validation found that σ_1 was not necessary for the stability of the proposed system and was therefore set to zero.

Once each test subject had read the experimental instructions, the functions of the three methods were explained, and they were allowed a training session to become familiarised with each. After the training, the subjects performed the task on ten pathways with each of the three assistance methods (30 tests per subject in total). The order of both the assistance methods and pathways was randomised to reduce any training bias across all subjects.

4.4 Results

Throughout each test, the user’s motion, positional error from the closest point on the pathway and haptic forces were logged. Three performance metrics were used to assess the assistance methods: root mean square error (RMSE); mean velocity and mean power generated by the haptic device. The means and standard deviations for each assistance method with these three metrics are given in table 4.2.

One-way analysis of variance (ANOVA) tests were applied to each metric, showing that they were all significantly ($p < 10^{-6}$) affected by the type of assistance method. The distributions produced as part of the ANOVA test are shown in figure 4.8. Tukey’s honest significant difference test [Steel et al., 1997] was then used to perform pairwise comparisons. These results, given in table 4.3, showed that with a 95 % confidence interval all methods were significantly different on all metrics.

	RMSE		Mean Velocity		Mean Power	
	Mean	Stdev	Mean	Stdev	Mean	Stdev
UC	20.07 mm	15.14 mm	24.93 mm/s	11.34 mm/s	0 W	0 W
VE	0.67 mm	0.52 mm	75.54 mm/s	50.77 mm/s	1.17 W	1.78 W
DFC	4.24 mm	4.25 mm	46.88 mm/s	29.28 mm/s	-2.25 W	2.52 W

Table 4.2: Results for the three assistance methods with the three evaluation metrics. The unconstrained test case is denoted UC, the viscoelastic test case is denoted VE and the dynamic frictional constraint test case is denoted DFC. The results shown for each test are the mean and standard deviation of a total of 170 test runs.

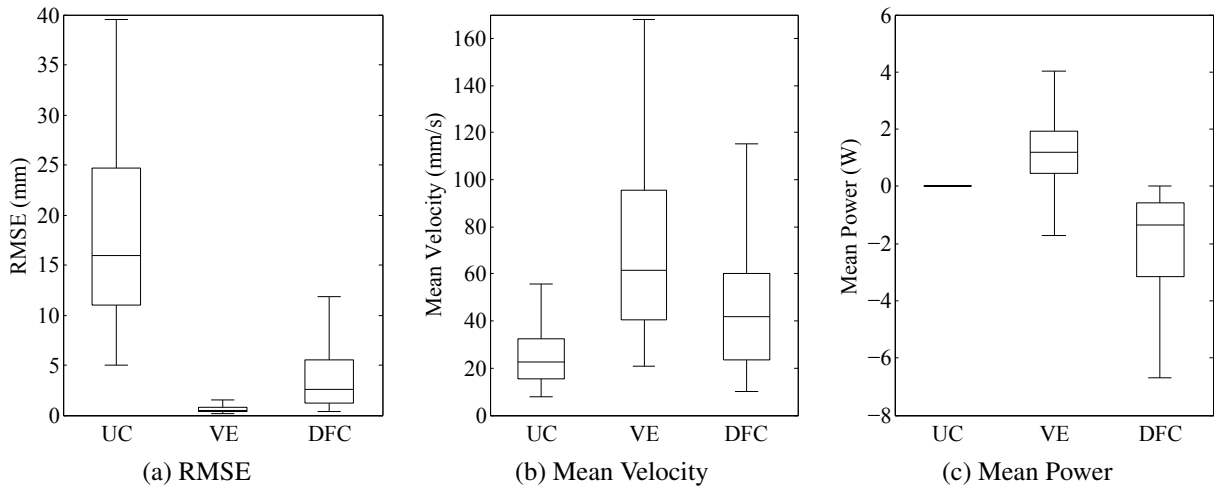


Figure 4.8: Experimental results of the three performance metrics for 17 test subjects. The unconstrained test case is denoted UC, the viscoelastic test case is denoted VE and the dynamic frictional constraint test case is denoted DFC. The outliers are not shown for clarity. Figure adapted from [Bowyer and Rodriguez y Baena, 2013]. ©2013 IEEE.

	RMSE	Mean Velocity	Mean Power
UC vs VE	**	**	N/A
UC vs DFC	**	**	N/A
VE vs DFC	**	**	**

Table 4.3: Results of Tukey's honest significant difference test. The symbol '**' identifies a constraint method pairing, and performance metric, for which a significant difference was identified at the 95 % confidence interval. Mean power is always zero for the unconstrained case and therefore comparisons in this category are not applicable. The unconstrained test case is denoted UC, the viscoelastic test case is denoted VE and the dynamic frictional constraint test case is denoted DFC.

4.5 Discussion

Within the above experiments, the RMSE indicates the efficacy of the constraint at its primary task of keeping the user out of the constrained region. The results show that the viscoelastic constraint has the lowest mean RMSE, while the frictional constraint is higher and the unconstrained case is the highest.

The very narrow distribution for the viscoelastic constraint is typical of the way in which it firmly controls the tool, forcing the user to follow the path. Even the least able users found the viscoelastic method easy to use, and it took significant, intentional, effort to circumvent it and move off of the pathway. Though the mean is approximately 3.5 mm higher, in some cases the dynamic frictional constraint approaches the high performance RMSE values of the viscoelastic method. However, the larger range for the frictional constraint implies that user ability is still an important factor in performance. This result conforms the concept behind dynamic frictional constraints; they cannot force an unskilled user to move in the right direction, only prevent them from moving in the wrong direction.

The distribution for the unconstrained cases is very wide; however, even the lowest values are higher than most of those achieved with the frictional constraints. Importantly, this shows that even the more skilled users gain a benefit from the subtle redirection in the frictional constraint.

In this experiment, the mean velocity during the experiments is believed to indicate how confident and comfortable the users felt within the task. The tests with the viscoelastic constraint had the highest mean velocities, with the frictional constraint tests slower and unconstrained tests as the slowest.

From observing the test subjects it is unsurprising that the viscoelastic constraint produced the fastest tests. In many cases, once the test subjects were comfortable with the force feedback, they performed the task while almost disregarding the visual feedback entirely. However, the level of confidence offered by the viscoelastic constraint could actually be detrimental within a clinical setting. In an environment where unpredictable complications and changes can arise very quickly, having a surgeon rely too much on the robotic assistance could prevent them from promptly responding to such cases.

The improvement in speed from the unconstrained to the frictional constraint cases implies that the users have some confidence in the guidance provided by the method; however this is not as much as in the viscoelastic case. It should be noted that the frictional redirection method actually leads to improved accuracy when the user moves faster. As the user's speed increases, they put more energy into the system, which the constraints can use to provide strong guidance, therefore, it is clear that faster user

motion tangential to the pathway will result in faster redirection and reduced constraint penetration. Obviously it is not appropriate to encourage surgeons to move faster, therefore this attribute should be considered as an incidental side effect rather than a particular advantage.

The mean power generated by the haptic device during the experiments establishes whether the assistance method was generally generating energy (positive mean power values) or dissipating energy (negative mean power values). Positive power values represent the active forcing of the tool which, it is hypothesised, is unacceptable for application to surgery. Negative power values, conversely, represent overall user control of the tool and are considered acceptable.

The power distribution for the viscoelastic assistance method shows that a significant amount of energy is generated by the dynamic nature of the constraint and transferred to the human user. All occurrences of positive haptic power are cases where the tool motion is not entirely controlled by the human user. As expected, the distribution for the frictional constraint shows that it dissipates energy from the system, and as a result the human user is guaranteed to be in full control of the tool. As there is no force, the power values for the unconstrained tests are always zero.

4.6 Summary

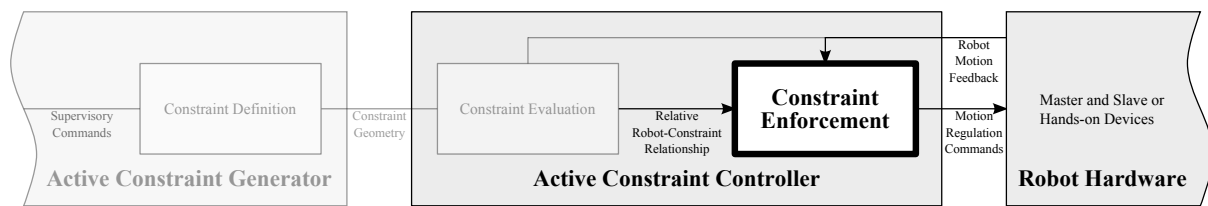
The experimental results for the proposed dynamic frictional constraints show that an energetically dissipative dynamic active constraint, which does not generate active forcing within surgical tools, is a viable alternative to the conventional methods presented within the literature. Clearly, the efficacy of the dynamic frictional constraint, as quantified by the RMSE, is not as high as that for dynamic active constraints of the conventional form; however, they allow some skilled users to achieve an equivalent performance. Additionally, these experiments have proven the expected result that conventional active constraints do indeed generate energy when they interact with a dynamic environment.

In order to apply dynamic active constraints in complex soft tissue environments, it is generally not sufficient to only constrain the tip of the tool in translation. Instead, to ensure that the user is most effectively assisted in controlling the entire body of the surgical instrument, constraints upon orientation are also necessary. Constraining orientation involves several difficulties not present for position, particularly with regards to the unique energetic formulation of dynamic frictional constraints. In addition to this, though it has been shown experimentally, it has not yet been proven that the dynamic frictional

constraint formulation is fully dissipative under all conditions. These two areas are the primary focus of chapter 5.

Chapter 5

Dynamic Frictional Constraints in Translation and Rotation



In this chapter, the dynamic frictional constraint formulation from chapter 4 is reformulated and extended for simultaneous application to both translation and rotation. The dynamic frictional constraints are able to make use of energy from the surgeon in any of the six dimensions and generate rich haptic feedback. The research presented within this chapter is an edited version of research previously published in:

S. A. Bowyer, and F. Rodriguez y Baena, “Dynamic frictional constraints in Translation and Rotation”, in Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2685-2692, June 2014. ©2014 IEEE.

5.1 Introduction

In chapter 4, the foundation of a dissipative dynamic active constraint was established which ensured that a user remains in overall control of a constrained tool’s motion when a constraint geometry deforms and moves. The experimental evaluation of the method showed its efficacy at guiding a user in a simple path following task; however, the practical applications of the method proposed are limited. With the

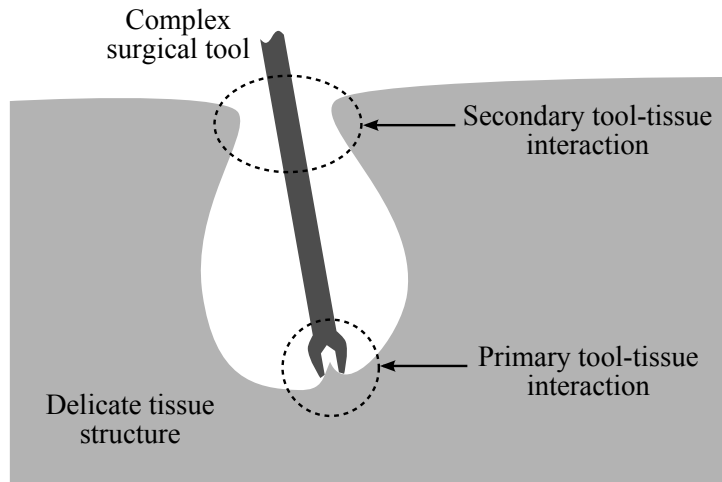
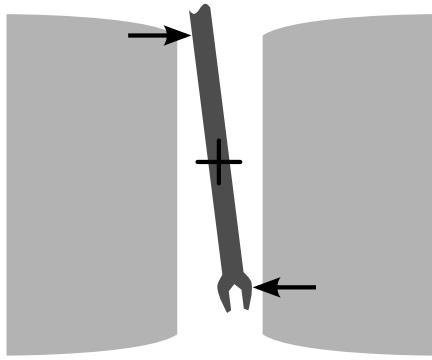


Figure 5.1: Illustration of a surgical scenario in which both the position and orientation of the complex surgical tool (i.e. one which cannot be represented by a rotationally invariant geometry, such as a point or sphere) must be controlled to ensure successful and safe operation. Though translational motion may be the dominant factor in performance at the primary tool-tissue interaction site, the orientation of the tool can cause contact with, and therefore damage to, the delicate tissue structure at the secondary tool-tissue interaction site.

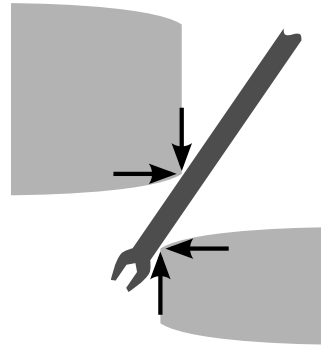
exception of a few surgical procedures such as milling with a spherical cutter [Ho et al., 1994], the overwhelming majority of the applications to which active constraints can be applied involve tool geometries which are not rotationally invariant. That is to say that, both the position and orientation of the tool must be considered in order to perform the procedure effectively and safely.

A surgical configuration in which both the position and orientation of a tool are important is illustrated in figure 5.1. It can be seen that simply constraining the tool tip is not sufficient to protect the tissue from damage at the secondary interaction location and therefore the tool's orientation must be considered. If a surgical task involves a tool with a complex geometry requiring orientation control, then any attempt to implement active constraints within the environment must also consider the tool geometry and its orientation. This means that the active constraint controller must consider the relative configuration between the constraint and the entire tool body.

It is possible to implement a 3-D translational active constraint in such a situation by simply summing the constraint interactions across the tool's entire body; however, the haptic characteristics of such a constraint are poor. The main problems with this formulation can be understood by considering figure 5.2. In figure 5.2a, if the user twists the tool in the anti-clockwise direction, the top and bottom of the tool will move towards the constrained regions on the left and right sides respectively. Crucially, they will penetrate the constraints in opposite directions with approximately equal magnitudes. The constraint



(a) In this configuration, as the tool rotates about the centre point, the effect of the two opposing constraints will cancel out. The result is the absence of a net constraint upon the user's motion.



(b) In this configuration, the constraints will resist the user's motion in any direction, so that the constrained device would essentially feel 'stuck' in place. To extract the tool from the constraint, it is necessary to rotate the tool; however, a translation only constraint would not communicate this to the user.

Figure 5.2: Illustrations of two active constraint configurations in which a translation only constraint would suffer from poor performance. The light grey regions identify constraint geometries, the dark grey shape represents a complex tool and the arrows represent the directions in which the constraints would resist motion in each scenario.

interaction from either end of the tool will therefore cancel out, giving an almost zero net constraint while the user twists the tool through the constrained region, potentially damaging the tissue. Another problem with a purely translational constraint is illustrated in figure 5.2b. In this configuration, the user cannot move the tool in any direction without moving towards a constraint. It would be necessary for the user to twist the tool to extract it from the constraint; however, as the user would only feel forces in any direction in which they moved, there would not be any haptic cues to alert the user of the need to rotate the tool.

Due to these limitations, in general, it is necessary for rotational active constraints to be applied to a tool if control of its orientation is required to successfully perform a task. In this context, rotational constraints can be considered as generating robot torques in an impedance controller or robot angular velocities in an admittance controller.

There are some active constraint methods within the literature which can be applied to orientation (as surveyed in chapter 2); however, there are surprisingly few, considering the prevalence of applications in which tool orientation is important. There are several likely causes for this lack of research focus. One important reason is that translation and rotation are inherently decoupled from one another, with different representations and units. This means that considering them in a unified way is difficult. Ad-

ditionally, many commercial haptic devices used in active constraint research (e.g. the popular Phantom Omni, Sensable Technology Inc.) do not have actuated control of orientation and therefore cannot apply rotational constraints. A final reason for the apparent lack of significant rotational constraint research is that regional constraints exist entirely in $\mathbb{R}^3$ space, and therefore the concept of “rotational constraint penetration” is unclear. It is necessary to implement a more complicated system which first defines an optimal pose from the contacting constraint surfaces, given a tool geometry and interaction scenario. This specific aspect of a rotational constraint is investigated in chapter 7.

Active forcing of tools due to motion within a dynamic active constraint is just as prevalent in rotation as it is in translation. Considering the constraint configuration shown in figure 5.2a, if the constraint walls were to move inwards towards the tool, then a conventional constraint would generate rotational motion which was not initiated by the user. It is posited that active forcing is just as unacceptable in rotation as it is in translation, and therefore a dynamic frictional constraint which operates in both translation and rotation is necessary.

In this chapter, a formulation for a dynamic frictional constraints in translation and rotation is proposed, based on the foundation established in chapter 4. There are two primary components to defining a dynamic frictional constraint which operates in six dimensions, i.e. $SE(3)$, the special Euclidean group of rigid body displacements in translation and rotation in three dimensional Euclidean space.

The first of these components is establishing a method to implement frictional redirection in $SO(3)$, the special orthogonal group of rigid body rotations in three dimensional Euclidean space. The dynamic frictional constraint formulation is based upon applying frictional resistance in a direction in Euclidean space; however, rotational representations in $SO(3)$ are non-Euclidean and therefore require reformulation. To achieve a unified constraint formulation which can implement frictional redirection in $SE(3)$, the method from chapter 4 is first reformulated in a more generalised way in Euclidean space $\mathbb{R}^3$. It is then shown how rotations in $SO(3)$ can be mapped into Euclidean space $\mathbb{R}^3$ such that the same constraint formulation, with frictional redirection, can be applied to rotation.

The second challenge in defining dynamic frictional constraints in $SE(3)$ is linking the naturally decoupled components of motion in translation and rotation to one another. In the initial stages of this research, dynamic frictional constraints were fundamentally applied as two independent constraints, one upon translation and one upon rotation. This was found to be reasonably effective; but the improvement over just translational constraints was small. The concept of a coupled dynamic frictional constraint is

therefore introduced, which extends the concept of frictional redirection to redirect translational energy into rotation, and vice versa, in order to most effectively constrain the user's motion.

5.2 Extending Dynamic Frictional Constraints to SE(3)

In chapter 4, 3-D constraints were achieved by rotating an anisotropic version of the 1-D elasto-plastic friction model [Hayward et al., 2009] in 3-D space, in such a way that it minimised constraint penetration. For two reasons, it was decided that dynamic frictional constraints should be reformulated for further development, while maintaining their novel energetic characteristics.

The primary reason for the reformulation was that the equations for the 1-D elasto-plastic friction model, as used in chapter 4, relied on scalar constants for properties such as the Coulomb friction force (f_C) and the elastic stiffness (σ_0). Within SE(3), there are different values and units for these constants in translation and rotation, making computing a single unified value for the model complicated. Moreover, when the stiffness is anisotropic (i.e. different in translation and rotation), rotating the 1-D model in 6-D space would not necessarily remain dissipative, or even passive. The potential for energetic activity comes from the fact that springs with the same extensions, and different stiffnesses, store different amounts of potential energy. This means that, if a certain amount of energy is stored in the 1-D elastic presliding region (given by the state variable $z(t)$) when it points in one direction, and then the model is rotated into a different direction, such that the effective stiffness increases, then the potential energy stored in the presliding region will increase without any energy being input from the user. In the coupled SE(3) constraint formulation, presented later in this chapter, this issue is dealt with by applying a mapping to the space in a way which could not be easily achieved using the method in chapter 4.

The second reason for the reformulation was that it made proving the dissipativity of the model much more straightforward. The reorientation of the 1-D model in chapter 4 introduced a non-linearity into the model which made such a proof significantly more difficult.

This section shows how a dynamic frictional constraint in SE(3), based on the properties described in chapter 4, can be formulated from the elasto-plastic friction model. The constraint formulation is developed in four stages: first, it is shown how the elasto-plastic friction model can be characterised in SE(3); then, it is shown how the 1-D elasto-plastic friction model can be simplified and applied isotropically in $\mathbb{R}^3$; next, it is shown how the isotropic formulation in $\mathbb{R}^3$ can be modified to apply

anisotropic, dynamic frictional constraints; and finally, it is shown how rotations in $SO(3)$ can be mapped into $\mathbb{R}^3$ such that the previously explained dynamic frictional constraints can be enforced upon them.

In this section, the dynamic frictional constraints in translation and rotation are enforced using the same framework; however, they are applied independently of one another. Each of them exhibits frictional redirection between their three respective degrees of freedom, but, there is no frictional redirection of energy from one to the other. A method for achieving full, coupled dynamic frictional constraints in $SE(3)$ is explained in the following section.

5.2.1 Elasto-plastic Friction Model in $SE(3)$

To extend the conventional, 1-D elasto-plastic model (as shown in figure 4.3) into 6-D, each of the displacement components (i.e. $x(t)$, $w(t)$ and $z(t)$) must be represented in the special Euclidean group $SE(3)$. In this work, the total displacement of an object from an inertial frame in $SE(3)$ is defined as $\mathbf{x}(t) \in \mathbb{R}^6$. $\mathbf{x}(t)$ and its time derivative $\dot{\mathbf{x}}(t)$ are composed as follows:

$$\mathbf{x}(t) = [\mathbf{x}_t(t), \mathbf{x}_r(t)] \quad (5.1)$$

$$\dot{\mathbf{x}}(t) = [\dot{\mathbf{x}}_t(t), \dot{\mathbf{x}}_r(t)], \quad (5.2)$$

where $\mathbf{x}_t(t) \in \mathbb{R}^3$ is the translational displacement, $\mathbf{x}_r(t) \in \mathbb{R}^3$ is the rotational displacement in $SO(3)$, represented as a rotation vector, $\dot{\mathbf{x}}_t(t) \in \mathbb{R}^3$ is the translational velocity and $\dot{\mathbf{x}}_r(t) \in \mathbb{R}^3$ is the angular velocity in $SO(3)$. These $\mathbb{R}^6$ vectors are defined similarly for $\mathbf{w}(t)$, $\mathbf{z}(t)$ and $\mathbf{p}(t)$.

Using these vector representations for the three components of the frictional displacement, the friction force equation, (4.1), becomes:

$$\mathbf{f}_f(t) = \Sigma_0 \mathbf{z}(t) + \Sigma_1 \dot{\mathbf{z}}(t) + \Sigma_2 \dot{\mathbf{x}}(t), \quad (5.3)$$

where $\mathbf{f}_f(t) \in \mathbb{R}^6$ is the frictional force-torque value, and the three gain matrices $\Sigma_0, \Sigma_1, \Sigma_2 \in \mathbb{R}^{6 \times 6}$ are

defined as follows:

$$\Sigma_0 = \text{diag}(\sigma_{t,0}, \sigma_{t,0}, \sigma_{t,0}, \sigma_{r,0}, \sigma_{r,0}, \sigma_{r,0}) \quad (5.4)$$

$$\Sigma_1 = \text{diag}(\sigma_{t,1}, \sigma_{t,1}, \sigma_{t,1}, \sigma_{r,1}, \sigma_{r,1}, \sigma_{r,1}) \quad (5.5)$$

$$\Sigma_2 = \text{diag}(\sigma_{t,2}, \sigma_{t,2}, \sigma_{t,2}, \sigma_{r,2}, \sigma_{r,2}, \sigma_{r,2}). \quad (5.6)$$

In these matrices, the scalar coefficients $\sigma_{t,0}$ and $\sigma_{r,0}$ define the friction stiffness in translation and rotation respectively; the scalar coefficients $\sigma_{t,1}$ and $\sigma_{r,1}$ define the tangential compliance damping in translation and rotation respectively; and the scalar coefficients $\sigma_{t,2}$ and $\sigma_{r,2}$ define the viscous friction coefficients in translation and rotation respectively.

5.2.2 Elasto-plastic Friction in $\mathbb{R}^3$

In the elasto-plastic model defined by Hayward *et al.* [Hayward et al., 2009], the value of the elastic pre-sliding state variable, $z(t)$, is updated at each time-step using two saturation operations (equations (12) and (13) in [Hayward et al., 2009]). The elasto-plastic model was originally developed as a way to simulate physical frictional properties, such as Stribeck curves; however, for application to constraints, these are unnecessary. A simple Coulomb friction model, where the Coulomb friction force f_C is the maximum renderable constraint force of the specific device, is sufficient. As proven in appendix A.1, when this simplification is applied to the full elasto-plastic model, at a discrete time-step k , it reduces to a single saturation operation as follows:

$$z_k = \begin{cases} z_c & |z_c| < z_{css} \\ z_{css} & z_c \geq z_{css} \\ -z_{css} & z_c \leq -z_{css} \end{cases}, \quad (5.7)$$

where z_c is the candidate z value and z_{css} is the Coulomb friction, steady state z value:

$$z_c = z_{k-1} + \Delta x_k \quad (5.8)$$

$$z_{css} = f_C / \sigma_0. \quad (5.9)$$

The three preceding equations encapsulate the state updates for the 1-D elasto-plastic friction model with the stated simplifications. Intuitively, these equations show that, at each time step, z_k is updated

by first establishing what its value would be if the displacement was purely elastic (i.e. z_c) and then clamping this value to $\pm z_{css}$.

For translations in $\mathbb{R}^3$, the scalar clamping operation in equation (5.7) can be thought of as clamping a candidate $\mathbf{z}_{t,c}$ value to within a 3-D sphere, centred at the origin, with a radius $z_{t,css}$:

$$\mathbf{z}_{t,k} = \begin{cases} \mathbf{z}_{t,c} & \|\mathbf{z}_{t,c}\| < z_{t,css} \\ z_{t,css} \hat{\mathbf{z}}_{t,c} & otherwise \end{cases}, \quad (5.10)$$

where:

$$\mathbf{z}_{t,c} = \mathbf{z}_{t,k-1} + \Delta \mathbf{x}_{t,k} \quad (5.11)$$

$$\hat{\mathbf{z}}_{t,c} = \mathbf{z}_{t,c} / \|\mathbf{z}_{t,c}\| \quad (5.12)$$

$$z_{t,css} = f_C / \sigma_{t,0}. \quad (5.13)$$

5.2.3 Dynamic Frictional Constraints in $\mathbb{R}^3$

Implementing the model in equation (5.10) would generate isotropic friction forces which oppose motion in all directions of $\mathbb{R}^3$. Anisotropy and frictional redirection can be included by projecting $\mathbf{z}_{t,c}$ into a closed conical set $C_{t,k} \subset \mathbb{R}^3$ rather than a spherical one. The conical subset of $\mathbb{R}^3$ is defined such that its axis is aligned with the constraint penetration vector $\mathbf{p}_{t,k}$, with an apex angle θ , as follows:

$$C_{t,k} = \{\mathbf{w} : \mathbf{w} \in \mathbb{R}^3, \Theta(\mathbf{w}, \mathbf{p}_{t,k}) \leq \theta, \|\mathbf{w}\| \leq z_{t,css}\}, \quad (5.14)$$

where the binary function $\Theta(\cdot, \cdot)$ defines the angle between two vectors.

The computation of $\mathbf{z}_{t,k}$ for a dynamic frictional constraint in $\mathbb{R}^3$ is then given by the projection of $\mathbf{z}_{t,c}$ into $C_{t,k}$ as follows:

$$\mathbf{z}_{t,k} = \arg \min_{\mathbf{w} \in C_{t,k}} \|\mathbf{w} - \mathbf{z}_{t,c}\|. \quad (5.15)$$

The closed-form expression of equation (5.15), for $C_{t,k}$ as defined in equation (5.14), is as follows:

$$\mathbf{z}_{t,k} = \begin{cases} \mathbf{z}_{t,c} & \Theta(\mathbf{z}_{t,c}, \mathbf{p}_{t,k}) \leq \theta \wedge \|\mathbf{z}_{t,c}\| \leq z_{t,css} \\ z_{t,css} \hat{\mathbf{z}}_{t,c} & \Theta(\mathbf{z}_{t,c}, \mathbf{p}_{t,k}) \leq \theta \wedge \|\mathbf{z}_{t,c}\| > z_{t,css} \\ y_{t,c} \hat{\mathbf{y}}_t & otherwise \end{cases}, \quad (5.16)$$

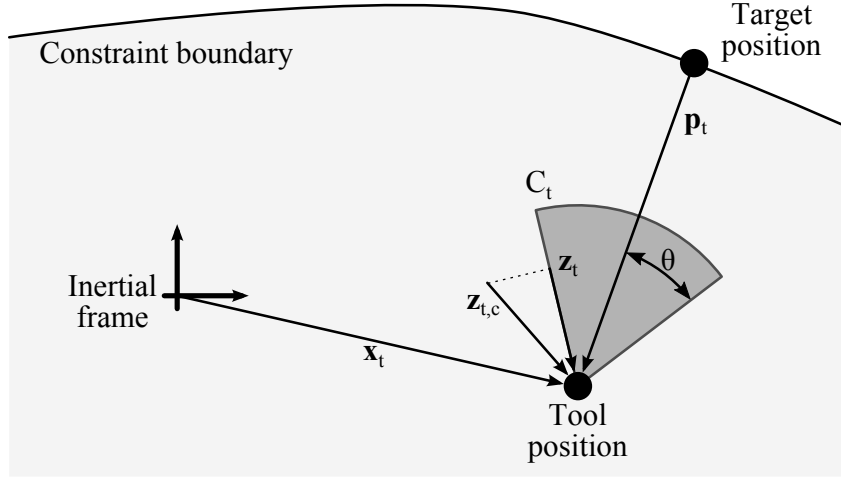


Figure 5.3: Illustration of the proposed translational dynamic frictional constraints. Shown is an example of projecting the candidate $\mathbf{z}_{t,c}$ into the conical vector subset C_t to compute $\mathbf{z}_t$. The axis of the conical set is the constraint penetration vector $\mathbf{p}_t$ and it has an apex angle θ . The system is considered at time step k and the subscripts have been removed for clarity. Figure adapted from [Bowyer and Rodriguez y Baena, 2014]. ©2014 IEEE.

where $\hat{\mathbf{y}}_t$ is a unit vector defining the edge of the cone $C_{t,k}$ on the plane through $\mathbf{z}_{t,c}$ and $\mathbf{p}_{t,k}$; and $y_{t,c}$ is the scalar projection of $\mathbf{z}_{t,c}$ onto $\hat{\mathbf{y}}_t$, clamped between 0 and $z_{t,css}$:

$$y_{t,c} = \begin{cases} 0 & \mathbf{z}_{t,c} \cdot \hat{\mathbf{y}}_t \leq 0 \\ \mathbf{z}_{t,c} \cdot \hat{\mathbf{y}}_t & 0 < \mathbf{z}_{t,c} \cdot \hat{\mathbf{y}}_t < z_{t,css} \\ z_{t,css} & \text{otherwise} \end{cases} \quad (5.17)$$

An illustration of the conical constraint formulation is given in figure 5.3. To understand its behaviour, the three constraint violation states used in chapter 4 can be considered. When the user is orthogonally moving into the constraint (orthogonal violation) the angle between $\mathbf{z}_c$ and $\mathbf{p}$ is small and the projection operation will limit the magnitude of $\mathbf{z}_c$ to z_{css} . Here the model acts in the same way as the standard elasto-plastic friction model in equation (5.10).

When the user is orthogonally retracting from the constraint (orthogonal retraction) the angle between $\mathbf{z}_c$ and $\mathbf{p}$ is almost 180 degrees and the projection operation will set $\mathbf{z}_k$, and subsequently the friction force, to zero.

When the user is moving parallel to the surface of the constraint, but within it (parallel violation), $\mathbf{z}_c$ will be projected onto the edge of the cone. This projection means that some component of the friction force will be pushing the user towards the surface. It is in this case where the constraint exhibits frictional

redirection such that the user is encouraged to retract from the constraint.

If θ is low, then the constraint will not exhibit much frictional redirection and will act largely like a conventional frictional constraint, such as in [Kikuuwe et al., 2008]. However, as θ increases, a larger and larger component of the motion parallel to the constraint will generate frictional redirection forces, reducing the constraint violation.

As this constraint formulation is based on a friction model, it dissipates the majority of the energy that is put in by the user. Formal proof of the dissipativity of this dynamic frictional constraint is shown in appendix A.2. It is known that a dissipative system is also passive [Brogliato et al., 2007], therefore it can be deduced that dynamic frictional constraints are also passive (subject to the inherent limitations of a discrete time system).

5.2.4 Dynamic Frictional Constraints in SO(3)

When applying dynamic frictional constraints to SO(3), the choice of rotation formalism requires consideration. Firstly, it must be possible to project a rotation into a finite, conical subset of the rotation space with a defined apex angle and radius. Secondly, when projecting into the conical subset, it must minimise the difference between the candidate and projected orientations.

Initially, unit quaternions [Dam et al., 1998] were selected for use in applying dynamic frictional constraints in SO(3). Quaternion rotations have the unique property that the angle between any two vector representations (i.e. $\arccos(\mathbf{q}_1 \cdot \mathbf{q}_2)$) is exactly half the minimum angle between the two orientations that they represent. Moreover, the set of unit quaternions can be considered as a unit 3-sphere, $S^3 \subset \mathbb{R}^4$ and therefore the geodesic distance between any two quaternions on S^3 is also exactly half the minimum angle between the two orientations that they represent. Subsequently, the direction of a geodesic between two quaternions on S^3 gives the most direct rotation between them. It is this property that makes quaternions adept at interpolating between rotations using the spherical linear interpolation (SLERP) method [Dam et al., 1998].

The importance of the relationship between geodesics on S^3 and minimum rotation angles is that quaternions can be used to accurately describe a finite conical subset and they guarantee that a projection minimises the true difference in orientation. An illustration of how the conical vector subset would look on a quaternion sphere is shown in figure 5.4. Here, rotations $\mathbf{z}_q$, $\mathbf{z}_{q,c}$, $\mathbf{x}_q$ and $\mathbf{p}_q$ are the quaternion

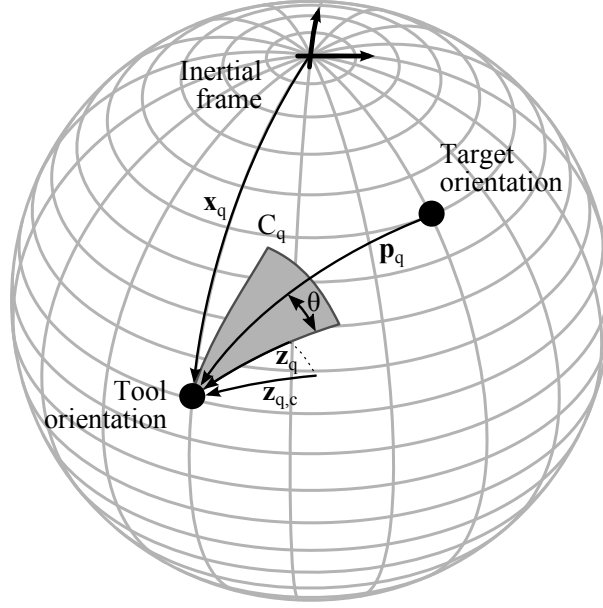


Figure 5.4: Illustration of the conical projection method on the surface of a unit quaternion 3-sphere for rotational dynamic active constraints. The projection operation is the direct parallel of that in figure 5.3, where the candidate orientation $\mathbf{z}_{q,c}$ is projected into the cone to form the orientation $\mathbf{z}_k$. The system is considered at time step k and the subscripts have been removed for clarity. Figure adapted from [Bowyer and Rodriguez y Baena, 2014]. ©2014 IEEE.

equivalents of the rotation vectors $\mathbf{z}_r$, $\mathbf{z}_{r,c}$, $\mathbf{x}_r$ and $\mathbf{p}_r$ respectively.

Ideally, it would be possible to apply the method from equation (5.16) directly to quaternion vectors in $\mathbb{R}^4$. However, these operations would not guarantee that the output remains as a valid unit quaternion in S^3 . Instead, it is necessary to map the surface of the quaternion 3-sphere into $\mathbb{R}^3$ Euclidean space where the validity of the output is guaranteed.

In general, it is not possible to find a mapping from a manifold, such as S^3 , into a Euclidean space where all relative distances and directions are preserved. However, for performing a conical projection, the azimuthal equidistant projection (AEP) [Pearson, 1990] provides a suitable approximation. Fortunately, the result of applying the AEP to a unit quaternion on S^3 is known, and the mapping is in fact equivalent to the conversion of quaternions into rotation vectors [Hartley and Kahl, 2009].

The AEP can be visualised as flattening out S^3 , starting from the projection point, where both distances and azimuth angles of all points with respect to the projection point are preserved. By taking the AEP at the apex of the cone (i.e. $\mathbf{x}_q$), the preservation of distances means that the radius of the cone will be correct, and the preservation of azimuth angles means that the apex angle of the cone will also be correct. The problem with the AEP is that the distance between point pairs which do not include the

projection point are not preserved. This means that minimising the distance between $\mathbf{z}_{r,k}$ and $\mathbf{z}_{r,c}$ does not necessarily minimise the geodesic distance between $\mathbf{z}_{q,k}$ and $\mathbf{z}_{q,c}$ on S^3 . However, as the radius of the cone is small, both $\mathbf{z}_{r,k}$ and $\mathbf{z}_{r,c}$ will be close to the origin, where the projection is almost Euclidean, making this difference negligible (as the results in section 5.5 show).

To apply frictional redirection to rotational motion using the conical projection method, the constraint penetration vector $\mathbf{p}_{r,k}$ and the candidate z vector $\mathbf{z}_{r,c}$ must both be represented as rotation vectors. As combining rotation vectors is inefficient, the candidate $\mathbf{z}_{r,c}$ should first be computed as a quaternion, before converting it into a rotation vector, as follows (taking consideration of the reference frame for each rotation):

$$\mathbf{z}_{q,c} = \Delta \mathbf{x}_{q,k} * \mathbf{z}_{q,k-1}, \quad (5.18)$$

where $*$ is the quaternion product operation and $\Delta \mathbf{x}_{q,k}$ is the change in x orientation, with respect to the inertial frame:

$$\Delta \mathbf{x}_{q,k} = \mathbf{x}_{q,k} * \mathbf{x}_{q,k-1}^{-1}. \quad (5.19)$$

Frictional redirection for rotation can be applied (similarly to equation (5.15)) by projecting the candidate rotation vector into the conical subset, as shown below:

$$\mathbf{z}_{r,k} = \arg \min_{\mathbf{w} \in C_{r,k}} \|\mathbf{w} - \mathbf{z}_{r,c}\|, \quad (5.20)$$

where the conical subset is given as:

$$C_{r,k} = \{ \mathbf{w} : \mathbf{w} \in \mathbb{R}^3, \Theta(\mathbf{w}, \mathbf{p}_{r,k}) \leq \theta, \|\mathbf{w}\| \leq z_{r,css} \}, \quad (5.21)$$

and where the cone's radius is computed from the Coulomb friction torque, τ_C and rotational stiffness as follows:

$$z_{r,css} = \tau_C / \sigma_{r,0}. \quad (5.22)$$

The implementation of this projection is the direct parallel of that for translation in equation (5.16):

$$\mathbf{z}_{r,k} = \begin{cases} \mathbf{z}_{r,c} & \Theta(\mathbf{z}_{r,c}, \mathbf{p}_{r,k}) \leq \theta \wedge \|\mathbf{z}_{r,c}\| \leq z_{r,css} \\ z_{r,css} \hat{\mathbf{z}}_{r,c} & \Theta(\mathbf{z}_{r,c}, \mathbf{p}_{r,k}) \leq \theta \wedge \|\mathbf{z}_{r,c}\| > z_{r,css} \\ y_{r,c} \hat{\mathbf{y}}_r & \text{otherwise} \end{cases}, \quad (5.23)$$

where $\hat{\mathbf{y}}_r$ is a unit vector defining the edge of the cone $C_{r,k}$ on the plane through $\mathbf{z}_{r,c}$ and $\mathbf{p}_{r,k}$; and $y_{r,c}$ is the scalar projection of $\mathbf{z}_{r,c}$ onto $\hat{\mathbf{y}}_r$, clamped between 0 and $z_{r,css}$:

$$y_{r,c} = \begin{cases} 0 & \mathbf{z}_{r,c} \cdot \hat{\mathbf{y}}_r \leq 0 \\ \mathbf{z}_{r,c} \cdot \hat{\mathbf{y}}_r & 0 < \mathbf{z}_{r,c} \cdot \hat{\mathbf{y}}_r < z_{r,css} \\ z_{r,css} & otherwise \end{cases} \quad (5.24)$$

As with the dynamic frictional constraint in $\mathbb{R}^3$, the projection operation in rotation is both dissipative and passive, as proven in appendix A.3.

5.3 Coupled Frictional Redirection between Translation and Rotation

The dynamic frictional constraint formulation for SE(3), defined above, considers the translational and rotational components of a tool's pose independently. Both components exhibit the, penetration minimising, frictional redirection between their respective degrees of freedom, but frictional redirection does not operate from one to the other. This dynamic frictional constraint will henceforth be referred to as the 'uncoupled' dynamic frictional constraint.

It has previously been stated in the literature [Castillo-Cruces and Wahrburg, 2010], and also found during preliminary experiments of the uncoupled dynamic frictional constraints in section 5.2.3, that, in tasks which require control of both position and orientation, the user often focuses on one of them to the detriment of the other. This means that, for example, if the user is concentrating closely on tracking the position of a target, they will pay less attention to its orientation, potentially jeopardising accuracy. In a dynamic task, such as can be assisted by a dynamic active constraint, the target position and orientation can change continuously and the user needs to actively track both simultaneously. It is not sufficient for the user to alternate between focusing on translation and rotation as the target for the unconsidered one of the two will potentially be changing, even when the tool is not moving.

With a friction based constraint, such as the uncoupled dynamic frictional constraint, the above issue of continuously tracking both position and orientation is particularly problematic. The reason for this is that friction can only generate forces when the user is moving. If the user is concentrating on one of either translation or rotation, then they will likely be almost static in the other. If they are almost static, then there will not be any significant constraint forces and they will not be corrected or guided. This

is fundamentally the same problem which was addressed by frictional redirection, as proposed for $\mathbb{R}^3$, in chapter 4. In that case, the user was within the constraint but not moving further into the constraint, instead moving parallel to the constraint's surface. When this happened, the frictional redirection took energy from the direction in which the user was moving and reapplied it to the user in the direction in which correction was required.

It is therefore proposed that frictional redirection can be applied to couple motion between translation and rotation, such that, if the user is moving in any one of the six primary degrees of freedom, they will input energy into the system which can correct their motion in any of the other five degrees of freedom. Indeed, the concept behind coupled frictional redirection, as illustrated in figure 5.5, is that energy which the user applies to the system in either translation or rotation can be reapplied to the user to correct errors in the other.

5.3.1 Combining Translation and Rotation

The main challenge with applying coupled frictional redirection between translation and rotation is that the two subspaces have different units and stiffnesses. This means that it is not possible to simply combine the two $\mathbb{R}^3$ spaces into a single $\mathbb{R}^6$ space and apply the conical projection upon $\mathbb{R}^6$ vectors. To explain this, notation to describe the coupled dynamic frictional constraint concept is first introduced.

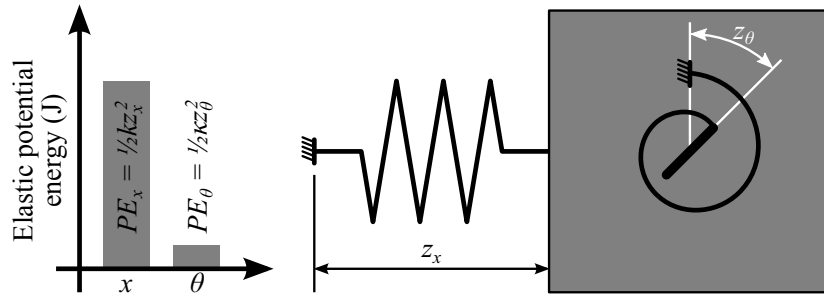
Within the discrete implementation of the coupled frictional redirection method at time k , the elastic displacement $\mathbf{z}_k \in \mathbb{R}^6$ and the candidate elastic displacement $\mathbf{z}_c \in \mathbb{R}^6$ are defined as follows:

$$\mathbf{z}_k = [\mathbf{z}_{t,k}, \mathbf{z}_{r,k}] \quad (5.25)$$

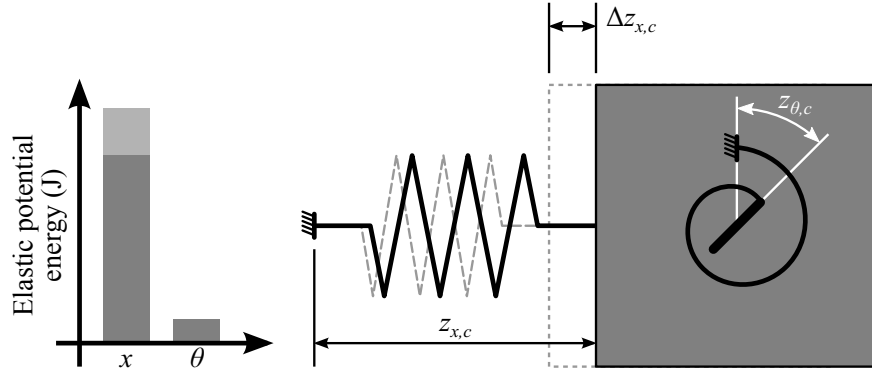
$$\mathbf{z}_c = [\mathbf{z}_{t,c}, \mathbf{z}_{r,c}], \quad (5.26)$$

where $\mathbf{z}_{t,k}$, $\mathbf{z}_{r,k}$, $\mathbf{z}_{t,c}$ and $\mathbf{z}_{r,c}$ are as defined previously.

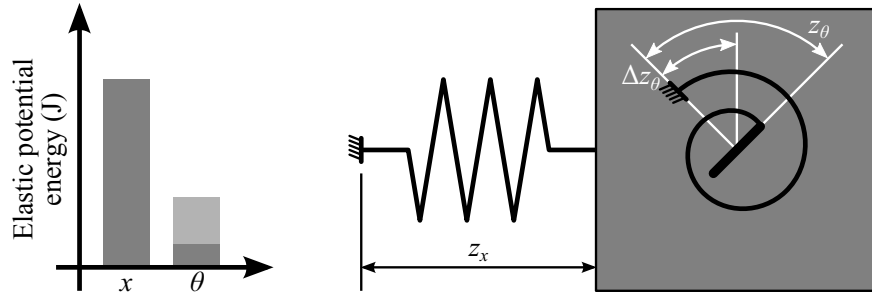
Frictional redirection is implemented by projecting $\mathbf{z}_c$ onto a vector $\hat{\mathbf{y}} \in \mathbb{R}^6$ to compute $\mathbf{z}_k$, such that energy applied to the system in one direction can generate forces upon the user in another direction. For this formulation to remain passive, it is imperative that the projection operation does not increase the potential energy stored in the elastic pre-sliding region. That is to say, the potential energy stored in the elastic displacement $\text{PE}(\mathbf{z}_k)$ - as the sum of translation $\text{PE}(\mathbf{z}_{t,k})$ and rotation $\text{PE}(\mathbf{z}_{r,k})$ - must be no more than was stored in the candidate elastic displacement $\text{PE}(\mathbf{z}_c)$ - as the sum of translation $\text{PE}(\mathbf{z}_{t,c})$



(a) $t = t_0$: The initial configuration of the system. There is a large elastic displacement in translation (z_x) and a small elastic displacement in rotation (z_θ). This means that the user will experience a large translational friction force and a small rotational friction force.



(b) $t = t_1$: The candidate configuration after a time-step in which the user has changed position ($\Delta z_{x,c}$) but not the orientation. Without frictional redirection, the translational friction force would increase, but the rotational friction torque would remain the same, since the system would not be provided with additional “rotational energy” to help direct the user’s motion in the direction of perfect alignment.



(c) $t = t_1$: The actual configuration after coupled frictional redirection has been applied to the candidate configuration. Rather than extending the translational spring, the equivalent potential energy has been transferred into the torsional spring (Δz_θ), increasing the torque applied to correct the user’s orientation. As the total potential energy in the system is the same before and after frictional redirection, the system remains passive.

Figure 5.5: Illustrations of the concept of coupled frictional redirection between translation (x) and rotation (θ) in a two degree of freedom system. The user interacts with this system at the handle in the centre of the torsional spring. The translational spring represents the elastic pre-sliding friction force in translation and the torsional spring represents the elastic pre-sliding friction force in rotation. z_x and z_θ are the pre-sliding displacements in translation and rotation respectively. $z_{x,c}$ and $z_{\theta,c}$ are the pure elastic displacement candidate pre-sliding displacements in translation and rotation respectively. The graphs on the left show the elastic potential energies, PE_x and PE_θ , stored in the translational and torsional springs respectively.

and rotation $\text{PE}(\mathbf{z}_{r,c})$:

$$\text{PE}(\mathbf{z}_c) \geq \text{PE}(\mathbf{z}_k) \quad (5.27)$$

$$\text{PE}(\mathbf{z}_{t,c}) + \text{PE}(\mathbf{z}_{r,c}) \geq \text{PE}(\mathbf{z}_{t,k}) + \text{PE}(\mathbf{z}_{r,k}) \quad (5.28)$$

$$\frac{1}{2}\sigma_{t,0}\|\mathbf{z}_{t,c}\|^2 + \frac{1}{2}\sigma_{r,0}\|\mathbf{z}_{r,c}\|^2 \geq \frac{1}{2}\sigma_{t,0}\|\mathbf{z}_{t,k}\|^2 + \frac{1}{2}\sigma_{r,0}\|\mathbf{z}_{r,k}\|^2. \quad (5.29)$$

This criteria for passivity can be represented from the $\mathbb{R}^6$ vectors, as follows:

$$\frac{1}{2}\mathbf{z}_c \cdot \Sigma_0 \mathbf{z}_c \geq \frac{1}{2}\mathbf{z}_k \cdot \Sigma_0 \mathbf{z}_k \quad (5.30)$$

$$\mathbf{z}_c \cdot \Sigma_0 \mathbf{z}_c \geq \mathbf{z}_k \cdot \Sigma_0 \mathbf{z}_k. \quad (5.31)$$

When the stiffness matrix Σ_0 is anisotropic (i.e. $\sigma_{t,0} \neq \sigma_{r,0}$) the projection of $\mathbf{z}_c$ onto $\hat{\mathbf{y}}$ in $\mathbb{R}^6$ is not guaranteed to satisfy the passivity criteria in equation (5.31) and therefore simply projecting in the combined $\mathbb{R}^6$ space is unsuitable.

5.3.2 Elastic Potential Energies in $\mathbb{R}^6$

The solution to the problem of unit parity between translation and rotation is to represent the elastic pre-sliding state variable $\mathbf{z}_k$ and the candidate state variable $\mathbf{z}_c$ in terms of the elastic potential energies stored within their representative springs. By projecting the potential energy, rather than the spring displacements, it can be guaranteed that the potential energy stored will not increase during the projection, and the system will subsequently remain passive.

To apply projections to potential energies, rather than spring displacements, it is necessary to map the vectors into the ‘potential energy space’. The potential energy mapping $\mathbf{M} : \mathbb{R}^6 \rightarrow \mathbb{R}^6$ is defined as:

$$\mathbf{M}(\mathbf{a}) = \tilde{\mathbf{a}}, \quad (5.32)$$

such that:

$$\tilde{\mathbf{a}}_i = \text{sgn}(\mathbf{a}_i) \sqrt{\text{PE}(\mathbf{a}_i)} = \text{sgn}(\mathbf{a}_i) \sqrt{\frac{1}{2} \Sigma_{0,i} \mathbf{a}_i^2}, \quad (5.33)$$

where $i = [1, 6]$, $\mathbf{a}_i$ is the i -th element of the input vector $\mathbf{a} \in \mathbb{R}^6$, $\tilde{\mathbf{a}}_i$ is the i -th element of the output vector $\tilde{\mathbf{a}} \in \mathbb{R}^6$ and $\Sigma_{0,i}$ is the i -th element along the diagonal of the stiffness matrix Σ_0 . The inverse of

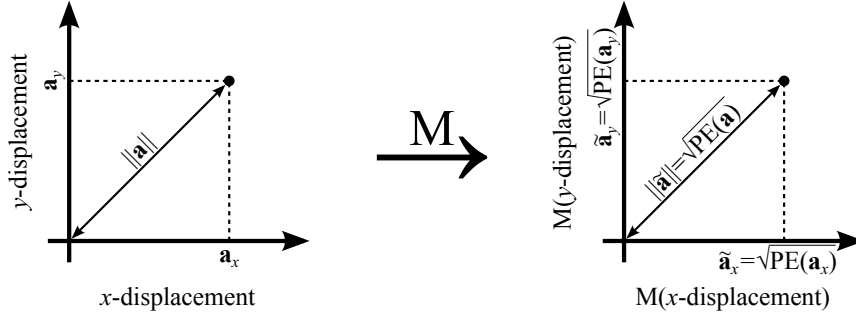


Figure 5.6: Illustration of the potential energy mapping, M , which converts a spring displacement vector $\mathbf{a}$ into the potential energy space $\tilde{\mathbf{a}}$, in which the magnitude of a vector is equal to the square root of the total elastic potential energy stored within elastic spring.

the potential energy mapping $M^{-1} : \mathbb{R}^6 \rightarrow \mathbb{R}^6$ is similarly defined as:

$$M^{-1}(\tilde{\mathbf{a}}) = \mathbf{a}, \quad (5.34)$$

such that:

$$\mathbf{a}_i = \text{sgn}(\tilde{\mathbf{a}}_i) \sqrt{\frac{2\tilde{\mathbf{a}}_i^2}{\sum_{0,i}}}. \quad (5.35)$$

Once the potential energy mapping has been applied to an elastic displacement vector, the i -th element within the output vector represents the square root of the elastic potential energy stored in the i -th dimension of $\mathbb{R}^6$. The reason that the value is the square root of the potential energy, rather than the potential energy itself, is that the total potential energy stored within a multi-dimensional spring system is the sum of the energy in each dimension and not the vector norm. Taking the square root ensures that the norm of the mapped vector is equal to the square root of the total potential energy within the system, illustrated in figure 5.6, as follows:

$$\|M(\mathbf{a})\| = \|\tilde{\mathbf{a}}\| = \sqrt{\text{PE}(\mathbf{a})}. \quad (5.36)$$

Given that $\tilde{\mathbf{z}}_k = M(\mathbf{z}_k)$ and $\tilde{\mathbf{z}}_c = M(\mathbf{z}_c)$, equation (5.36) ensures that:

$$\|\tilde{\mathbf{z}}_c\| \geq \|\tilde{\mathbf{z}}_k\| \iff \sqrt{\text{PE}(\mathbf{z}_c)} \geq \sqrt{\text{PE}(\mathbf{z}_k)} \iff \text{PE}(\mathbf{z}_c) \geq \text{PE}(\mathbf{z}_k). \quad (5.37)$$

This means that, as long as the frictional redirection projection of $\tilde{\mathbf{z}}_c$ to form $\tilde{\mathbf{z}}_k$ does not increase its magnitude, the system's energy will not increase and thus it will remain passive.

5.3.3 Conical Projections of Potential Energies

Applying coupled frictional redirection between translation and rotation in $\mathbb{R}^6$ is almost identical to the methods for translation and rotation individually, with the exception that the projection is performed in the potential energy space.

The cone into which the elastic displacement vectors are projected for coupled frictional redirection is a closed conical set $\tilde{C}_k \subset \mathbb{R}^6$, defined as follows:

$$\tilde{C}_k = \{ \mathbf{w} : \mathbf{w} \in \mathbb{R}^6, \Theta(\mathbf{w}, \tilde{\mathbf{p}}_k) \leq \theta, \|\mathbf{w}\| \leq \tilde{z}_{css}(\mathbf{w}) \}, \quad (5.38)$$

where $\tilde{\mathbf{p}}_k = \mathbf{M}(\mathbf{p}_k)$ is the mapped penetration vector, as explained above, and $\tilde{z}_{css}(\cdot)$ is a function which computes the radius of the cone in the direction of the supplied vector, defined as follows:

$$\tilde{z}_{css}(\mathbf{w}) = \left\| \mathbf{w} \min \left(\frac{\tilde{z}_{t,css}}{\|H\mathbf{w}\|}, \frac{\tilde{z}_{r,css}}{\|T\mathbf{w}\|} \right) \right\|, \quad (5.39)$$

and $H \in \mathbb{R}^{6 \times 6}$ and $T \in \mathbb{R}^{6 \times 6}$ are diagonal matrices selecting the three head and tail elements respectively and $\tilde{z}_{t,css}$ and $\tilde{z}_{r,css}$ are the square roots of the steady state potential energies, due to Coulomb friction, in translation and rotation respectively, as follows:

$$H = \text{diag}(1, 1, 1, 0, 0, 0) \quad (5.40)$$

$$T = \text{diag}(0, 0, 0, 1, 1, 1) \quad (5.41)$$

$$\tilde{z}_{t,css} = \sqrt{\frac{1}{2} \boldsymbol{\sigma}_{t,0}^2 z_{t,css}^2} \quad (5.42)$$

$$\tilde{z}_{r,css} = \sqrt{\frac{1}{2} \boldsymbol{\sigma}_{r,0}^2 z_{r,css}^2}. \quad (5.43)$$

A function is used to compute $\tilde{z}_{css}$, rather than defining it statically, because both translation and rotation will very likely have different Coulomb steady state potential energies (as they have different forces due to Coulomb friction and stiffnesses) meaning that a single value would have to be conservatively set to the minimum of the two, thus underutilising the maximum force of the other. The definition of $\tilde{z}_{css}$ as a function of $\mathbf{w}$ in equation (5.39) allows both translation and rotation forces to be maximised independently, providing the strongest guidance to the user.

When defining the axis of the frictional redirection cone $\mathbf{p}_k \in \mathbb{R}^6$ from the constraint penetration vectors

$\mathbf{p}_{t,k}$ and $\mathbf{p}_{r,k}$, it was decided to weight their relative contributions to allow constraining either translation or rotation constraints to be prioritised. The axis of the frictional redirection cone was therefore composed as:

$$\mathbf{p}_k = [\psi \mathbf{p}_{t,k}, (1 - \psi) \mathbf{p}_{r,k}], \quad (5.44)$$

where $\psi \in [0, 1]$ is the weighting factor which defines whether translation or rotation takes constraint dominance, such that when $0.5 < \psi \leq 1$ translation is given precedence, when $0 \leq \psi < 0.5$ rotation is given precedence and when $\psi = 0.5$ they are given equal precedence.

The, potential energy mapped, elastic displacement $\tilde{\mathbf{z}}_k$ for a coupled dynamic frictional constraint can therefore be computed, similarly to the unmapped case, from the projection of $\tilde{\mathbf{z}}_c$ into $\tilde{C}_k$ as follows:

$$\tilde{\mathbf{z}}_k = \arg \min_{\mathbf{w} \in \tilde{C}_k} \|\mathbf{w} - \tilde{\mathbf{z}}_c\|, \quad (5.45)$$

which can be approximated with the following the closed-form expression:

$$\tilde{\mathbf{z}}_k = \begin{cases} \tilde{\mathbf{z}}_c & \Theta(\tilde{\mathbf{z}}_c, \tilde{\mathbf{p}}_k) \leq \theta \wedge \|\tilde{\mathbf{z}}_c\| \leq \tilde{z}_{css}(\tilde{\mathbf{z}}_c) \\ \tilde{z}_{css}(\tilde{\mathbf{z}}_c) \hat{\tilde{\mathbf{z}}}_c & \Theta(\tilde{\mathbf{z}}_c, \tilde{\mathbf{p}}_k) \leq \theta \wedge \|\tilde{\mathbf{z}}_c\| > \tilde{z}_{css}(\tilde{\mathbf{z}}_c) \\ \tilde{y}_c \hat{\tilde{\mathbf{y}}} & otherwise \end{cases}, \quad (5.46)$$

where $\hat{\tilde{\mathbf{y}}}$ is a unit vector defining the edge of the cone $\tilde{C}_k$ on the hyperplane through $\tilde{\mathbf{z}}_c$ and $\tilde{\mathbf{p}}_k$; and $\tilde{y}_c$ is the scalar projection of $\tilde{\mathbf{z}}_c$ onto $\hat{\tilde{\mathbf{y}}}$, clamped between 0 and $\tilde{z}_{css}(\hat{\tilde{\mathbf{y}}})$, as follows:

$$\tilde{y}_c = \begin{cases} 0 & \tilde{\mathbf{z}}_c \cdot \hat{\tilde{\mathbf{y}}} \leq 0 \\ \tilde{\mathbf{z}}_c \cdot \hat{\tilde{\mathbf{y}}} & 0 < \tilde{\mathbf{z}}_c \cdot \hat{\tilde{\mathbf{y}}} < \tilde{z}_{css}(\hat{\tilde{\mathbf{y}}}) \\ \tilde{z}_{css}(\hat{\tilde{\mathbf{y}}}) & otherwise \end{cases}. \quad (5.47)$$

5.4 Experiments

5.4.1 Experimental Apparatus

The effectiveness of the proposed dynamic frictional constraint formulations were quantified with a simulated version of *The Steady-hand Game*. In this game, a user moves a ring along a curving wire pathway, while attempting to avoid the ring and wire making contact. This game is a test of both manual

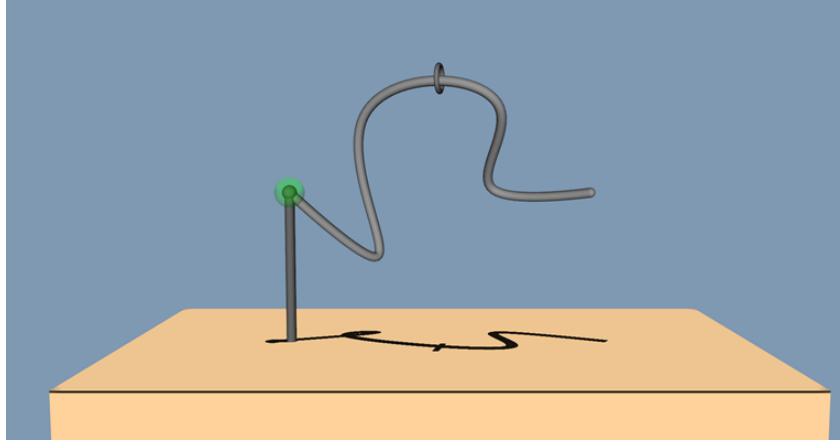


Figure 5.7: Simulation of the dynamic steady-hand game used within the experiments. Shown is the wire pathway which randomly deformed over time, the ring (in the middle of the wire) which was controlled by the human user with a sigma.6+ and the active target (green sphere) towards which the user is moving.

dexterity and hand-eye coordination for the user. The primary reason for selecting this game to validate dynamic frictional constraints in SE(3) is that the user must simultaneously control both the position and orientation of the ring.

The visualisation for the simulation was created using The Visualization Toolkit (VTK, Kitware Inc.) and is shown in figure 5.7. To assist the user in determining the depth of the ring and wire within the scene, it was displayed on a 3-D monitor using NVIDIA 3D Vision (NVIDIA Corp.). A sigma.6+ haptic device (Force Dimension Inc.) allowed the user to control the position of the ring within the simulation and rendered both the constraint forces and torques. The sigma.6+ used in the experiments was right-handed and therefore all test subject, irrespective of their dominant hand, used their right hand to undertake the experiment.

To validate the dynamic nature of the dynamic frictional constraints, the wire pathways within the simulated dynamic steady-hand game randomly deformed over time. The deformation was achieved by applying a bounded random walk to each control point of the cardinal spline that defined the wire pathway. For these experiments, the speed of each deforming control point was set at approximately 12 mm.s^{-1} . The inner diameter of the ring was set to 14 mm and the diameter of the wire pathway was 8 mm. All of these dimensions were initially set arbitrarily and then tuned empirically to create a task of suitable difficulty to challenge all user's abilities while remaining achievable.

The constraint control algorithms were implemented in C++ on a Linux PC with a 3 GHz Core 2 Duo CPU (Intel Corp.) at a haptic rate of 1,000 Hz. The 3-D visualisation was implemented in C++ on a

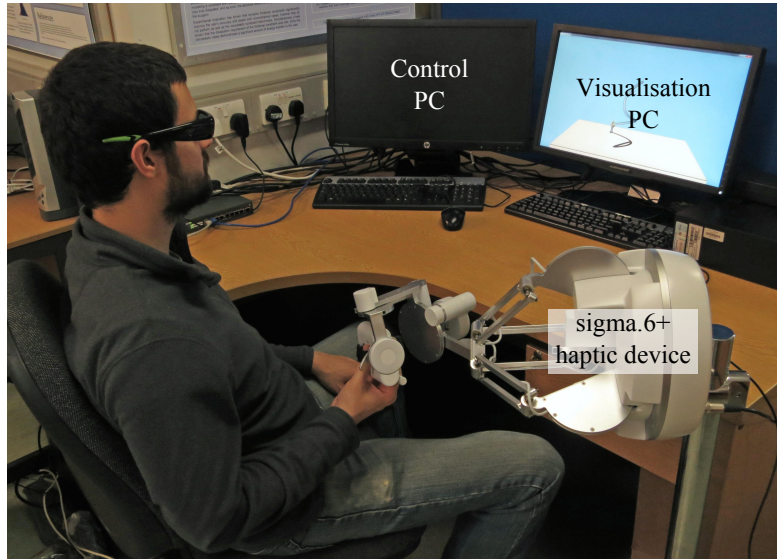


Figure 5.8: A photo of the experimental apparatus used in the simulation of the dynamic steady-hand game to evaluate dynamic frictional constraints in SE(3). Annotated are the three primary components: the 3-D visualisation; the 6-D haptic device and the control PC.

Windows PC (Microsoft Corp.) with a 2.93 GHz Core i7-870 CPU (Intel Corp.)/Quadro 4000 GPU (NVIDIA Corp.) at a display rate of approximately 40 Hz. The user datagram protocol (UDP) was used to relay ring positions from the control PC to the visualisation PC. The complete visualisation, haptic and control system used for the experiment is shown in figure 5.8.

5.4.2 Experimental Procedure

Six constraint methods were used to assist the test subjects within the experiments. These are shown below.

- **UC** - Unconstrained, no constraint forces or torques were applied.
- **DFC3T** - 3-D translational dynamic frictional constraints, as described in section 5.2.3, with no constraint torques and $\theta = 30$ degrees.
- **DFC3R** - 3-D rotational dynamic frictional constraints, as described in section 5.2.4, with no constraint forces and $\theta = 30$ degrees.
- **FC6** - 6-D frictional constraints without frictional redirection. Equivalent to DFC3T and DFC3R being applied simultaneously with $\theta = 0$ degrees. This method approximates a conventional static frictional constraint, similar to that described in [Kikuuwe et al., 2008], and was included in the experiment to quantify the potential benefits of frictional redirection.

Parameter	Value	Parameter	Value
f_C	15.0 N	τ_C	400 N.mm
$\sigma_{t,0}$	2.0 N.mm ⁻¹	$\sigma_{r,0}$	1600 N.mm.rad ⁻¹
$\sigma_{t,1}$	0 N.s.mm ⁻¹	$\sigma_{r,1}$	0 N.mm.s.rad ⁻¹
$\sigma_{t,2}$	0.007 N.s.mm ⁻¹	$\sigma_{r,2}$	60 N.mm.s.rad ⁻¹

Table 5.1: Parameters for translation and rotation used in the frictional constraints.

- **DFC6** - 6-D dynamic frictional constraints with frictional redirection in translation and rotation. Equivalent to DFC3T and DFC3R being applied simultaneously with $\theta = 30$ degrees.
- **DFC6C** - 6-D coupled dynamic frictional constraints with coupled frictional redirection in translation and rotation, as described in section 5.3, with $\theta = 30$ degrees and $\psi = 0.9$. This ψ value was selected in order to prioritise constraints in position over orientation as translational motion was the primary mode by which the user was likely to penetrate the constraint.

The constraints were applied based on an optimal target pose for the ring which maximised its separation from the wire. The target position was defined at the closest point on the centreline of the pathway to the ring's current position. The target orientation was defined as a coordinate frame at the target position where the axis of the ring was aligned with the tangent of the pathway. The closest points were found by exhaustively searching a discretised set of line segments which represented the centreline.

The parameters for the constraints were defined empirically based on the capabilities of the selected haptic device and are shown in table 5.1.

Each subject was given a document explaining the experimental procedure and then allowed a period of familiarisation with the haptic device, simulation and task. After this training period, 30 test runs were undertaken by each subject. Five different wire pathways were used in the tests, each of which was run with all six of the constraint methods above. This gave a total of five runs, per subject, per constraint method. The ordering of the test runs was randomised and the users were not informed of the nature of the constraints, or which one was assisting them. The sole requirement placed upon the subject was that they avoid contact between the ring and the wire while moving from one end of the wire to the other.

A total of 20 people took part in the experiments. This consisted of 17 (3) males (females), 18 (2) right (left) handed people and 7 (13) people who had (had no) previous haptic experience, with an age range of 22 to 38.

5.5 Results

The primary metrics used to assess the performance of the each constraint method were (as explained below): the contact percentage, the error from the optimal pose and the haptic velocity. The haptic power was also recorded as a secondary metric to validate the energetic properties of the different constraint methods. Summaries of the results for each method, with each of the metrics, are presented in tables 5.2 and 5.3.

The contact percentage represents the proportion of the pathway for which the ring was touching, or completely pushed through, that pathway. The distributions for the contact percentage are shown in figure 5.9.

The translational RMSE represents the root mean square of the positional difference between the centre of the ring and the optimal pose at the centre of the pathway. The distributions for the translational RMSE are shown in figure 5.10.

The rotational RMSE represents the root mean square of the angular difference between the orientation of the ring and the optimal pose at the centre of the pathway with its centreline parallel to the tangent of the pathway. The distributions for the rotational RMSE are shown in figure 5.11.

The mean translational velocity represents the mean of the translational velocity of the ring as it moved along the pathway. The distributions for the mean translational velocity are shown in figure 5.12.

The mean rotational velocity represents the mean of the rotational velocity of the ring as it moved along the pathway. The distributions for the mean rotational velocity are shown in figure 5.13.

The mean translational power represents the mean of the power transferred from the haptic device to the user due to translational forces. The distributions for the mean translational power are shown in figure 5.14.

The mean rotational power represents the mean of the power transferred from the haptic device to the user due to rotational torques. The distributions for the mean rotational power are shown in figure 5.15.

The mean total power represents the mean of the power transferred from the haptic device to the user due to both translational forces and rotational torques. The distributions for the mean total power are shown in figure 5.16.

One-way analysis of variance (ANOVA) tests were applied to the distributions for each of the primary

Constraint method	Contact percentage (%)	Translational RMSE (mm)	Rotational RMSE (degrees)	Mean translational velocity (mm.s ⁻¹)	Mean rotational velocity (degrees.s ⁻¹)
UC	91.1 ± 6.8	10.5 ± 3.8	31.4 ± 8.9	16.7 ± 7.6	10.8 ± 5.4
DFC3T	34.7 ± 19.6	2.4 ± 0.7	30.1 ± 9.8	27.9 ± 16.4	22.6 ± 15.0
DFC3R	91.2 ± 6.5	10.8 ± 3.9	29.9 ± 9.6	16.4 ± 6.7	9.9 ± 4.8
FC6	69.9 ± 19.7	4.7 ± 1.6	30.0 ± 10.1	25.5 ± 16.1	17.7 ± 10.9
DFC6	32.5 ± 17.6	2.4 ± 0.7	27.9 ± 8.8	27.1 ± 15.8	19.7 ± 11.2
DFC6C	7.5 ± 9.0	1.4 ± 0.5	16.2 ± 5.9	22.9 ± 12.3	20.9 ± 10.3

Table 5.2: Summary of the results for the six constraint methods across the primary metrics. The abbreviations for the test conditions are as defined in section 5.4.2. Each value shows the mean and standard deviation from a total of 100 tests across 20 test subjects.

Constraint method	Mean translational power (mW)	Mean rotational power (mW)	Mean total power (mW)
UC	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
DFC3T	-17.5 ± 27.8	0.0 ± 0.0	-17.5 ± 27.8
DFC3R	0.0 ± 0.0	-5.0 ± 6.0	-5.0 ± 6.0
FC6	-11.2 ± 18.8	-13.4 ± 17.2	-24.6 ± 35.2
DFC6	-16.3 ± 21.5	-16.9 ± 19.9	-33.2 ± 40.0
DFC6C	-35.4 ± 46.4	-10.3 ± 15.8	-45.7 ± 54.8

Table 5.3: Summary of the results for the six constraint methods across the secondary metrics. The abbreviations for the test conditions are as defined in section 5.4.2. Each value shows the mean and standard deviation from a total of 100 tests across 20 test subjects.

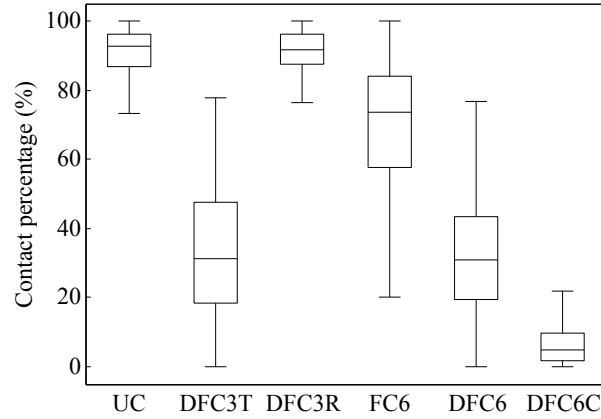


Figure 5.9: Distributions of the contact percentage across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

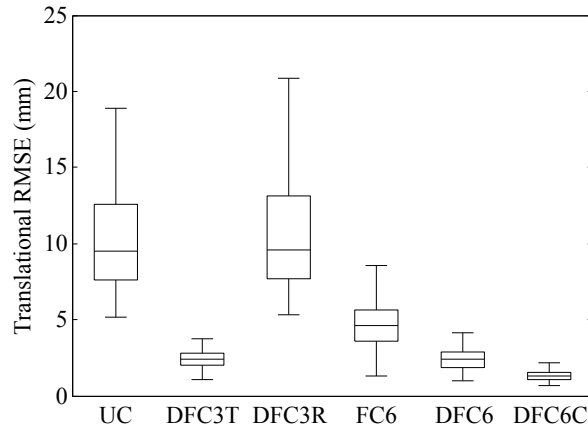


Figure 5.10: Distributions of the translational RMSE across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

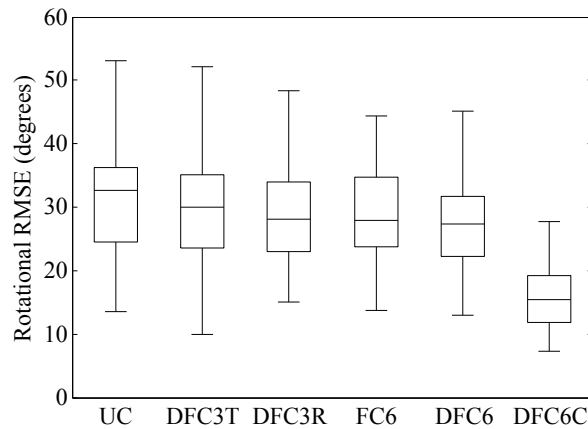


Figure 5.11: Distributions of the rotational RMSE across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

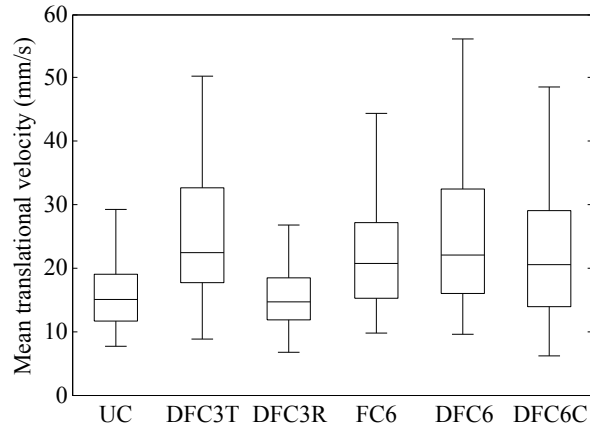


Figure 5.12: Distributions of the mean translational velocity across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

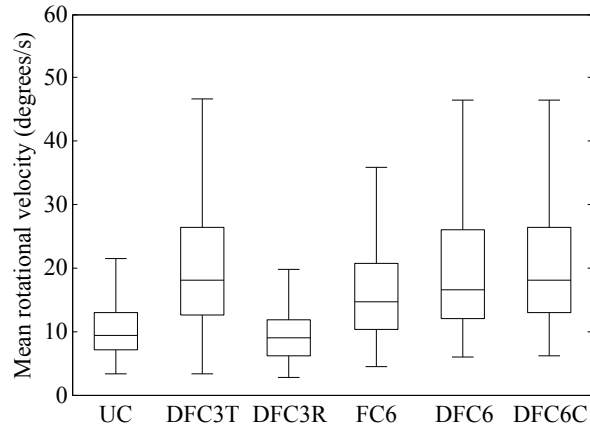


Figure 5.13: Distributions of the mean rotational velocity across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

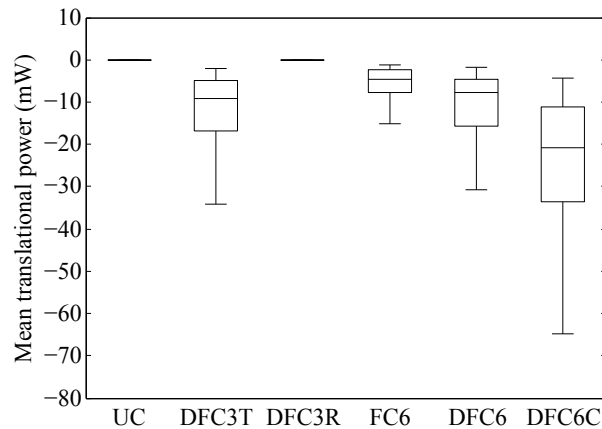


Figure 5.14: Distributions of the mean translational power across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

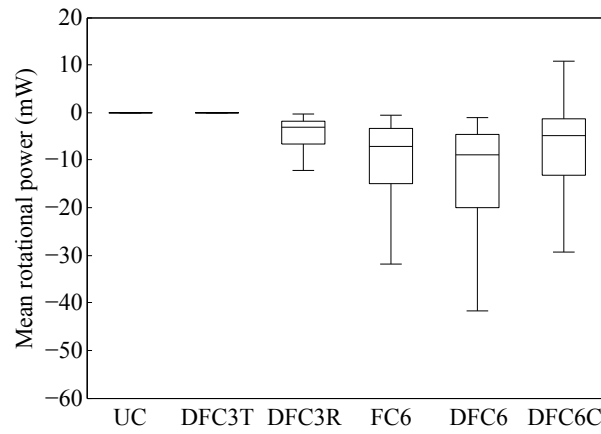


Figure 5.15: Distributions of the mean rotational power across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

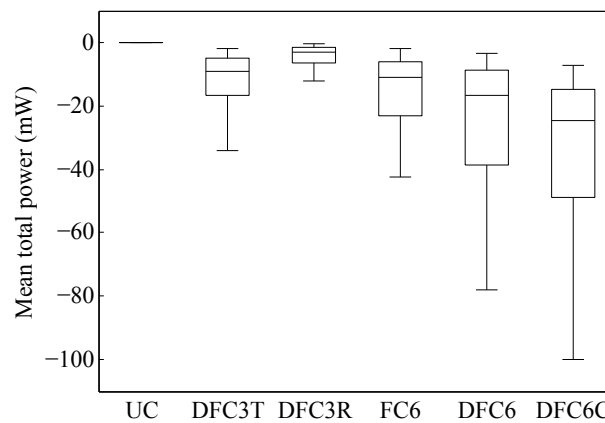


Figure 5.16: Distributions of the mean total power across the six constraint methods. The abbreviations for the test conditions are as defined in section 5.4.2. Each box shows the distribution from 100 tests across 20 test subjects. The outliers are not shown for clarity.

metrics and they were all found to be significantly affected by the constraint method used ($p < 10^{-6}$). Pairwise comparisons were subsequently made between each constraint method pairing using Tukey's honest significant difference (HSD) test [Steel et al., 1997] for each metric at the 95 % confidence interval. The full results from these pairwise comparisons are given in table 5.4.

5.6 Discussion

The results shown in the previous section for the experimental validation of dynamic frictional constraints in SE(3) illustrate that they can be of significant benefit to a user in a dynamic task requiring control of both position and orientation.

The principal metric for the performance of the user within the task was the contact percentage. The only requirement placed upon the user was to keep the ring and pathway from making contact, and therefore this metric quantifies how able they were to do this with the different constraint formulations. The results show that, in the unconstrained case (UC), the contact percentage was very high (91.1 %), identifying poor performance. The randomised deformation of the pathway meant that the task itself was difficult as it required quick reactions, high dexterity and good spatial perception from the user. This result shows that, without assistance, the majority of the test subjects simply did not have the physical ability to successfully navigate along the pathway while it deformed.

During the experiments where translational dynamic frictional constraints were applied (DFC3T), the mean contact percentage (34.7 %) improved significantly from UC. These results confirm those presented in chapter 4 and, importantly, also show that constraining position is beneficial even when orientation is a factor in the task. As tracking the Cartesian position of the deforming pathway is a significant component of the task's difficulty, this result is to be expected; however, it presents the possibility that the method reduces the cognitive load of positioning the ring sufficiently that the user can consider and control the orientation in parallel.

When dynamic frictional constraints were applied to rotation in isolation (DFC3R), the mean contact percentage (91.2 %) was almost identical to UC; and the small difference was not found to be statistically significant. During the tests of UC and DFC3R, the users spent most of the time with the ring deeply embedded in, or completely pushed through, the pathway. The magnitude of the assistance that a rotational constraint can offer is much smaller than that for translation as, at most, it can only separate

	DFC3T	DFC3R	FC6	DFC6	DFC6C
UC	**		**	**	**
DFC3T		**	**		**
DFC3R			**	**	**
FC6				**	**
DFC6					**

(a) Contact percentage

	DFC3T	DFC3R	FC6	DFC6	DFC6C
UC	**		**	**	**
DFC3T		**	**		**
DFC3R			**	**	**
FC6				**	**
DFC6					**

(b) Translational RMSE

	DFC3T	DFC3R	FC6	DFC6	DFC6C
UC					**
DFC3T					**
DFC3R					**
FC6					**
DFC6					**

(c) Rotational RMSE

	DFC3T	DFC3R	FC6	DFC6	DFC6C
UC	**		**	**	**
DFC3T		**			
DFC3R			**	**	**
FC6				**	
DFC6					

(d) Mean translational velocity

	DFC3T	DFC3R	FC6	DFC6	DFC6C
UC	**		**	**	**
DFC3T		**	**		
DFC3R			**	**	**
FC6				**	
DFC6					

(e) Mean rotational velocity

Table 5.4: Results of Tukey's honest significant difference tests for the primary metrics of the six dimensional dynamic frictional constraint validation at the 95 % (**) confidence interval. The abbreviations for the test conditions are as defined in section 5.4.2.

the ring and pathway by the inner radius of the ring. This limited improvement from controlling orientation implies that, for this specific task, it was not possible for the users to overcome the positional difficulty associated with following the pathway.

The 6-DOF frictional constraints, without frictional redirection, (FC6) were included within the experiments to identify if, and how, the frictional redirection assisted the user. The mean contact percentage (69.9 %) shows that the performance for FC6 was significantly improved upon UC and DFC3R; however, it was significantly worse than DFC3T, DFC6 and DFC6C. These results conform to the hypothesis behind frictional redirection that, once a user penetrates a constraint, a conventional frictional constraint will permit them to move parallel to the constraint indefinitely. It is expected that, if the speed of deformation on the pathway was increased further, the difference between FC6 and those employing frictional redirection in translation would become even more pronounced.

The mean contact percentage (32.5 %) for the uncoupled dynamic frictional constraint (DFC6) was the most unexpected result from the experiments. As presumed, the contact percentage for DFC6 was significantly improved upon the poor performing assistance methods (UC, DFC3R and FC6); however, though it was slightly more effective, the contact percentage was not significantly different from DFC3T. This result implies that, in the selected task at least, adding a rotational constraint to a translational constraint does not significantly improve the assistance offered to the user. The likely reason behind the apparent ineffectiveness of DFC6 is that which was proposed earlier in this chapter for the introduction of coupled dynamic frictional constraints. In a task such as the dynamic steady-hand game, it is not sufficient to simply prevent the user from moving into the wrong orientation when twisting their hand. The target orientation is continuously changing as a result of deformation and translation of the target pose, and the user needs to permanently be correcting their orientation to match this. For frictional constraints to work, the user must provide some energy to the system in the form of motion in the tool. If the user is not rotating the tool, because they are concentrating on position, then they will never get any assistance from the constraint. Once the user makes a concerted effort to correct their orientation, the constraint can guide them to the target orientation, and prevent them from overshooting, but the benefit from this is lost to the amount of time it takes for the user to initially respond. With greater familiarity with the guidance forces generated by the active constraints, the latency in the user's response could likely be reduced, improving the constraint's effectiveness.

The coupled 6-DOF dynamic frictional constraint (DFC6C) had the lowest overall contact percentage (7.5 %) and significantly improved upon all other constraint methods. This result robustly shows the

efficacy of the coupled frictional redirection method within a task in which the tool orientation is important. The significant improvement of performance from DFC6 to DFC6C is due to the way in which the method redirects a user's energy from translation to rotation and vice-versa.

The high performance achieved by DFC6C can be understood further by considering the results for the translational and rotational RMSE. These metrics show how effective each constraint method was at assisting the user to stay in the optimal position or orientation as they moved along the pathway. The translational RMSE results are similar to the overall contact percentage results, with UC and DFC3R performing the worst, with FC6 next, followed by DFC3T and DFC6, then finally DFC6C the best. This ordering shows that, to some extent, the energetic coupling from rotational motion to translation forces benefits the positional accuracy of the user in this task. It is the rotational RMSE results, however, which are most telling. All of the methods, except the coupled dynamic frictional constraint, have very similar rotational RMSEs, at approximately 30 degrees. This shows that friction based constraints on orientation, with or without frictional redirection, do not assist the user particularly effectively. This corroborates the previous theory about the impotence of a frictional constraint in orientation to adequately support a task such as this, where the user is simply not rotating the tool in such a way that they get any benefit from the frictional torques. In these experiments, the coupled dynamic frictional constraint does, however, lead to a significant improvement in orientation control in a way which was not achieved by any of the uncoupled rotational constraints. This shows that the energetic coupling from translational motion to rotational torques is a definite benefit to the user and allows the advantages of constraining rotation to be maximised in this task. Whether the benefits from energetic coupling extend to a wider range of tasks is not proven in these experiments; however, the abstract, skill-based, nature of the task does imply a certain degree of generality and wider applicability.

The results for the user's velocity in translation and rotation are generally similar to one another, with users moving slowly for constraint methods UC and DFC3R and faster for the remaining cases. The user's velocity identifies how much confidence they have in performing the task, and gives a reasonable quantification of which methods the users felt assisted them the most. The slow usage of UC and DFC3R is unsurprising considering their poor performance in the other metrics. The users identify that the task is difficult to perform and move slowly to give them time to control both the position and orientation of the tool. The velocities recorded for the four remaining constraint methods were generally similar. This shows that the users did not necessarily identify which constraint methods were helping them more or less than others, likely because they were not informed about which assistance method was helping on

each test run. This meant that, at the start of each run, the users would move conservatively until they had identified the effectiveness of method for that run, at which point they could speed up if they felt it was appropriate. However, each run was relatively short and they never got much of an opportunity to establish which methods would help and how, and adjust their movement accordingly.

The results for the haptic power are not particularly indicative of the effectiveness of each constraint method; however, they are shown to illustrate an important property of the coupled dynamic frictional constraint. As expected, all of the methods had negative mean total power values for all of the experiments. This empirically confirms that the frictional constraint formulations are all dissipative and never exhibit active forcing of the user's hand. The interesting aspect of the haptic power results is in the rotational power distribution for the coupled dynamic frictional constraint. This distribution shows that, in some cases, the haptic device is generating positive power in rotation, and is highly negative in translation. Positive rotational power values occur when the coupled frictional redirection is taking energy from the user in translation, and applying it back upon the user in rotation, to correct for the error in their orientation. As the total power values are negative, the overall system remains dissipative, but the frictional redirection uses the users' energy most efficiently to guide them during the task.

The results for the coupled dynamic frictional constraints, presented in this chapter, make a strong case for their efficacy in dynamic tasks which require simultaneous control of both position and orientation. Though it has not been explicitly tested here, it is believed that the coupled dynamic frictional constraint formulation exhibits characteristics which are useful in all active constraints, not just those within dynamic environments. This claim is justified by describing a static active constraint applied to a conventional (non-dynamic) version of the steady-hand game. If one were to apply a standard, 6-DOF visco-elastic constraint in this task, which controlled the ring to the optimal pose on the pathway, then, while moving along the pathway, the system could be energetically active and generate autonomous motion of the tool, uncontrolled by the user, i.e. active forcing. The energetic activity would be caused by the continuously changing target orientation as the user translated along the pathway. Though the pathway itself is static, translational motion induces dynamism in the target orientation. Consider a scenario where the user was moving along the centre of a straight section of pathway in the correct orientation. As the user is in the correct pose, they would feel no resistance and the system power would be zero. If the user reached a sharp corner on the pathway, they could remain in largely the correct position, but with a drastically incorrect orientation. The visco-elastic constraint would then apply a large torque to the user to try to correct the orientation error, inducing significant energetic activity. This form of

active forcing is potentially just as harmful and undesirable as it is within a dynamic environment, and therefore it is argued that the coupled dynamic frictional constraint, which would not do this, is equally advantageous in a static 6-DOF constraint.

Though it has been proven that dynamic frictional constraints are effective at guiding a user, there are some limitations of the method. The main limitation is that the coupling between translation and rotation can cause the user to ‘stall’ when they do not respond to corrective forces. Within the experiments, stalling tended to occur when the user was moving along the pathway, close to the correct position, but with the incorrect orientation. When this happened, the dynamic frictional constraint generated a resistance in translation so that it could use the translational energy of the user to correct for the incorrect orientation. If, however, the translational resistance became too high, before the torque was high enough to correct the orientation, the user felt as though they were moving in the wrong direction and would stop. When this happened, the frictional redirection opposed their translational motion in any direction until they corrected the orientation. Generally, the torque corrected the users’ orientations before they got into this situation; however, it can occur. If stalling became particularly problematic for a system, in future work, it would be simple to define the frictional redirection angle θ based on the user’s velocity, so that when they stopped, the redirection would release them so that they could start moving again. Another solution, would be to define one frictional redirection angle for translation and rotation independently, and have a separate variable frictional redirection angle which could control the energy redirection between the two.

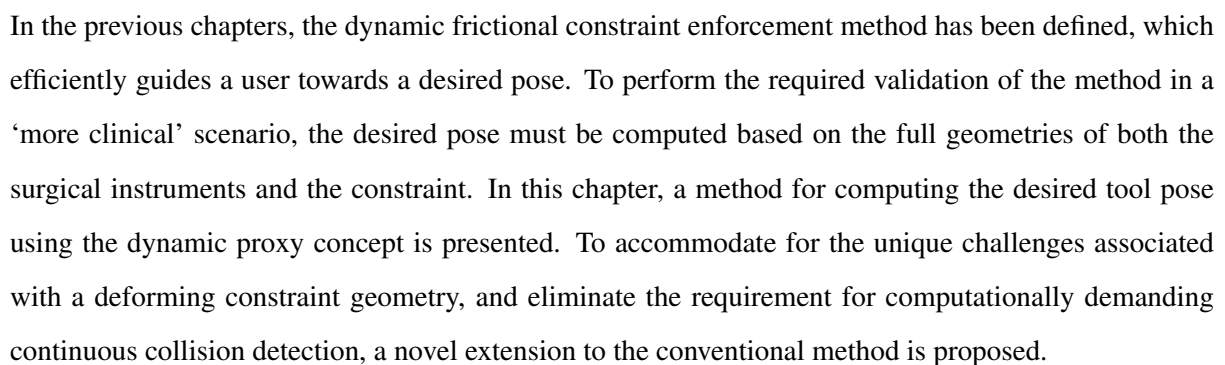
Another limitation of the coupled dynamic frictional constraint formulation is that it is more susceptible to low amplitude vibrations than the uncoupled methods. In order to compute the friction direction, the method considers the direction of constraint penetration in all six dimensions concurrently. This means that, if the components of the direction of constraint penetration are of low quality (due to signal noise or encoder discretisation), then the compounding of the six degrees of freedom will induce large changes in the friction direction, and hence vibrations. If necessary in the future, vibrations of this form can be suppressed by reducing the frictional stiffnesses. As the performance of the coupled dynamic frictional constraint is so much greater than that of the other methods, the small reduction in stiffness is unlikely to be problematic.

5.7 Summary

In this chapter, the dynamic frictional constraint concept has been rederived for translations in $\mathbb{R}^3$, proven to be dissipative and has been extended for rotations in $SO(3)$. The energy redirection in the new formulation is based around the abstracted projection of a vector into a cone. The abstraction of this operation means that it can be applied to constrain both rotation and translation in an almost identical way. In addition to an uncoupled dynamic frictional constraint which independently constrains both position and orientation, a coupled dynamic frictional constraint has been proposed. The coupled constraint applies frictional redirection of energy between translation and rotation in such a way that it can effectively constrain both, even when the user is only focusing on one.

Within this chapter, the dynamic frictional constraint formulation has been fully defined and some abstract experiments have shown its efficacy. In the following chapter, a method for defining the target pose of a tool geometry in contact with a complex and deforming constraint surface is explained. In the subsequent chapter, these methods are combined together and applied to a clinically grounded, simulated scenario with a physical soft tissue simulation, where the potential benefits of dynamic frictional constraints in a tissue dissection task can be evaluated.

A Dynamic Proxy Formulation for Deforming Geometries



In the dynamic active constraint experiments in chapters 4 and 5, the target poses for the tool (i.e. the poses to which the active constraints guided the users) were defined explicitly based upon the closest points on the pathways to the tools' actual positions. This meant that the target poses did not directly

consider the full geometries of the tools, only the poses of their representative coordinate frames. Within a practical surgical, scenario the entire body of the surgical instruments can penetrate and damage the constrained anatomy and therefore the full geometry of the tool needs to be considered in the system and constrained.

As detailed in chapter 3, the constraint evaluation between a tool geometry and a constraint can be performed using deformation invariant bounding sphere (DIBS) hierarchy proximity queries. However, simply identifying the relative configuration is not sufficient to define the target pose necessary for the dynamic frictional constraints. A dynamic proxy simulation, as introduced in section 2.6.2, can be used to establish the target pose for the active constraint, based on the relative configurations computed using the DIBS hierarchy.

Proxies, also known as ‘god objects’, were initially proposed for haptic rendering in virtual environments, and involve computing the motion of a simulated version of a tool which follows the actual tool, while not penetrating any virtual objects within its workspace [Ruspini et al., 1997; Zilles and Salisbury, 1995]. In general, proxy methods employ a virtual linkage formed of a spring, and possibly a damper, between the actual tool and the proxy tool to define its ideal motion, and then restrict the motion using a range of techniques. Figure 6.1 illustrates how a proxy could be applied to define a non penetrating pose for a scalpel colliding with an active constraint. It can be seen that the virtual linkage pulls the proxy as close towards the actual tool as the tool and constraint configuration will allow.

Proxies present many advantages for use with an active constraint. One of the most important is that in most methods there is temporal coherence between the target poses generated using a proxy. The simulated dynamics of the proxy means that it is unlikely to make large discrete jumps in position within the workspace. Dynamic frictional constraints ensure that sudden jumps in target pose do not generate force spikes which could jerk the surgeon’s hand; however, they do not currently prevent the sudden removal of force, which could be undesirable for the surgeon.

Another advantage of proxies is that they prevent ‘pop-through’, when the actual tool reaches the middle of a constrained region. Pop-through is where the target pose suddenly switches sides of a constrained region, such that users are encouraged to pass completely through it, rather than retract along the pathway by which they entered the constraint. When protecting a tissue structure with a constraint it can be vastly more preferable that the surgeon penetrates most of the way through the tissue before retracting along the same path, rather than completely severing it.

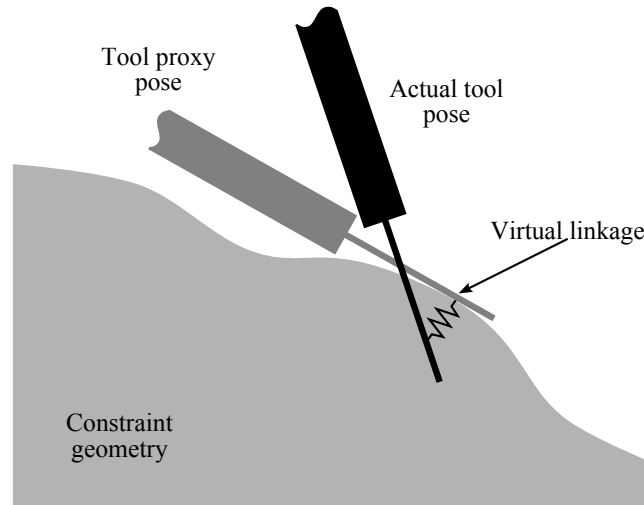


Figure 6.1: Illustration of a proxy and virtual linkage configuration for a scalpel-like surgical tool. It can be seen that the actual tool has penetrated the constrained region, but the proxy remains on the surface. The virtual linkage between the actual and proxy tools ensures that the proxy follows as close as possible the pose of the actual tool, while remaining on the surface.

There are three primary approaches within the haptics literature for modulating the effects of the virtual linkage in such a way that the proxy does not penetrate any virtual objects. ‘Penalty-based methods’ compute a set of penalty forces upon the proxy, based on the magnitude of its penetration into the other objects [Barbic and James, 2008; McNeely et al., 1999; Otaduy and Lin, 2006]. By combining the penalty forces from the proxy penetration points with the forces from the virtual linkage, the overall motion of the proxy can be computed which minimises penetration while following the actual tool. Though penalty-based methods can be simple to implement, the fact that there must be some penetration before the penalty forces are generated means that there will be some inaccuracy in the proxy placement. An additional issue with certain penalty based methods in a dynamic environment is that the movement of a surface can suddenly release an embedded proxy. When this happens the proxy will suddenly accelerate, unconstrained, towards the actual tool, producing discontinuities in the constraint. ‘Impulse-based methods’ generate impulse response forces from each contact between the proxy and objects [Mirtich and Canny, 1995]. These methods perform well in certain configurations; however, it is generally necessary to integrate penalty forces in order to prevent slow penetrations during resting contacts [Constantinescu et al., 2004, 2005].

For a dynamic active constraint, smoothness of proxy motion is of paramount importance, and for this reason a ‘constraint-based method’ for computing proxy motion was selected. Within a constraint-based method, the unconstrained acceleration of the proxy, resulting from the virtual linkage, is first computed,

and then the contact information is used to project this unconstrained acceleration into one which is as close to it as possible, without penetrating any other objects [Baraff, 1989; Ortega et al., 2007; Redon et al., 2002]. The primary advantage of constraint-based methods is that they guarantee that the proxy will not penetrate any objects and its motion will be smooth and consistent. However, the process requires convex optimisation to establish the constrained projection of the unconstrained acceleration and continuous collision detection, both of which can be computationally demanding.

In this chapter it is explained how the conventional constraint based method for defining proxy dynamics can be applied in a computationally efficient way for use with dynamic active constraints. By considering a wider range of potential contacts between the tool and constraint geometries, the novel extension to the method proposed here eliminates the requirement for continuous collision detection and allows the method to continue to operate accurately when the constraint surface deforms. Initially, the conventional constraint-based method from [Ortega et al., 2007] is briefly explained, before it is shown how it can be adapted specifically for dynamic active constraints.

6.2 Computing Constraint-based Proxy Motion using Gauss' Principle

A well established method for computing constraint-based proxy motion with six degrees of freedom is described in [Ortega et al., 2007], based on [Redon et al., 2002]. This method is briefly described here for completeness. The first stage in defining the motion of a proxy, at a given time-step, is to compute its unconstrained acceleration $\mathbf{a}_u \in \mathbb{R}^6$, due to the virtual linkage between it and the physical tool, as follows:

$$\mathbf{a}_u = K_{vl} \mathbf{x}_{vl}, \quad (6.1)$$

assuming unit mass and inertia, where $K_{vl} \in \mathbb{R}^{6 \times 6}$ is the diagonal stiffness matrix for the virtual linkage and $\mathbf{x}_{vl} \in \mathbb{R}^6$ is the extension of the virtual linkage, computed from the poses of the proxy and the physical tool:

$$\mathbf{x}_{vl} = [\mathbf{x}_{t,vl}, \mathbf{x}_{r,vl}], \quad (6.2)$$

where $\mathbf{x}_{t,vl} \in \mathbb{R}^3$ is the translational extension and $\mathbf{x}_{r,vl} \in \mathbb{R}^3$ is the rotational extension in $\text{SO}(3)$.

From Gauss' principle, it is known that the constrained acceleration of an object $\mathbf{a}_c \in \mathbb{R}^6$ (in this case the proxy) is the one which minimises the, so-called, 'kinetic distance' between the unconstrained and

constrained accelerations:

$$\mathbf{a}_c = \arg \min_{\mathbf{a} \in A_c} \frac{1}{2} (\mathbf{a} - \mathbf{a}_u)^T M (\mathbf{a} - \mathbf{a}_u), \quad (6.3)$$

where A_c is the set containing all permissible accelerations of the body which do not penetrate the surrounding objects and $M \in \mathbb{R}^{6 \times 6}$ is the proxy's diagonal mass matrix.

Within the constraint-based proxy method, the set A_c is defined using a contact Jacobian $J_c \in \mathbb{R}^{6 \times n}$ which defines the separating acceleration of the proxy, along the virtual object surface normals, at n discrete points where it makes contact with the virtual objects within its workspace. i.e. when the contact Jacobian is multiplied by the acceleration of the centre of mass of the proxy, it produces a vector of length n which defines the linear acceleration of the proxy away from the surrounding virtual objects at each contact point, along the contact normals. To ensure that the proxy does not penetrate the virtual objects, it is necessary for all elements of the produced vector to be non-negative. Formally, J_c defines A_c as follows:

$$A_c = \{ \mathbf{a} : \mathbf{a} \in \mathbb{R}^6, J_c \mathbf{a} \geq 0 \}. \quad (6.4)$$

Establishing $\mathbf{a}_c$ from equations (6.3) and (6.4) is a quadratic programming problem and can be solved in a variety of ways, e.g. [Boyd and Vandenberghe, 2004; Wilhelmsen, 1976]. Once $\mathbf{a}_c$ is known at a given time, it is necessary to integrate the motion of the proxy across the subsequent time-step, in the direction of $\mathbf{a}_c$. As the constraint-based method relies on surface contacts, continuous collision detection is used for the motion across the time-step to ensure that its motion is stopped if the object makes contact with any other objects in its workspace. If collisions are detected, then J_c is recomputed from the new contact points and a new value for $\mathbf{a}_c$ is solved.

This process of computing constrained acceleration and continuous collision detection for a simplified proxy is shown in figure 6.2. It can be seen that the proxy moves in discrete steps, ensuring that it does not penetrate the restricted region.

6.3 An Implementation for Deforming Bodies

There are two reasons which make implementing the above method with a high resolution deforming body challenging. The first is that the continuous collision detection is computationally demanding for a deforming body. As explained in chapter 3, collision detection is difficult in deforming bodies

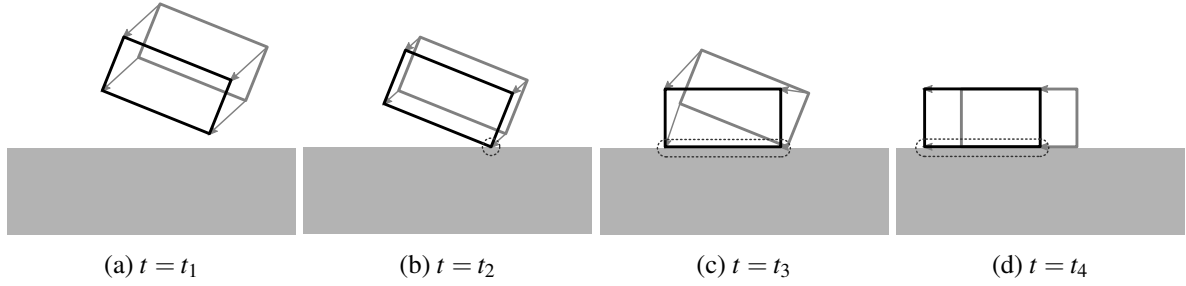


Figure 6.2: Illustration of the conventional proxy motion scheme used in [Ortega et al., 2007], where the actual tool (not shown) is moving down and to the left while the rectangular proxy attempts to follow it. The grey region represents the region into which the proxy is not permitted. In this method, continuous collision detection is used between each time step to ensure that the proxy does not penetrate the surface. Highlighted at each time-step are the contacts between the proxy and surface which will be used to define the constrained acceleration of the proxy at the subsequent step.

because conventional search structures become invalidated after deformation. Most continuous collision detection methods rely on repeated discrete collision detection searches and this can be prohibitively complex.

The second is that the deformation of the body can cause a proxy to become embedded within the virtual object when it deforms, and the algorithm needs to be able to resolve this. The definition of A_c in equation (6.4) only ensures that surface contacts do not become penetrations due to the proxy motion, it does not eliminate penetrations which have already occurred.

The solution to these problems is to compute the contact Jacobian based on a wider set of points which are ‘close to’ contacts, or already in penetration. Based on the expected velocity of the constrained tool, it is possible to estimate an upper bound for the displacement of all points on the surface of the proxy in a single time-step Δx_{max} , as follows:

$$\Delta x_{max} = \Delta t (\dot{x}_{max} + \omega_{max} r_{max}), \quad (6.5)$$

where Δt is the time-step, $\dot{x}_{max}$ is the estimated maximum translational velocity of the tool’s centre of mass, ω_{max} is the estimated maximum angular velocity of the tool’s centre of mass and r_{max} is the maximum radius of any point on the tool from its centre of mass.

By identifying all point pairs between the proxy and virtual objects which are within the upper displacement limit, a contact Jacobian can be assembled which ensures that none of these point pairs penetrate during the following time-step. As all potential contact locations for the subsequent time-step are consid-

ered by the single contact Jacobian, continuous collision detection is not required and it can be assured that the proxy will not enter the virtual object, or, in the case of an active constraint, the constrained region.

As close contact point pairs are not necessarily collocated at the start of the time-step, simply ensuring their relative acceleration is non-negative is not sufficient to generate correct motion which follows the surface of the virtual object. Instead, the accelerations at each close contact point must be set individually, based on the proximity, such that, at the end of the next time-step, they remain non-penetrating. This formulation for constraining the acceleration of the proxy also resolves the issue of penetrations due to surface deformation because it can recover point pairs which have penetrated. Rather than having negative limits for the acceleration for each contact point, which prevent penetrations from occurring during the subsequent time-step, a positive limit can be set for the acceleration of an already penetrating contact point which ensures that the proxy accelerates away from the contact such that the penetration is eliminated by the end of the time-step.

Within an active constraint, the aim is generally to stop a user from moving before they violate a constraint. As impedance controlled devices used for enforcing active constraints have a finite stiffness, this means that the enforcement should start before the user begins violating a constraint so that the forces can build and stop the user stably. For a regional constraint this means offsetting the active constraint boundary from the surface of the tissue being protected by a fixed distance d_{offset} [Jakopec et al., 2003b]. This surface offset is beneficial for the proposed proxy formulation because it means that all close contacts and penetrations can be identified and treated in a unified way. A range query, as detailed in algorithm 2 in chapter 7, can identify all triangles within the constraint mesh which are within a pre-defined distance of the surgical tool. If the range value r is the sum of the active constraint offset and the proxy displacement upper bound:

$$r = d_{offset} + \Delta x_{max}, \quad (6.6)$$

then all triangles found by the range query should have close contact points included within the contact Jacobian so that the constrained acceleration ensures that they are non-penetrating by the end of the following time-step. This range query variation on the conventional constraint-based method is illustrated in figure 6.3.

To implement this method using Gauss' Principle it is necessary to substitute the set of permissible accelerations based on contacts, A_c , in equation (6.3) with a set of permissible accelerations based on

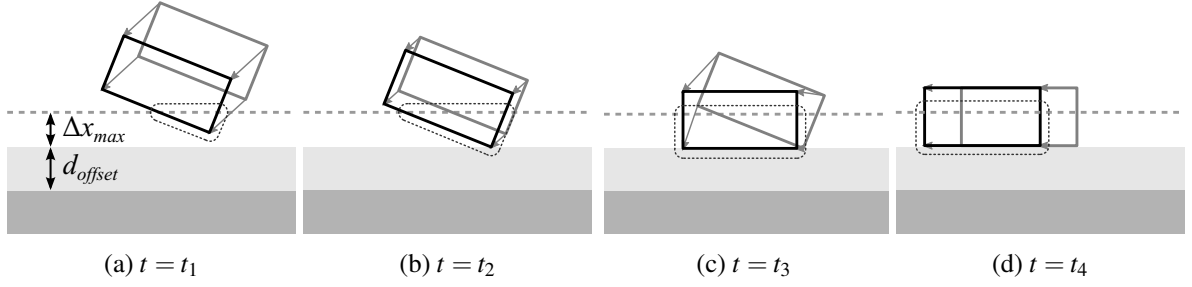


Figure 6.3: Illustration of the proxy motion scheme used in the dynamic active constraint which avoids the necessity for continuous collision detection. The actual tool (not shown) is moving down and to the left while the rectangular proxy attempts to follow it. The dark grey region represents the region into which the proxy is not permitted, the light grey region represents the constraint offset of thickness d_{offset} and the dashed grey line represents the boundary inside which close contacts are considered, Δx_{max} from the constraint surface. In this method, range queries are used between the proxy and surface to identify close contacts, as highlighted, which are used to form the contact Jacobian.

close contacts, A_p , defined as follows:

$$A_p = \{ \mathbf{a} : \mathbf{a} \in \mathbb{R}^6, J_p \mathbf{a} \geq \mathbf{b} \}, \quad (6.7)$$

where $J_p \in \mathbb{R}^{6 \times m}$ is the proximity Jacobian for the m points which are within the close contact region and $\mathbf{b} \in \mathbb{R}^m$ is the acceleration limit vector for each of the close contact points. The i -th element of the acceleration limit vector $\mathbf{b}_i$ can be computed from the close contact point details, based on the standard equations of motion, as follows:

$$\mathbf{b}_i = 2 \frac{\Delta x_{pr,i} - \dot{x}_{pr,i} \Delta t}{\Delta t^2}, \quad (6.8)$$

where $\Delta x_{pr,i} \in \mathbb{R}$ is the close contact point pair separation projected onto the virtual object surface normal $\hat{\mathbf{n}}_i$, $\dot{x}_{pr,i} \in \mathbb{R}$ is the close contact point pair relative velocity projected onto the virtual object surface normal and Δt is the time-step. The motion projections are computed using the following equations:

$$\Delta x_{pr,i} = (\mathbf{x}_{ac,i} - \mathbf{x}_{pr,i}) \cdot \hat{\mathbf{n}}_i + d_{offset} \quad (6.9)$$

$$\dot{x}_{pr,i} = -\dot{\mathbf{x}}_{pr,i} \cdot \hat{\mathbf{n}}_i, \quad (6.10)$$

where $\mathbf{x}_{ac,i} \in \mathbb{R}^3$ is the Cartesian position of close contact point i on the constraint mesh, $\mathbf{x}_{pr,i} \in \mathbb{R}^3$ is the Cartesian position of close contact point i on the proxy and $\dot{\mathbf{x}}_{pr,i} \in \mathbb{R}^3$ is the Cartesian velocity of close contact point i on the proxy.

To ensure that the proxy's motion along the surface was smooth, two adaptations were made to the above formulation. The first was that edge normals were computed explicitly in equations (6.9) and (6.10) when the close point on the constraint mesh $\mathbf{x}_{ac,i}$ was on the edge of a triangle rather than using the surface normal of the triangle. The edge normals were computed using the following equation:

$$\hat{\mathbf{n}}_i = \frac{\mathbf{x}_{pr,i} - \mathbf{x}_{ac,i}}{\|\mathbf{x}_{pr,i} - \mathbf{x}_{ac,i}\|}. \quad (6.11)$$

This formulation for the normals ensured that the convex edges of the constraint were rounded rather than sharp and did not cause erratic motion in the proxy. The second modification was that the set of close contacts identified by the range query were filtered to remove any which had inconsistent normals.

6.4 Discussion

The dynamic proxy formulation described in this chapter allows for the computation of a target tool pose which follows that of the actual tool, while not penetrating any of the constraint geometries. The method adapts the conventional Gauss' principle method so that it can operate with deforming constraint geometries by optimising the motion based on a set of 'close contacts' between the tool and constraint rather than actual contacts. The wider range of contacts considered by the method allows it to operate without the requirement for continuous collision detection and also allows it to recover intersecting geometries which can occur as the result of constraint deformation.

As continuous collision detection is computationally demanding, particularly with a deforming mesh, the removal of this requirement for the constraint evaluation could have a significant implication for the resolution of the geometries which can be used in the constraint. As with the deformation invariant bounding sphere hierarchy described in chapter 3, the control rate required for a smooth and stable haptic interaction with an active constraint is approximately 1 kHz. The computational demands of the constraint system therefore limit the size, and hence resolution, of the geometries used to define the constrained regions. By reducing the computational demand of the constraint evaluation process, the proxy method presented here could allow for higher resolution constraint surfaces which fit to, and hence protect, the underlying patient's anatomy more effectively.

Due to its requirement for integration with other components in the active constraint system, the proposed dynamic proxy formulation was not experimentally validated in isolation. Instead, the dynamic

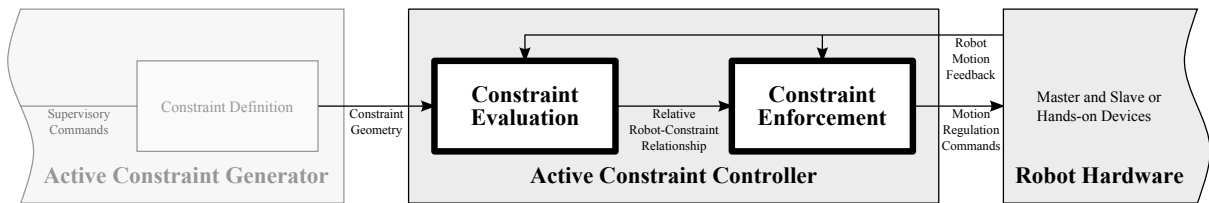
proxy method described in this chapter was implemented and used within the complete active constraint system applied to the clinically founded task described in chapter 7. A detailed discussion of the effectiveness of the proxy method is therefore given then.

6.5 Summary

In this chapter, a dynamic proxy formulation for defining a surgical instrument target pose has been presented. The method is based those in the literature using Gauss' principle and extends them for efficient use with deforming constraint geometries. In the following chapter, the proposed proxy formulation is implemented and combined with the remaining elements required for a complete active constraint system, as introduced in the previous chapters of this thesis. The combined system is validated with an experimental assessment of a simulated dissection task based on a physical soft tissue model.

Chapter 7

Integrated Trials within a Surgical Simulation Environment



In this chapter, the deformation invariant bounding sphere hierarchy from chapter 3 is combined with the dynamic frictional constraint formulation from chapters 4 and 5 and the dynamic proxy method from chapter 6, in order to validate their combined efficacy within a simulation of a clinically founded task. In addition to the components taken from earlier in this thesis, a physical soft tissue simulation of a nerve dissection was created to allow the experimental investigation of the active constraint efficacy.

7.1 Introduction

Within the three preceding chapters, independent elements of a dynamic active constraint system have been conceived, formalised and validated. All of the experiments within the previous chapters have been based on somewhat abstract, non-clinical tasks which demonstrated specific aspects of a dynamic active constraint system. In order to investigate the proposed concepts within a clinical task and validate the methods when combined together, a surgical simulation environment was assembled.

It was decided to perform this final validation within a simulated procedure, rather than a physical procedure, for two reasons. The primary reason was that it precludes the necessity for the constraint generation process. Within a dynamic active constraint, the constraint generation, i.e. establishing where the constraint geometry is within the environment, is particularly challenging as the environment and constraint can change continuously. This aspect of a dynamic active constraint has not been investigated within this thesis and there are currently very few generally applicable methods described within the literature. By simulating the surgical environment, the constrained anatomy, and hence the constraint, can be definitively known. An additional benefit to using a simulated surgical environment is that quantifying aspects, such as tissue penetration, can be achieved accurately.

The surgical scenario selected for simulation was the robot assisted dissection of a nerve from its surrounding soft tissue. Nerves permeate many tissue structures throughout the body and it is often important that they are protected when the surgeon is working within close proximity to them. Poor visibility, inter-patient anatomical variation and the small size of many nerves can make this challenging for the surgeon and damage can occur [Toninato et al., 2004]. Nerve damage is generally classified into three degrees of severity: neurotmesis; axonotmesis and neurapraxia [Seddon, 1942]. Neurotmesis refers to a complete division of the nerve such that it cannot transmit any signals and will not heal without surgical repair. When the fibres within the nerve are broken, but the supporting structure remains intact, transmission of signals is blocked, but the likelihood of natural healing is significantly increased. This form of nerve damage is referred to as axonotmesis. The least severe form of nerve damage, neurapraxia, occurs when a nerve's conductivity is temporarily impaired; however, the nerve fibres themselves are not severed. The recovery from this type of nerve injury is both rapid and complete.

When nerves are damaged during surgery, and there is a partial (paresis) or complete (paralysis) interruption of the signals being transferred through them, there are a range of side effects for the patient, depending on the nerve's function. For example, during a radical prostatectomy, which is widely performed with the da Vinci surgical system (Intuitive Surgical, Inc.) [Badani et al., 2007], damage to the nerves within the pelvic plexus, adjacent to the prostate gland, can lead to erectile dysfunction [Dubbelman et al., 2006] and urinary incontinence [Sacco et al., 2006]. The proximity of nervous tissue to the prostate has led to the development of the 'nerve sparing' radical prostatectomy [Walsh and Mostwin, 1984; Walsh et al., 1983], in which, the conventional procedure is modified based on the anatomical relationships with the pelvic plexus to reduce nerve damage. However, even with the nerve sparing prostatectomy, the effects of nerve damage are still found [Catalona and Bigg, 1990].

It is hypothesised here that a tubular dynamic active constraint could be applied along the body of a nerve, in such a way, that it can prevent direct, accidental damage due to surgical instruments cutting or crushing the nerve. It is this scenario which is used to validate the research from the previous chapters in a simulated surgical task here. The conceptual basis for the simulation is a robotic thyroidectomy [Kang et al., 2009]. In addition to already being a robotic procedure, the thyroidectomy is suitable for demonstrating a dynamic active constraint because of the recurrent laryngeal nerves (RLN), that are posterior to the two lobes of the thyroid gland, and which must be dissected away from the thyroid before it can be removed. Occurrences of damage to the RLN during thyroid surgery are well documented within the literature [Jatzko et al., 1994; Wagner and Seiler, 1994]. The possible effects of nerve damage include weakness or complete loss of voice [Crumley, 1994], strained breathing and obstruction of the airway [Jeannon et al., 2009].

Within the remainder of this chapter, it is demonstrated how an active constraint can be applied to protect a nerve while it is dissected from a larger soft tissue block, loosely based upon the anatomy of a human thyroid and RLN.

7.2 Simulated Nerve Dissection Concept

The custom simulation created for the nerve dissection is shown in figure 7.1. The simulation was of a hypothetical, teleoperated surgical robot, similar to the da Vinci Surgical System (Intuitive Surgical, Inc.), with accurate global position control of the end effectors and haptic feedback at the master console. There were four primary elements within the simulation: a soft tissue block (approximating a lobe of the thyroid); an embedded soft tissue tube (approximating the RLN); a robotic forceps; and a robotic scalpel. The two tools within the simulation were controlled by two haptic devices which simulate the haptic master console.

The soft tissue block in the simulation was approximately $45 \times 20 \times 15$ mm, based on the thyroid lobar dimensions in [Brander et al., 1992], and the nerve diameter was 2 mm, based on the RLN diameter in [Serpell et al., 2011]. The path of the nerve across the surface of the tissue block was generated arbitrarily. The simulated forceps was based on the da Vinci Surgical System DeBakey forceps (Intuitive Surgical, Inc.) and the simulated scalpel was based on the da Vinci Surgical System Snap-fit Scalpel (Intuitive Surgical, Inc.).

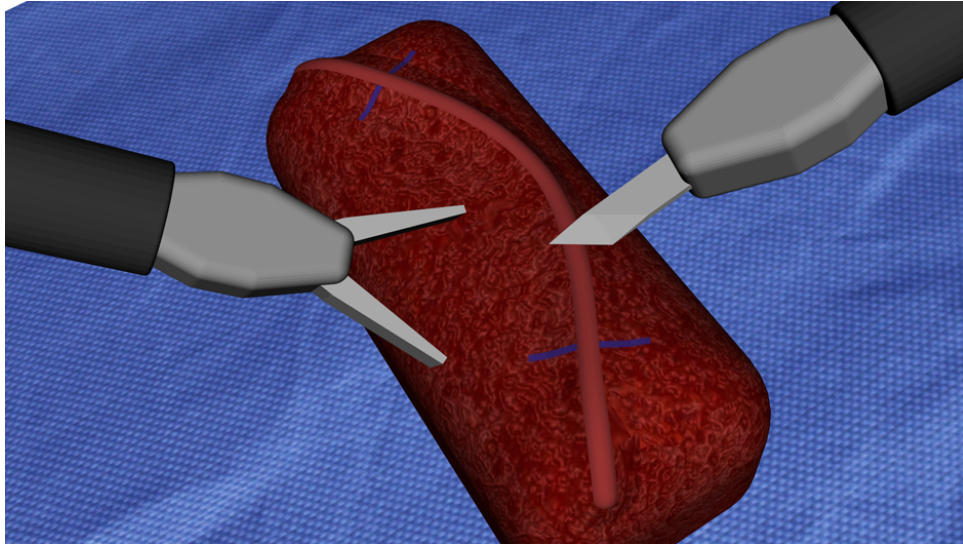


Figure 7.1: Preview of the clinical simulation developed for the validation of dynamic active constraints. Shown are: the soft tissue block; the embedded soft tissue tube; the left-handed, robotic forceps; and the right-handed, robotic scalpel.

To dissect the nerve from the tissue block within this simulation, it was necessary to manipulate the soft tissue block with the forceps, while parting the nerve from the tissue block with the scalpel. It should be noted that the dissection of the RLN is actually performed using a spreading motion with the da Vinci Surgical System Maryland Dissector (Intuitive Surgical, Inc.). Though such a tool could be constrained using the dynamic active constraint concepts presented here, for this initial validation, a simpler scalpel was deemed to be more appropriate.

The active constraints within the simulation were applied to prevent the blade of the scalpel from penetrating or crushing the nerve. It was assumed for the simulation that the forceps would be far enough from the nerve during normal use that constraints were not necessary to ensure that they did not cause damage, while the scalpel was used in close proximity to the nerve and thus needed to be constrained.

7.3 A Complete 6-D Dynamic Active Constraint System

The complete system used for the final validation of the dynamic active constraint research within a surgical simulation environment is shown in figure 7.2. The system consists of four primary components: an active constraint controller; a physical soft tissue simulation; an interface with the haptic devices; and a 3-D visualisation.

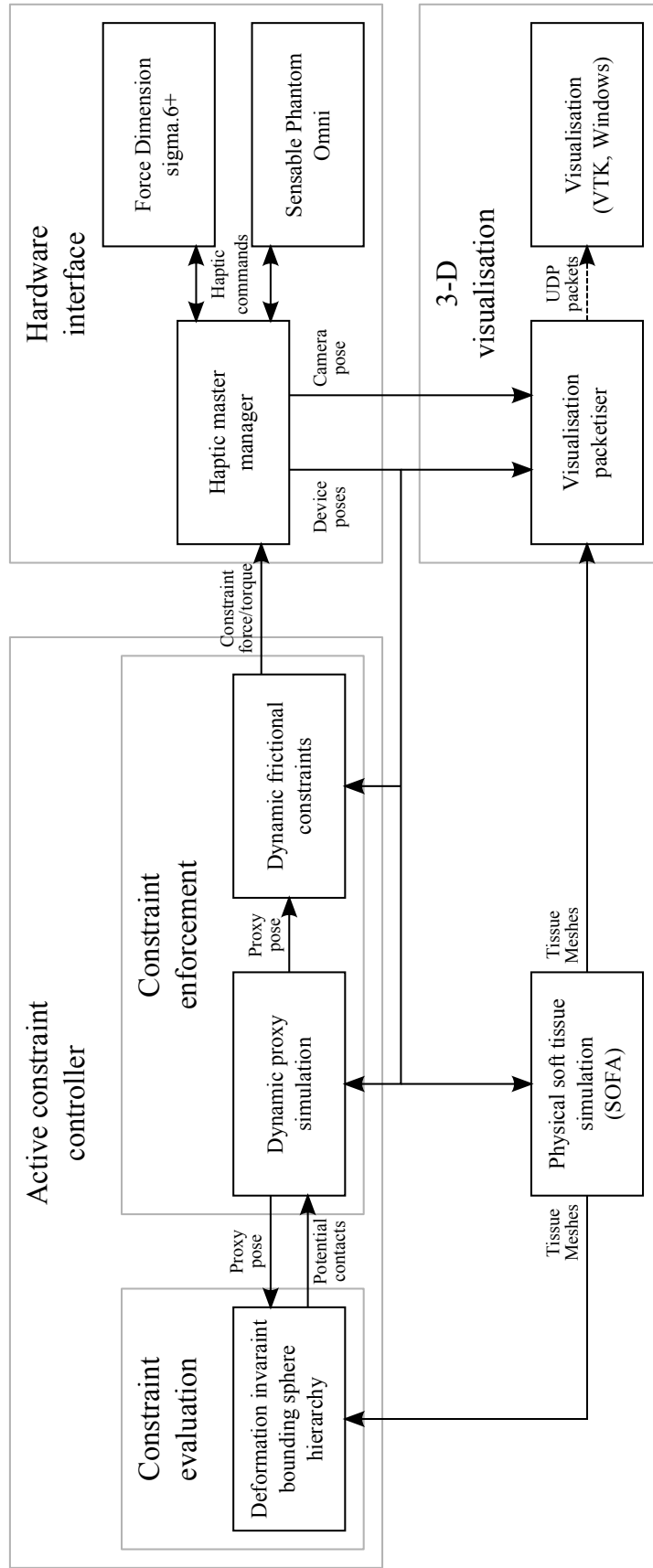


Figure 7.2: System diagram for the software used within the simulated surgical environment as implemented within the OROCOS framework. Included are: the active constraint controller, based on the research shown in the previous chapters; the physical soft tissue simulation, based on the simulation open framework architecture (SOFA); the hardware interface for the haptic devices; and the visualisation, based on the Visualization toolkit (VTK).

The system was implemented in multi-threaded C++ using the ‘Open Robot Control Software’ (OROCOS) [Bruyninckx, 2001] for inter-module communications and synchronisation. Due to the variation in computational demand of the different modules, the OROCOS framework allowed for a multi-rate system in which the components each ran at a separate frequency, thus ensuring stable haptic interactions. With the exception of the visualisation, all of the software was run through Linux on a quad-core, 3.0 GHz, Core i7 CPU (Intel, Corp.). The visualisation was run on a separate PC, through Windows (Microsoft Corp.), with a 2.93 GHz Core i7 CPU (Intel Corp.)/Quadro 4000 GPU (NVIDIA Corp.).

A description of the primary elements of the dynamic active constraint system follow.

7.3.1 Deformation Invariant Bounding Sphere Hierarchy

Within the active constraint controller, the nerve surface itself (as generated by the physical soft tissue simulation) was used to define the surface of the active constraint. This surface was represented computationally with a triangular mesh. Due to its geometric shape, the scalpel blade could be efficiently represented computationally with a single, convex quadrilateral.

As explained, before a constraint can be enforced, it is necessary to evaluate the relative configuration between the constraint and tool, which, in this case, means between the triangular mesh and quadrilateral. As the nerve was continuously deforming as it was cut away from the tissue block, in this task, the constraint evaluation was performed efficiently using a deformation invariant bounding sphere hierarchy, as described in chapter 3.

For the 6-DOF dynamic proxy simulation, discussed in chapter 6, it is not sufficient to identify the single closest point pair between the mesh and quadrilateral. Instead, a ‘range query’ is required, in which, the location of all triangles which are within a predefined proximity of the tool must be established. Range queries can be performed using a DIBS hierarchy with a variant of the recursive proximity query function described in algorithm 1. The recursive range query function for a DIBS hierarchy is shown in algorithm 2. Similarly to the proximity query method, the initial parameters for the algorithm are the DIBS hierarchy D , the current tool geometry T and the query range r . The algorithm returns the list of closest point pairs $[\langle cp_D, cp_T \rangle]$.

Within the complete simulation, for a nerve mesh of 1,909 vertices and 3,814 triangles, range queries using this method took consistently less than 1 ms and, as they were based on DIBS hierarchies, they

Algorithm 2 Recursive range query function

```
procedure DIBS-RANGE-QUERY( $D, T, r$ )  
   $n \leftarrow \text{root}(D)$  ▷ Initialise search node structure  
  return RECURSIVE-RQ( $n, T, r$ )  
end procedure  
  
procedure RECURSIVE-RQ( $n, T, r$ )  
   $[\langle cp_D, cp_T \rangle] \leftarrow \text{NULL}$  ▷ Initialise list of closest points pairs  
  if  $\text{leaf}(n)$  then ▷ Leaf node  
     $\text{poly} \leftarrow \text{getAssociatedPolygon}(n)$   
     $\langle cp_D, cp_T \rangle \leftarrow \text{findClosestPts}(\text{poly}, T)$   
     $\text{sep} \leftarrow \text{separation}(cp_D, cp_T)$   
    if  $\text{sep} \leq r$  then ▷ Create list of closest points pairs  
       $[\langle cp_D, cp_T \rangle] \leftarrow \text{pushBack}([\langle cp_D, cp_T \rangle], \langle cp_D, cp_T \rangle)$   
    end if  
  else ▷ Branch node  
    for all  $c \in \text{children}(n)$  do ▷ Compute child DIBS separations  
       $pp \leftarrow \text{getPinningPt}(c)$   
       $\text{rad} \leftarrow \text{getRadius}(c)$   
       $\text{dibssep} \leftarrow \text{separation}(pp, T) - \text{rad}$   
      if  $\text{dibssep} \leq r$  then ▷ Update list of closest points pairs  
         $[\langle c1, c2 \rangle] \leftarrow \text{RECURSIVE-RQ}(c, T, r)$   
         $[\langle cp_D, cp_T \rangle] \leftarrow \text{concatenate}([\langle cp_D, cp_T \rangle], [\langle c1, c2 \rangle])$   
      end if  
    end for  
  end if  
  return  $[\langle cp_D, cp_T \rangle]$   
end procedure
```

did not require rebuilds or refits after mesh deformation.

7.3.2 Dynamic Proxy Simulation

To define the target pose for the scalpel blade to which the active constraint guided the user, the dynamic proxy method described in chapter 6 was used. The dynamic proxy followed the pose of the scalpel, while not penetrating the constrained region surrounding the nerve. The proxy method for defining the target pose was particularly beneficial for the proposed nerve dissection task because of the way in which it prevents pop-through of the scalpel. This means that if the user does penetrate the nerve, then they will be encouraged to retract along the same pathway (causing axonotmesis which will heal naturally) rather than completely severing a nerve (causing neurotmesis which will require remedial surgery).

Parameter	Value
K_{vl}	diag(500, 500, 500, 1000, 1000, 1000)
M_{3D}	diag(1, 1, 1, 0.5, 0.5, 0.5)
M_{6D}	diag(1, 1, 1, 5×10^{-5} , 5×10^{-5} , 5×10^{-5})
d_{offset}	0.8 mm
Δx_{max}	0.5 mm

Table 7.1: Parameters used in the dynamic proxy simulation for the nerve dissection task.

The set of close contacts used to compute the proximity Jacobian J_p in the dynamic proxy method were identified using deformation invariant bounding sphere hierarchy range queries, as described in the previous section. The quadratic programming problem of computing the constrained acceleration of the proxy from the proximity Jacobian was solved using the Goldfarb-Idnani active-set dual method [Goldfarb and Idnani, 1983] as implemented in QuadProg++ [Di Gaspero]. For the nerve dissection experiments, the parameters within the dynamic proxy simulation were tuned empirically and set as shown in table 7.1.

Two different mass matrices were used within the dynamic proxy simulation during the experiments reported in section 7.4. The ratio between the top three elements on the mass matrix diagonal and the bottom three elements controls the prioritisation of position or orientation of the proxy during interaction with a constraint. For example, if the top three elements are much larger than the bottom three, then the translational motion of the proxy is prioritised and the orientation of the proxy will be more compliant. For a constraint in both translation and rotation, the proxy’s mass matrix should be balanced, i.e. M_{6D} , so that any penetration is spread between a translational error and a rotational error and the six force and torque components can be used most effectively.

If positional constraints in three dimensions are used, rather than full constraints in six dimensions, then only a translational error can be corrected by the system. In this situation, the orientation of the proxy should match the orientation of the actual tool as closely as possible. This means that tool’s entire penetration is considered in terms of a translation, with no rotational error, and can therefore be constrained effectively. Within the proxy formulation, this prioritisation of orientation can be achieved with a biased proxy mass matrix, i.e. M_{3D} .

Parameter	Value	Parameter	Value
f_C	15 N	τ_C	500 N.mm
$\sigma_{t,0}$	15 N.mm ⁻¹	$\sigma_{r,0}$	2,000 N.mm.rad ⁻¹
$\sigma_{t,1}$	0 N.s.mm ⁻¹	$\sigma_{r,1}$	0 N.mm.s.rad ⁻¹
$\sigma_{t,2}$	0.035 N.s.mm ⁻¹	$\sigma_{r,2}$	20 N.mm.s.rad ⁻¹

Table 7.2: Parameters used in the frictional constraints for the nerve dissection task.

7.3.3 Dynamic Frictional Constraints

Dynamic frictional constraints were used for the enforcement of active constraints within the surgical simulation. The constraints were implemented exactly as described in chapter 5, with the parameters as shown in table 7.2. The stiffness parameters were based on the capabilities of the master haptic device used to constrain the robotic scalpel. Due to the small sizes of the anatomy being simulated in the experiment, and the delicate nature of the task, inline with the capabilities of the da Vinci Surgical System [Hubens et al., 2003], five-times motion scaling was employed within the active constraint system. As this was used, the translational stiffness parameters of the dynamic frictional constraint could be increased accordingly (i.e. by five times), while remaining within the capabilities of the constrained haptic devices. The other slight variations between the parameters used for the nerve dissection and those used for the steady-hand game, given in table 5.1, are due to the differences in the general operating workspace.

At each time-step, the constraint penetration $\mathbf{p}_k$, necessary for the computation of constraint forces using the method in chapter 5, was defined as the transform from the target pose on the surface of the constraint (i.e. that of the proxy) to the tool’s pose. Using the dynamic frictional constraint formulation, new constraint forces and torques were computed within the system at a rate of more than 1,000 Hz.

7.3.4 Physical Soft Tissue Simulation

A fundamental requirement for the validation of dynamic active constraints within a surgical task is that the environment must deform throughout the procedure. This means that the nerve geometry must change as the user interacts with it in such a way that the constraint must be dynamic. Simulating ‘real-time’ interaction with, and deformation of, soft bodies is a computationally complex task. There are several physics simulators freely available; however, in general, their support of soft bodies is limited. To construct a surgical simulation environment, in which a user could interact with, and deform, soft

bodies, the open source library: ‘Simulation Open Framework Architecture’ (SOFA) was used [Allard et al., 2007]. In addition to functions for many common soft body simulation methods, the primary feature of SOFA is that it uses a multi-representation approach to simulating deforming bodies. This means that each of the components within the simulation is represented, simultaneously, in several different ways, each of which is specialised for one part of the overall simulation. The three primary representations taken by each component are: a mechanical model; a collision model; and a visual model. As a SOFA simulation progresses, interactions in the individual representation are mapped between one another, using a Barycentric mapping, such that they stay aligned and continue to represent the overall physical model together.

The SOFA simulation constructed for the physical soft tissue simulation consisted of four physical components: the tissue block; the nerve; the forceps; and the scalpel, of which, the former two components were deformable and the latter two were rigid. The poses of the forceps and scalpel were controlled within the simulation to follow the poses of the master haptic devices (with appropriate scaling and transformations) and the responses of the two soft bodies were output from the simulation for use in the remainder of the system. The details of the SOFA simulation for each of these components, as shown in figure 7.3, were as explained in the following sub-sections. With all elements combined, the simulation time for each time-step within the simulation ranged between approximately 5 and 55 ms depending on the deformation of the model and number of contacts.

7.3.4.1 Mechanical Models

To simulate the deformation of the tissue block and nerve during interactions, the finite element method (FEM) was used. Initially, the geometries were modelled as solid components in 3-D CAD software based on the dimensions given previously. The models were then meshed into volumetric, linear, tetrahedral elements using ICEM CFD (ANSYS, Inc.) by the Biomechanics Laboratory, Department of Mechanical Engineering, Imperial College, London, U.K.. The tetrahedral mesh for the tissue block consisted of 8,867 elements/1,949 nodes and the nerve mesh consisted of 607 elements/253 nodes. At each step of the simulation, the force fields across the deformable bodies were computed using the large strain, linear FEM and the time based integration of displacements was achieved using an implicit, conjugate-gradient Euler solver, all of which were available in SOFA.

The material properties for the two soft bodies within the SOFA simulation were selected for their ability

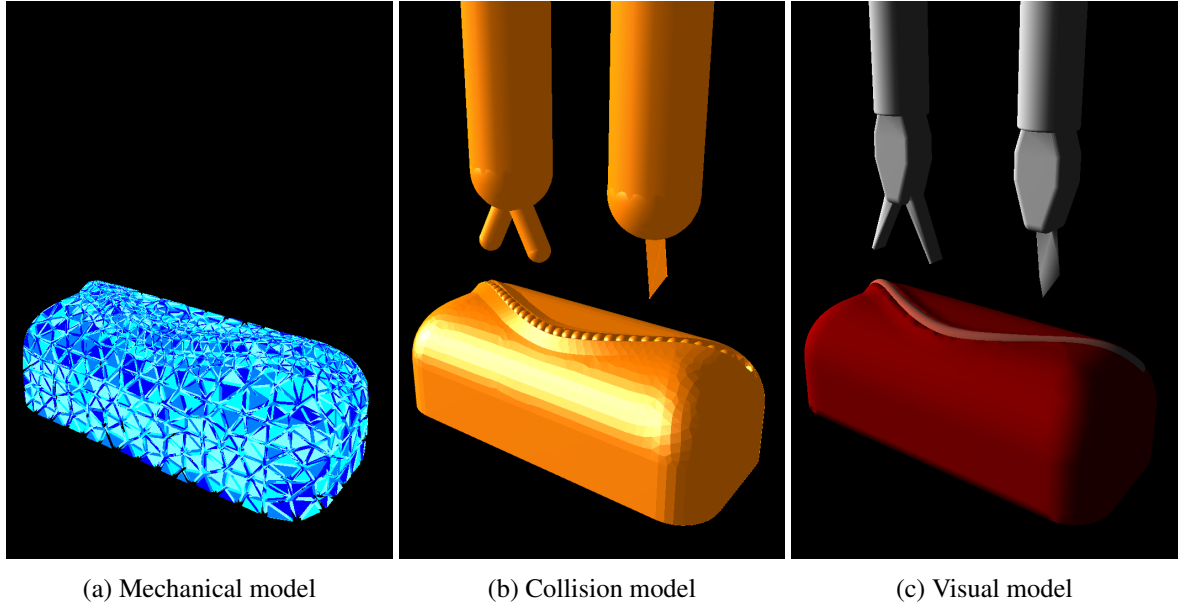


Figure 7.3: Images of the multi-representation models used for the physical soft tissue simulation. The mechanical model simulates the deformation and movement within the soft tissue structures, the collision model identifies and computes the responses from collisions between different elements within the simulation and the visual model contains the high-resolution soft tissue surfaces that constitute the simulated anatomy in the rest of the system. Within the SOFA library, the different representations of the scene are coordinated by a barycentric mapping between them.

Parameter	Solver value	Parameter	Tissue value	Nerve value
Rayleigh stiffness damping	0.07 s^{-1}	Mass	1 g	1 g
Rayleigh mass damping	1 s	Young's modulus	0.05 GPa	5 GPa
Velocity decay coefficient	1	Poisson's Ratio	0.35	0.35

Table 7.3: Parameters used in the mechanical models.

to respond in a stable and visually appealing way, rather than being based on realistic tissue properties taken from the literature. To ensure contact stability between the two soft bodies, it is necessary within SOFA for them to have similar masses, even if they have significantly different volumes. This guideline was therefore followed. The complete set of material and solver properties for the mechanical models are shown in table 7.3.

7.3.4.2 Collision Models

Due to the multi-representation approach taken within SOFA, it is possible to simplify the component geometries within the collision models in order to reduce the computational demand of identifying collisions. The collision models used for the four components within the nerve dissection simulation

are shown in figure 7.3b. The only component represented with a surface mesh for collisions was the tissue block, while the remaining three components had their geometries simplified to spheres and capsules. In addition to reducing the time taken to identify contact locations, using spheres and capsules improves the robustness of the simulation to low update frequencies. When surfaces are represented using ‘thin’ geometric primitives, such as triangles, lines and points, it is possible for them to completely pass through each other within a single time-step, such that the simulation does not identify the contact, unless continuous collision detection is used. As spheres and capsules are represented with a radius, they can recover contacts which have passed the surface, but are still embedded within the volume. This significantly improves the robustness of the collision detection and therefore also the collision response. The collision response used within the SOFA simulation was based on the magnitude of object interpenetrations, with a contact stiffness of 500 N.m^{-1} used to define the contact forces applied to the bodies.

7.3.4.3 Visual Models

Triangular meshes were used for the visual models for all four of the components within the physical simulation. These meshes were shown previously (with textures) in figure 7.1. Though these meshes played no role in the physical simulation, the surface meshes of the two soft bodies were output from the simulation and represented the exact geometries of the tissue block and nerve throughout the rest of the system. That is to say, these meshes were visualised, used to compute experimental metrics and used to define the active constraint geometries.

7.3.4.4 Manipulating the Models

Within the physical simulation, there were three modes by which the user interacted with the soft tissue models using the forceps and scalpel. General contact between the tools and tissue was covered by the collision models explained above; however, the cutting and gripping were both considered independently.

Free-form cutting within a deformable simulation is possible [Meier et al., 2005], though the meshes used within the simulation need to be updated as triangles and tetrahedra are cleaved. As this would lead to variable mesh topologies throughout the simulation, this would have required significant modification of the remaining dynamic active constraint components. Additionally, the computational demand of the

mesh updates would have reduced the simulation update rate. Instead, the cutting was simplified so that the user could only cut along the seam between the tissue block and nerve. With this simplification, it was possible to start the simulation with mechanical constraints along the collocated faces of the two meshes, fixing the nodes within the respective FEM meshes together, and then, as the user passed them with the cutting edge of the scalpel, releasing these constraints from the simulation such that the meshes could pull apart. This method was computationally efficient and gave a reasonable cutting effect during the dissection, as demonstrated in the experimental results reported in section 7.5.

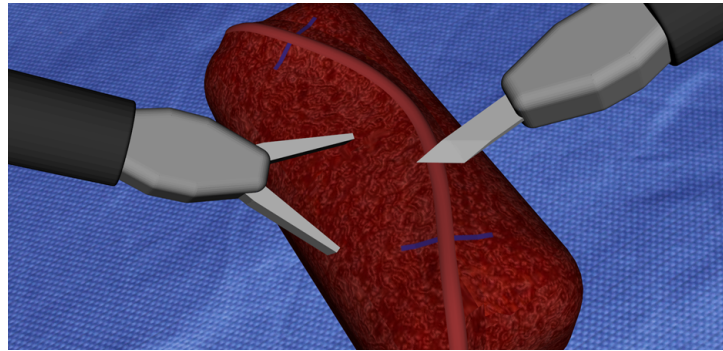
Rather than relying on the system's physics to grip the tissue between the grasping surfaces of the forceps, the gripping mechanism was 'hard-coded' into the simulation. When the grip command was sent from the user (as explained in section 7.3.5), three line segments were projected from the apex of the forceps across its width. If these rays intersected either of the soft tissue surface meshes, those triangles within the simulation were manually constrained to follow the forceps, until the grip command was released.

7.3.5 Hardware Interface

A Phantom Omni haptic device (Sensable Technologies, Inc.) and a Sigma.6+ haptic device (Force Dimension, Inc.) were used as the master devices within the simulation system to control the forceps and scalpel respectively. The alignment of the haptic devices to the simulation is shown in figure 7.4. To command the forceps to close, gripping the tissue, the user was required to press and hold the front button on the stylus of the Phantom Omni.

A haptic master manager was included in the system, between the haptic devices and the remainder of the system, to implement motion scaling. For the nerve dissection task, a motion scaling value of 5:1 was implemented, which meant that for each 5 mm moved by the user's hands, the simulated instruments would move by only 1 mm. It should be noted that motion scaling was not used in orientation due to the confusion this can cause once the difference in orientations of master device and simulation becomes large.

In addition to motion scaling, the haptic master manager included a clutching mechanism, similar to that on the da Vinci Surgical System (Intuitive Surgical, Inc.). When the user pressed the rear button on the stylus of the Phantom Omni, the translational motion of the simulated tools was stopped, allowing the user to move the master devices freely. The clutching of surgical instruments allowed the user to



(a) Simulated surgical instruments



(b) Master haptic devices

Figure 7.4: The alignment of the master haptic devices with the simulated surgical instruments. The left-hand haptic device, the Phantom Omni (Sensable Technologies, Inc.) controlled the forceps, and the right-hand device, the sigma.6+ (Force Dimension, Inc.) controlled the scalpel.

manoeuvre them around a larger workspace than would be achievable otherwise. As with the motion scaling, clutching of orientations was not used.

To allow the user to visualise the area in which they were cutting, and also dissect along the full length of the nerve, the user was able to control the pose of the camera, i.e. the view, within the simulation. The control of the camera was also inspired by the method in which the endoscope is controlled by the master console in the da Vinci Surgical System. When the grip button on the sigma.6+ was pressed, the simulated surgical instruments were clutched, and then the user could move the view with the right-hand master device.

7.3.6 3-D Visualisation

To display the simulation to the user, a 3-D visualisation of the surgical scene was constructed using the Visualization ToolKit (VTK, Kitware, Inc.). This visualisation is shown in figure 7.1 and was presented

to the user on a 3-D monitor using NVIDIA 3D Vision (NVIDIA Corp.) on a dedicated visualisation PC. The communication between the control PC, which hosted all of the other software components, and the visualisation PC was via the user datagram protocol (UDP). As the topologies of the surface meshes for the two soft bodies were fixed, only the locations of the vertices were transmitted for each frame, along with the poses of the instruments. A frame-rate of 50 Hz was used for the visualisation and communication in order to ensure the tool motion looked smooth, even though the soft bodies were not updated so frequently.

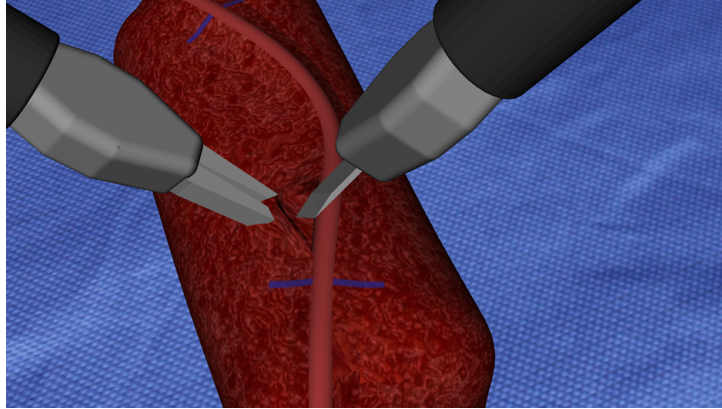
7.4 Experiments

To validate the dynamic active constraints within the surgical simulation described previously, several users performed repeated dissections of the nerve from the tissue block, while dynamic active constraints provided assistance to them. In each test, a user dissected the nerve from the tissue block, in the area between the two blue marks, as shown in figure 7.5. To dissect the nerve, the user was required to tension the tissue block with the forceps and then cut along the nerve-tissue interface using the scalpel. To prevent ‘damaging’ the nerve, the users were told to avoid cutting or crushing it with the scalpel while they performed the dissection, and they were also not permitted to grasp the nerve directly with the forceps, apart from when visually inspecting to show task completion.

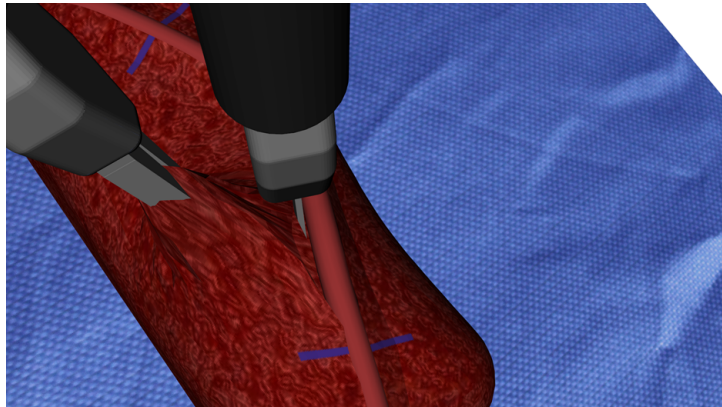
In the experiments, three active constraint assistance conditions were considered. These are shown below.

- **UC** - Unconstrained, no constraint forces or torques were applied.
- **DFC3T** - 3-D translational dynamic frictional constraints. As in section 5.2.3, with no constraint torques and $\theta = 30$ degrees.
- **DFC6C** - 6-D coupled dynamic frictional constraints with coupled frictional redirection in translation and rotation. As described in section 5.3 with $\theta = 30$ degrees and $\psi = 0.9$.

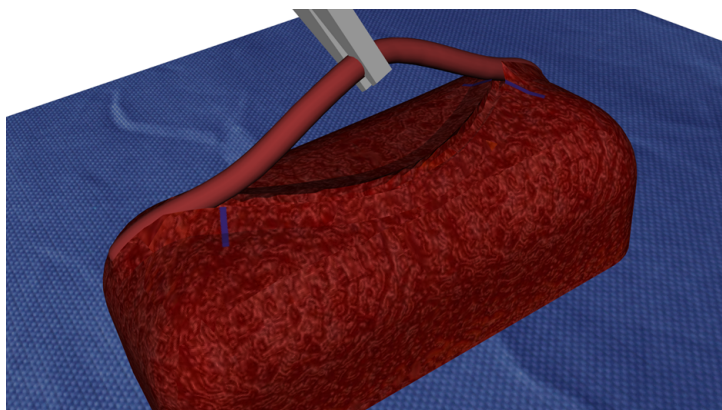
A total of 10 test subjects took part in the experiments, all of whom were right-handed, non-surgeons, aged between 21 and 32. This consisted of eight males and two females. Each test subject underwent a compulsory training session of 10 minutes with each of the three assistance methods, after which they were permitted to continue training with an equal time spent on each assistance method as they felt necessary. Once each user was comfortable with the controls for the system and understood the



(a) Surface dissection to separate the top half of the nerve from the tissue block. The forceps were used to tension the tissue block such that the cut and uncut sections of the interface could be identified.



(b) Subsurface dissection to separate the bottom half of the nerve from the tissue block. The forceps were used to manoeuvre the tissue block so that the user had access to the subsurface interface and was able to see and cut along it.



(c) A completed dissection of the nerve between the two blue markers.

Figure 7.5: The dissection process used by the test subjects during the experimental validation.

techniques required to perform the dissection, they were asked to dissect the nerve twice using each of the assistance methods. The ordering of the assistance methods for the six dissections performed by each test subject was randomised to minimise any training bias. The users were informed of which assistance method would be in operation for each test so that they could work cooperatively with the assistance from the beginning of the dissection, rather than having to work out which method was in use for themselves. Each individual trial, after training, lasted 5 to 6 minutes on average.

7.5 Results

Four primary metrics were used within the experiments to investigate the efficacy of the different constraint methods. These metrics were: mean penetration depth; mean penetration area; contact displacement and dissection time. As in chapter 5, the haptic power was recorded as a secondary metric so that the energetic properties of the constraint methods could be validated. The results for each of the metrics are summarised in tables 7.4 and 7.5.

The mean penetration depth represents the mean of the maximum depths by which the scalpel penetrated the nerve during the dissection. To avoid biasing the results when the user was reorienting the tools and camera, only times when the scalpel was within 1 mm of the nerve were considered for the averaging. The distributions for the mean penetration depth are shown in figure 7.6a.

The mean penetration area represents the mean of the approximate scalpel area intersecting the nerve during the dissection. As before, to avoid biasing the results when the user was reorienting the tools and camera, only times when the scalpel was within 1 mm of the nerve were considered for the averaging. The distributions for the mean penetration area are shown in figure 7.6b.

The contact displacement represents the total displacement of the deepest penetrating part of the scalpel while it was intersecting the nerve. The distributions for the contact displacement are shown in figure 7.6c.

The task time represents the total time for the user to dissect the nerve from the first cut to the last cut. The distributions for the task time are shown in figure 7.6d.

The mean translational power represents the mean of the power transferred from the right-hand haptic device to the user due to translational forces. The distributions for the mean translational power are shown in figure 7.7a.

Constraint method	Mean penetration depth (μm)	Mean penetration area (mm^2)	Contact displacement (mm)	Dissection time (s)
UC	32.8 ± 36.4	140.0 ± 168.9	14.3 ± 12.9	324.0 ± 154.2
DFC3T	4.5 ± 7.3	19.3 ± 32.6	2.0 ± 3.0	401.6 ± 111.5
DFC6C	2.5 ± 5.5	13.1 ± 32.1	1.2 ± 2.1	467.2 ± 208.0

Table 7.4: Summary of the results for the three constraint methods across the primary metrics. The unconstrained test cases are denoted by UC, the translational dynamic frictional constraint test cases are denoted by DFC3T and the full translational and rotational dynamic frictional constraint test cases are denoted by DFC6C. Each value shows the mean and standard deviation from a total of 20 tests across 10 test subjects.

Constraint method	Mean translational power (mW)	Mean rotational power (mW)	Mean total power (mW)
UC	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
DFC3T	-1.28 ± 0.57	0.0 ± 0.0	-1.28 ± 0.57
DFC6C	-1.34 ± 0.57	-2.57 ± 0.95	-3.91 ± 1.40

Table 7.5: Summary of the results for the three constraint methods across the secondary metrics. The unconstrained test cases are denoted by UC, the translational dynamic frictional constraint test cases are denoted by DFC3T and the full translational and rotational dynamic frictional constraint test cases are denoted by DFC6C. Each value shows the mean and standard deviation from a total of 20 tests across 10 test subjects.

Constraint method pairing	Mean penetration depth	Mean penetration area	Contact displacement	Dissection time
UC vs DFC3T	** / ●●	** / ●●	** / ●●	
UC vs DFC6C	** / ●●	** / ●●	** / ●●	**
DFC3T vs DFC6C	●●	●●	●●	

Table 7.6: Results of pairwise statistical significance tests for means and variances at the 95 % confidence interval. The symbol ‘**’ identifies a constraint method pairing, and performance metric, for which significantly different means were identified. The unconstrained test cases are denoted by UC, the translational dynamic frictional constraint test cases are denoted by DFC3T and the full translational and rotational dynamic frictional constraint test cases are denoted by DFC6C. The symbol ‘●●’ identifies a constraint method pairing, and performance metric, for which significantly different variances were identified.

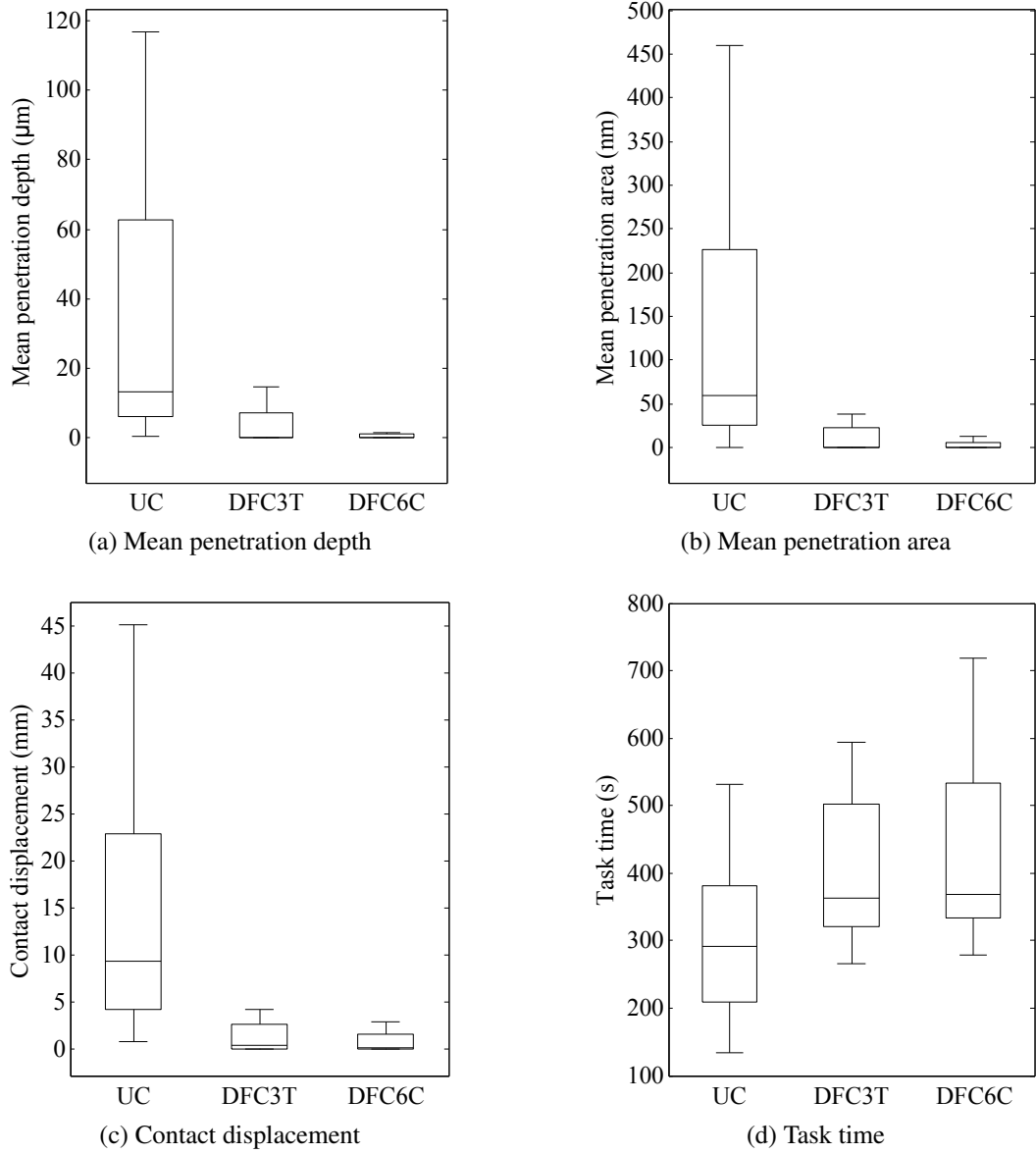


Figure 7.6: Experimental results of the four primary performance metrics across the three constraint methods. The unconstrained test cases are denoted by UC, the translational dynamic frictional constraint test cases are denoted by DFC3T and the full translational and rotational dynamic frictional constraint test cases are denoted by DFC6C. Each box shows the distribution from 20 tests across 10 test subjects. The outliers are not shown for clarity.

The mean rotational power represents the mean of the power transferred from the right-hand haptic device to the user due to rotational torques. The distributions for the mean rotational power are shown in figure 7.7b.

The mean total power represents the mean of the power transferred from the right-hand haptic device to the user due to both translational forces and rotational torques. The distributions for the mean total power are shown in figure 7.7c.

As each of the primary metrics was found to be non-normally distributed by the one-sample Kolmogorov-Smirnov test, the non-parametric Kruskal-Wallis one-way ANOVA test was applied. All four of the primary metrics were found to be significantly affected by the assistance condition ($p < 0.05$). Pairwise comparisons were then made between the means of the three assistance condition pairings using the Dunn-Sidak multiple comparison test, for each primary metric, at the 95 % confidence interval. Additionally, pairwise comparisons were made between the variances of the three assistance condition pairings using the Brown-Forsythe test, for each primary metric, also at the 95 % confidence interval. The results of the Dunn-Sidak and Brown-Forsythe pairwise comparisons are shown in table 7.6.

7.6 Discussion

The experimental validation of the two dynamic frictional constraint formulations conclusively show that they are of significant benefit, in terms of preventing damage, during a simulated nerve dissection task.

The three metrics used to quantify the efficacy of constraints at preventing nerve damage (mean penetration depth, mean penetration area and contact displacement) show similar trends across the three assistance conditions. The mean penetration depth identifies how deep the user was cutting into the nerve, on average, during a dissection. It is hypothesised that, the deeper a surgeon penetrates a nerve with a surgical instrument, the more fibres they damage and the more significant the effect will be upon the signal transmission. The mean penetration area shows how much of the scalpel blade penetrates the nerve. The key distinction between penetration depth and area is that the area can identify when the user has the scalpel blade parallel to the nerve but in penetration. In this case, the depth of penetration can be small but the whole face of the scalpel can be pressing on the nerve causing damage. The contact displacement quantifies how far the user moves with the scalpel embedded within the nerve. If the

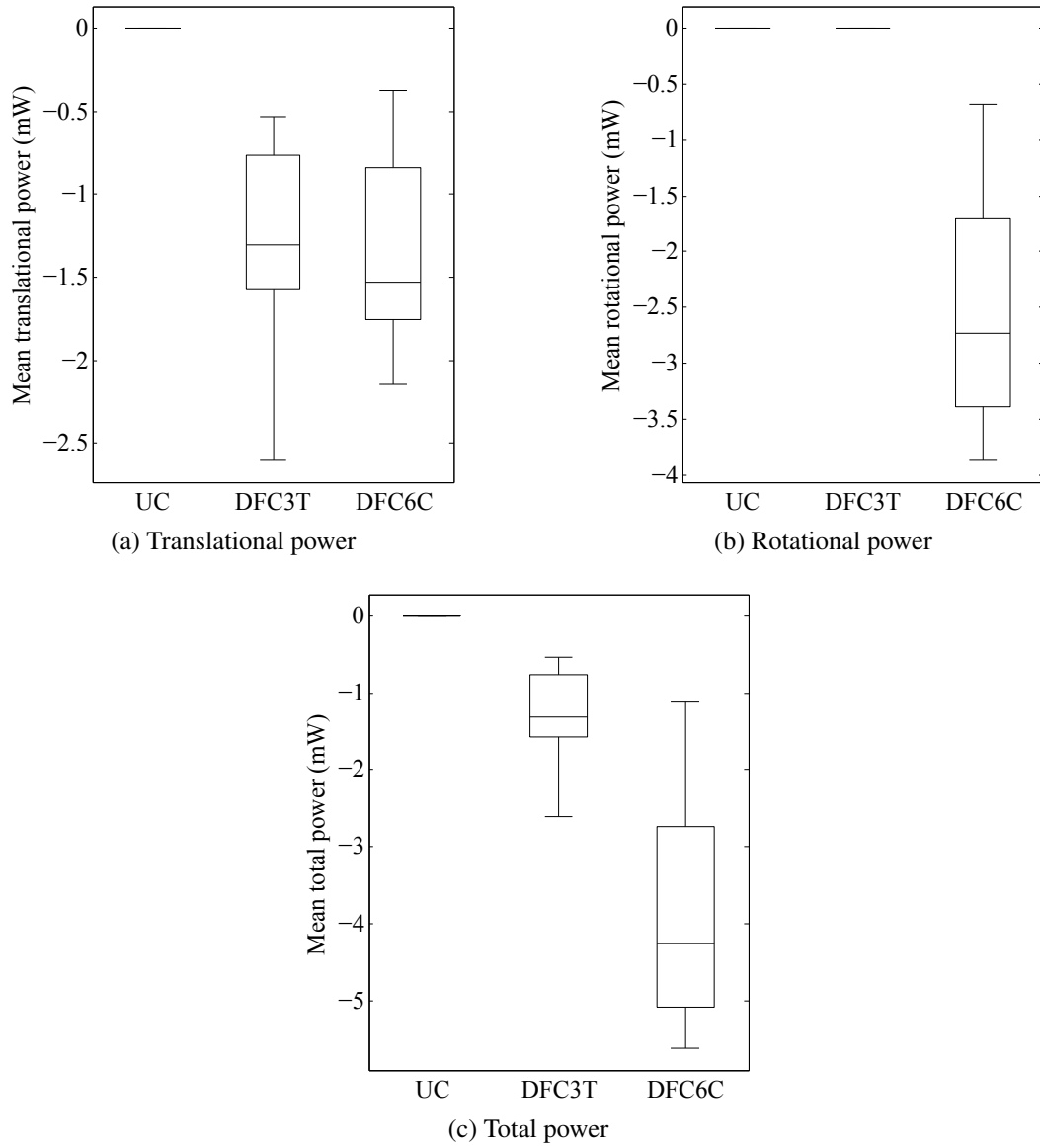


Figure 7.7: Experimental results of the three secondary performance metrics across the three constraint methods. The unconstrained test cases are denoted by UC, the translational dynamic frictional constraint test cases are denoted by DFC3T and the full translational and rotational dynamic frictional constraint test cases are denoted by DFC6C. Each box shows the distribution from 20 tests across 10 test subjects. The outliers are not shown for clarity.

user penetrates the nerve, but stops before retracting, it is better than if the user penetrates then moves sideways before retracting, as this will likely damage fewer nerve fibres, and this metric captures this.

In all three of the nerve damage metrics, the unconstrained assistance condition (UC) had the worst performance, with the highest penetrations and contact displacement values. These values show that the task was sufficiently difficult that the non-surgical users could not achieve it without damaging the nerve to some degree. Even though the motion of the nerve was entirely controlled by the users themselves, visualising the subsurface dissection planes and having the dexterity to follow their intended motion pathway were a challenge for all users.

The three degrees of freedom, translational, dynamic frictional constraint (DFC3T) had the second best performance across all three nerve damage metrics, with mean values significantly improved upon UC. The mean and median values for DFC3T were near to zero in all three metrics. These results show that even a comparatively simple, translational dynamic frictional constraint can generally prevent almost all constraint penetration in the simulated nerve dissection task. The performance of DFC3T in these experiments was unexpectedly strong when compared to its performance in chapter 5. After some investigation, the discrepancy between the efficacy when not constraining orientation in this surgical task and that shown in the previous chapter was found to be due to the tubular nature of the constraint geometry. For any orientation in which the user has the tool, there is a configuration on the surface of the constraint which has that orientation and which does not violate the constraint. By prioritising the orientation in the proxy's mass matrix, this non-penetrating proxy position is computed by the constraint controller. The translational separation between the orientation prioritised proxy and actual tool may be larger than if the proxy were not orientation prioritised, but nevertheless, translational movement alone was shown to be almost always enough to extract the tool from the constrained region. Conversely, in the previous task, there were many orientations in which there were no valid positions for the ring on the pathway, meaning that constraining orientation was of greater importance.

Though the overall performance was good, there were cases when the users were able to circumvent the DFC3T assistance method achieving relatively large penetrations or displacements into the nerve. These cases typify the primary limitation of a three dimensional dynamic frictional constraint, which is that a penetration caused by rotation cannot be prevented unless the user is simultaneously translating the tool. Even with the effective dynamic proxy formulation, if the user rotates the tool in an incorrect direction, such that the scalpel penetrates the nerve, then the proxy will separate from the scalpel and move into a non-penetrating pose. However, if the user is not translating, the frictional redirection cannot generate

any constraint forces to guide the user towards the proxy pose and out of the constraint. The solution to this problem is to implement the six degrees of freedom, coupled dynamic frictional constraint (DFC6C). In fact, in the experimental results, DFC6C is shown to have the best mean performance across all three nerve damage metrics. The mean performance of DFC6C was found to be significantly improved upon UC; however, it was not found to be significantly improved from DFC3T. As the means for both DFC3T and DFC6C are close to zero, finding a significant difference between them would require large sample sizes which would have involved a prohibitive amount of time to produce. The Brown-Forsythe tests, however, proved that the variances for DFC6C were significantly less than the variances for DFC3T. This reduced variance indicates that, by incorporating the coupled constraint upon orientation, there will be fewer occurrences when the user can overcome the constraint and damage the nerve, which in a clinical context would represent a significant improvement in itself.

From observing the test subjects during the experiments, two primary penetration modes were identified. The first of these involved deep penetrations which occurred when the user was dissecting the subsurface interface and they had to rotate the scalpel to access the underside of the nerve. Some subjects would misjudge the tissue configuration under the nerve and rotate the scalpel in such a way that it penetrated the nerve deeply. This form of nerve damage is most suited to prevention by DFC6C as it can resist the user making the incorrect rotation. This feature of DFC6C is illustrated by the results for the mean penetration depth, in which the difference in variance between DFC3T and DFC6C is greatest. The second penetration mode occurred when the user did not control the forceps precisely during the dissection and they pulled the nerve onto the scalpel causing shallow, but sustained, penetrations. Though the aim of all dynamic frictional constraints is to respond to changes in the constraint environment, the frictional redirection cannot respond instantaneously to changing environments as it needs to receive enough energy from the user to enact the redirection. This means that all dynamic frictional constraints, including DFC6C, have a limited response to this second penetration method. This is illustrated by the smaller, but still significant, improvement in variance between DFC3T and DFC6C for the contact displacement.

The results for the dissection task time show that users under assistance method UC were faster than under the two constrained assistance methods, significantly so for DFC6C. This metric could show that the users were more comfortable without any constraint assistance; however, it is more likely that the difference in task time is due to the way in which the unconstrained users penetrated the nerve much more often and deeply so that they inadvertently dissected the underside of the nerve with far less effort. The mean task times for DFC3T and DFC6C were generally comparable and the increase in peak task

times for DFC6C is also likely to be due to the way in which it prevents inadvertent dissection of the nerve's underside.

As expected, the total power results for both constraint methods were found to be negative in all cases. Unlike in chapter 5, the rotational power for DFC6C was also negative. Whilst the method still relies on frictional redirection between translation and rotation, this result shows that the transfer was less one-sided in these experiments. In other words, rather than lots of energy being transferred from translation into rotation and very little energy being transferred in the other direction, in these experiments the results seem to show that there was a similar amount of energy being transferred in each direction. This balance is indicative of the way in which the proxy was defined to share the burden of a constraint penetration equally between a translational error and a rotational error.

Though the above results substantiate the abstract experiments from the previous chapters, there are still some limitations within the method and experimental validation. The most significant limitation of the experiments was that the dissections were performed by non-surgical personnel. Though the task was clinically inspired and the users were given clear instructions on how to perform the dissection, they were not surgeons and therefore their performance is not known to be comparable. Surgeons were not asked to perform the experiments because of the high number of experiments required to reach statistical significance and the realism limitations of the physical simulation, which it was believed would be too distracting and dissimilar from real surgery for the surgeons. If the simulation could be sufficiently improved, in the ways detailed below, then it would be possible to perform a small trial with surgeons; however, the statistical power of the results would remain limited. Additionally, as stated previously, the scalpel based dissection is not clinically accurate. Therefore, to make the simulation more pertinent to a surgeon, it would be beneficial to implement the constraints on the spreading action used with the dissector tool in place of the scalpel.

There are several aspects of the physical soft tissue simulation which could be improved so that the realism of the simulation would increase and the overall experiment would bear a greater resemblance to a real dissection. Most of the potential improvements to the simulation would require increased computational power. Many of the SOFA components used for the physical simulation are graphics processing unit (GPU) compatible and could be sped up by utilising a GPU; however, not all of them are currently supported, and implementing a mixed GPU/non-GPU system would be excessively complicated. By developing the open-source SOFA code to ensure all components could be GPU accelerated, the overall simulation rate could be increased. Simply increasing the simulation rate would have significant benefits

for the realism of the simulation because it would allow for the stable use of more realistic mechanical tissue properties and would make the contacts with the surgical instruments more stable, and less prone to pop-through, as can occur when the slow update rate misses surface contacts. Another significant improvement which could be made to the physical soft tissue simulation would be to implement free form cutting [Courtecuisse et al., 2010], rather than restricting the cutting to along the nerve-tissue interface. In addition to making the dissection feel more natural for the user, if the user had greater freedom over the pathway of the dissection, metrics relating to positive and negative surgical margins could be taken into consideration.

The significant time demands of the experiment prevented it here; however, it would be of interest to perform the dissection experiments with the same range of assistance methods as was used in chapter 5 and the viscoelastic constraint assistance method used in chapter 4. The two constraint methods were selected for these experiments as they allowed the investigation of the importance of two of the key research contributions of this thesis, namely applying constraints to the nerve dissection task and also constraining orientation. With more time, it would be useful to quantify the performance of independent aspects of the constraint such as was achieved in chapter 5 with the rotation only, uncoupled and conventional frictional constraints. Additionally, quantifying the level of performance with a rigid viscoelastic constraint would provide a useful ‘gold standard’ against which the performance of the friction based constraints could be measured.

7.7 Summary

In this chapter, it has been shown how the dynamic active constraint elements conceived in chapters 3, 4 and 5 can be assembled into a complete active constraint controller which can assist a human user in a surgically based task. In order to validate the active constraint controller, a complete surgical simulation was assembled with a physical soft tissue simulation, 3-D visualisation and two master haptic devices. The full dynamic active constraint system was used to perform the simulated dissection of a nerve from a soft tissue block, inspired by a real surgical procedure in which nerve damage is a considerable problem for surgeons.

Experimental validation of dynamic frictional constraints in the simulated nerve dissection task showed that they were of significant benefit for 10 non-surgical users. The results showed that, with active constraint assistance, the metrics identifying damage to the nerve could be reduced to almost zero for the

majority of users. The results also showed that including coupled constraints upon orientation could significantly reduce the number of occurrences where the user circumvented the translation only constraint and damaged the nerve.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

To date, robot assisted surgery has been the field in which the advantageous attributes of robotic devices have been integrated within surgery most successfully. Active constraints are collaborative human-robot control algorithms which have a well established history of use within robot assisted surgery. Active constraints anisotropically regulate the motion of a human user in such a way that it combines the competencies of surgeons and robots, allowing for improved clinical outcomes and surgeon experience. The significant majority of research previously presented for active constraints focusses on their application to static procedures, where the surgical environment is assumed to be rigid throughout. Though static constraints can be applied to certain clinical tasks, such as arthroplasty, the majority of practical surgical procedures involve deforming soft tissue structures.

Within deforming soft tissue environments, conventional static active constraints must be replaced with dynamic active constraints, where the geometries used to define the constraints change and deform in conformance with the patient's anatomy. Introducing dynamism to an active constraint generates several research challenges which have not previously been addressed within the literature. Within this thesis, two of the major obstructions to applying dynamic active constraints have been investigated. It was initially hypothesised that, by overcoming these challenges, dynamic active constraints could become applicable to a wider range of unstructured, practical surgical tasks, allowing surgeons to benefit from robot assistance in a greater range of procedures, eventually improving outcomes for patients.

In addition to the publication of a comprehensive survey which summarised the presently available literature, by investigating the hypothesis stated, several novel research contributions have been generated which extend the state of the art in the field of active constraints.

The most significant contribution of this thesis has been in the conception of a dynamic active constraint formulation in chapters 4 and 5 which ensures that overall control of a surgical procedure remains with the surgeon, while providing effective assistance in dynamic tasks. The constraint formulation, termed a dynamic frictional constraint, is based on a model for friction and therefore never generates or stores energy while the user interacts with a constraint, that is to say it is dissipative. The dissipative energetic properties of dynamic frictional constraints mean that all of the constraint forces applied to guide the surgeon are the result of energy which is simultaneously being put into the system by the surgeon. This means that a surgeon using a dynamic frictional constraint is never forced to move the surgical instruments by the constraint and therefore retains absolute control of the procedure.

The concept of a non-energy storing constraint formulation was not in itself novel. Instead, the primary novelty came from the way in which a dynamic frictional constraint redirects the surgeon's energy between all six dimensions of $SE(3)$ in order to provide the most effective guidance. By applying a simple friction based constraint which diametrically opposes motion, it is possible resist a surgeon who is directly moving in a direction not permitted; however, once penetration has occurred, the constraint cannot cause the surgeon to retract. By applying the frictional resistance in an oblique angle, it is possible to harvest the surgeon's energy in different dimensions and reapply it in the direction required to best encourage retraction from a constraint. Considering the friction simultaneously in both translation and rotation makes this even more effective. The importance of this concept for the wider research field is that it allows for a much richer haptic interaction with a constraint than would be possible using conventional friction. For active constraints, the experimental results shown in chapters 4, 7 and particularly 5, prove that this increased haptic information makes the constraint more effective at communicating with the users and guiding them in the way required. Specifically, the experimental validation within this thesis shows that, in terms of minimising erroneous movements from the users, dynamic frictional constraints are a significant benefit to users in several clinical and non-clinical tasks. These results in simulation imply that the benefits of increased user accuracy could be translated into practical surgical tasks whereby they could improve clinical outcomes for patients.

Another major contribution presented in this thesis is the development of a hierarchical search structure which allows for the use of more detailed constraint representations. The deformation invariant

bounding sphere hierarchy proposed in chapter 3 exploits the way in which soft tissue structures generally only stretch by a limited amount, allowing the efficient computation of the relative configuration between a constraint surface and a surgical instrument. Using the limited tissue stretch assumption, the hierarchy is constructed in such a way that it does not need rebuilding or refitting when the soft tissue deforms. Within conventional hierarchical search structures, performing rebuilds or refits after deformation is computationally expensive, therefore precluding this can save significant computation time when the tissue deforms regularly. Indeed, the experimental validation of the new hierarchy, in comparison with conventional methods from the literature, showed that, when the constraint deformed often, it was significantly faster across a range of mesh sizes, with an increasing benefit as the meshes became larger. As there is a rigid requirement for the rate at which haptic controllers, such as those used in active constraints, must operate, the computational demands of conventional search hierarchies has historically limited the resolution of the constraint surface which can be used. As the increased efficiency of the proposed method allows for higher resolution constraint representations, it means that the underlying patient's anatomy can be tracked more accurately and therefore the constraints can be more effective at assisting the surgeon.

To apply regional dynamic active constraints based on surface geometries effectively, a secondary contribution of this thesis, shown in chapter 6, was to extend the constraint-based method for computing rigid body collision responses to eliminate the need for continuous collision detection. To apply a regional dynamic active constraint it is necessary to define a target pose for the tool that does not penetrate the constraint, to which the user should be guided. Within the literature, a well established method for achieving this is with the use of a proxy. A proxy is a dynamically simulated object which follows the surgical instruments, but is unable to penetrate any of the constrained regions. In the literature, simulating the non-penetrating collision response of a proxy with a constraint requires continuous collision detection to identify the exact configuration at first contact. By adapting this method to operate with discrete range queries using deformation invariant bounding sphere hierarchies, rather than continuous collision detection, the computational requirement of the process could be reduced. As above, reducing the computational demands of dynamic active constraints in this way meant that more detailed constraint surfaces could be used to guide the users. In the experimental validation in chapter 7, the effective way in which the constraint-based proxy method defined the target pose meant that the constraints were able to stably guide the user while performing a simulated surgical task, particularly demonstrated by the strong performance of the translation only constraint.

An additional secondary contribution of this thesis was the proposal and initial validation, in chapter 7, of applying dynamic active constraints to preserve nerves during their dissection from soft tissue structures. Nerve damage presents a significant challenge for surgeons in a range of surgical procedures. The effects of surgical instruments cutting and crushing nerves can be severe for a patient and therefore applying robotic assistance in order to prevent this from happening inadvertently is beneficial. In soft tissue dissection tasks, a protective field around delicate structures, such as nerves, could be used as a cutting guide for the surgeon to ensure suitable surgical margins and prevent damage. This surgical scenario is well suited to the dynamic frictional constraints proposed in this thesis because the frictional redirection leads to smooth surface interactions which will not overpower the surgeon when the tissue deforms. In the experimental validation of dynamic frictional constraints for simulated nerve dissection it was found that they could be of significant benefit to non-surgical users for minimising multiple metrics associated with nerve damage.

Overall, the research presented in this thesis has addressed several of the major challenges associated with the application of active constraints to soft tissue surgery. There are two primary implications of these research contributions to the state of the art in active constraints: extending applicability and increasing performance. The creation and development of the dynamic frictional constraint formulation, and the concept of applying dynamic active constraints to nerves, both extend the range of surgical applications to which dynamic active constraints can be applied. In other words, they permit conventional active constraint benefits such as increased accuracy, precision and surgeon ergonomics in procedures where they would not otherwise be available. The proposal of a deformation invariant bounding sphere hierarchy, and the extension of the dynamic proxy simulation, both increase the performance of active constraints within existing applications. Both of these developments reduce the computational demand of important components of dynamic active constraint systems and, consequently, increasingly detailed representations can be made of the patient's anatomy, improving effectiveness. The expected implication of both extending the applicability, and increasing the performance, of dynamic active constraints is that surgical performance can be improved across a range of procedures, reducing the risk to the patient of complications and increasing the probabilities of surgical success.

In addition to these contributions to active constraint research, the implications of this work are believed to extend to several, wider, research fields. Though the issues of reduced human control during human-robot collaboration may be less crucial outside of surgery, there are many other areas in which robotics is used in an environment which is not known with absolute certainty, and sudden forcing of robotic devices

could lead to damage to equipment or harm to users. For example, research has been carried out for using active constraints to assist in cutting tape on an orbiting satellite while avoiding the underlying insulation [Xia et al., 2012]. In tasks such as this, an error in the environmental model could cause a constraint to jerk the robot, potentially damaging itself or the satellite. The dynamic frictional constraint formulation would be of use here. In addition to being relevant to non-surgical dynamic active constraints, the deformation invariant bounding sphere hierarchy and proxy motion formulation could both be used effectively in many other haptic applications, such as in the simulation of virtual environments. The more detail which can be included in a virtual environment to be rendered haptically, the more realistic and immersive the experience will be for the user.

8.2 Limitations and Future Work

Throughout the individual chapters of this thesis, limitations and future directions of the individual research contributions have been identified. In addition to these specific areas, there are several high-level areas in the research presented, and the wider field, which are undeveloped and could be exploited to increase the utility and performance of active constraints.

The largest limitation of the research presented, in terms of clinical conclusions, is that almost all of the experiments were performed by non-surgeons. It has been shown that surgeons have above average hand-eye coordination and dexterity [Francis et al., 2001], therefore it is reasonable to expect that their performance within the type of tasks considered here would also be above average. The results of the experiments in chapters 4, 5 and 7 show that applying constraints increased the highest values of most performance metrics, suggesting that they were effective at assisting users of all ability levels; however, this is not proven explicitly. The most incisive way of adding clinical impact to the presented results would be to re-run the nerve dissection experiments in chapter 7, once the modifications suggested to improve the realism of the simulation have been implemented. Due to the expected increase in user ability, the performance benefit from constraints would be likely to decrease; however, the trends within a variety of tasks in the wider literature on active constraints support the expectation that they would still be beneficial.

It has been stated within this thesis that the advantage gained by the improved computational efficiency of the deformation invariant bounding sphere hierarchy and the proxy formulation is that they permit finer representations of the patient's anatomy and therefore more accurate constraint enforcement. The

final, system wide, experiments in chapter 7 do not thoroughly demonstrate the benefit of this attribute. Though the nerve is represented as a reasonably high resolution mesh, its surface is not large enough, and does not contain enough features, to necessitate the level of mesh detail at which the computational optimisation becomes significant. To really see the effects of the computational efficiency, a far more detailed simulation environment is required in which the constraints protect multiple large tissue structures and where the working spaces are more restricted. If a laparoscopic simulation of a procedure, such as the robotic radical prostatectomy, were created, and constraints were applied to all surrounding soft tissue structures which should not be cut during the procedure, then the mesh resolution would be expected to have a noticeable impact. In such an intricate environment, the constraint meshes would need to be large and the cavernous nature of the anatomy would mean that the geometries would need to be convoluted. If the resolution of the constraint mesh was restricted, then some of the detail in the surface undulations would be lost and this could lead to underconstraining the delicate tissue structures, potentially permitting damage, or overconstraining the free space, potentially occluding the working environment and preventing the surgeon from manoeuvring effectively. In the context of surgical oncology, the benefits of fine constraint representations are particularly salient and would be a good way to validate them. Here, fine constraint representations could allow for slim negative surgical margins, avoiding unnecessary damage to healthy tissue and ensuring complete tumour removal.

All of the experiments within this thesis were performed using simulated environments. This is a useful way to perform precise, initial validations; however, to be of practical use, dynamic active constraints must be applied to physical robots in real surgical environments. Before any final conclusions can be made regarding the clinical importance of the dynamic active constraint components presented, they must be taken out of simulation. All of the research contributions are applicable to both teleoperated and hands-on robotic systems; however, the dynamic frictional constraint formulation relies on a delicate, low energy, haptic interaction. If these constraints were used with a hands-on robot with high mass and inertia and significant energy losses due to effects such as friction, their effectiveness would very likely decrease. For this reason, the first stage for validating these constraints in a physical system would be to implement them in a teleoperated system such, as the RAVEN II [Hannaford et al., 2013], with six degree of freedom master devices, such as the sigma.7 (Force Dimension Inc.). Conceptually, the research in this thesis could be transplanted directly into a conventional teleoperation system in which the instruments on the slave robots are controlled to follow the poses of the master devices [Ryu et al., 2004]; however, some aspects would require detailed consideration. One decision which must be made

is with regards to whether the constraints are applied based on the master or slave poses. As investigated in [Abbott and Okamura, 2003a], the selection can affect the magnitude of constraint penetration and the coordination of master and slave devices, therefore a compromise between these factors is likely. Another challenge with active constraints in teleoperation systems pertains to whether, and how, haptic feedback of tool interaction forces at the slave robot are presented to the surgeon. Directly mixing forces from both tool interactions and active constraints could be confusing for the surgeon, therefore research has already been carried out into methods for distinguishing these two sources [Peon et al., 2013]. Due to the energy based approach taken in dynamic frictional constraints, a consideration would have to be made regarding the effects of modulating or vibrating these forces in a disconnected process. Even with an implementation on a physical teleoperated system, this does continue to leave unresolved the challenge of interpreting the physical environment in which the slave robot operates.

Within the research presented in this thesis, one fundamental component of all practical active constraint systems has been neglected. Constraint generation, the method by which a patient's deforming anatomy is interpreted by the system and used to define the geometry of the constraint, has been avoided. There are some limited approaches within the literature to this challenge; however, computationally identifying soft tissue structures with poor boundary contrast and occlusions caused by other tissue and surgical instruments makes this a considerable challenge. One particularly promising method for constraint generation in certain soft tissue structures, including nerves, is with the use of near infra-red (NIR) fluorescence imaging. NIR fluorescence imaging is a widely researched technique that allows targeted soft tissue structures to be marked with a dye which fluoresces at a wavelength for optimal soft tissue penetration [Frangioni, 2003; Troyan et al., 2009]. The penetration properties of the fluorescent wavelengths mean that the marked tissue can be accurately visualised, even when obscured by up to 10 mm of other soft tissue. It is possible that a tissue structure requiring protection or dissection could be marked in such a way and an imaging system could be used to generate dynamic active constraints which assist with the surgical procedure, much like the formulation shown in chapter 7. Scattering and absorption of the fluorescent signal, and occlusion of surgical tools, are likely to be challenges within this approach; however, a probabilistic model of the tissue of interest could be used to generate an estimate of the anatomy.

It is believed that integrating dynamic active constraints with the biological marking and real-time imaging of specific parts of a patient's anatomy poses the greatest potential for future research on active constraints in soft tissue surgery. Biologists are currently developing a wide range of inorganic, organic and biological nanoparticles which can be used to target and mark many structures [Rao et al., 2007]. It is

possible that a future active constraint system could be developed, around a haptically enabled master-slave surgical robot, where the patient is pre-operatively injected with a mixture of markers, each of which has a wavelength tuned to define a certain type of constraint and bind to a certain tissue structure. In this system the robot will simply see which structures the surgeon can, and cannot, cut into and build precisely defined constraints automatically to provide efficient and effective assistance.

Bibliography

- Active Constraint Technologies for Ill-defined or Volatile Environments (ACTIVE), 2012. URL <http://www.active-fp7.eu/>.
- D. Aarno, S. Ekvall, and D. Kragic. Adaptive virtual fixtures for machine-assisted teleoperation tasks. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 897–903, Apr. 2005. doi: 10.1109/ROBOT.2005.1570231.
- J. J. Abbott and A. M. Okamura. Virtual fixture architectures for telemanipulation. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 2, pages 2798–2805, Sep. 2003a. doi: 10.1109/ROBOT.2003.1242016.
- J. J. Abbott and A. M. Okamura. Analysis of virtual fixture contact stability for telemanipulation. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 2699–2706, Oct. 2003b.
- J. J. Abbott and A. M. Okamura. Pseudo-admittance bilateral telemanipulation with guidance virtual fixtures. In *Proc. Symp. Haptic Interf. Virt. Env. and Teleop. Sys.*, pages 169–175, Mar. 2006a. doi: 10.1109/HAPTIC.2006.1627057.
- J. J. Abbott and A. M. Okamura. Stable forbidden-region virtual fixtures for bilateral telemanipulation. *J. Dyn. Sys., Measur., and Cont.*, 128(1):53–64, 2006b. doi: 10.1115/1.2168163.
- J. J. Abbott, P. Marayong, and A. M. Okamura. Haptic virtual fixtures for robot-assisted manipulation. In *Robot. Res., Springer Tracts in Adv. Robot.*, volume 28, pages 49–64, 2007.
- R. J. Adams and B. Hannaford. Stable haptic interaction with virtual environments. *IEEE Trans. Robot. Autom.*, 15(3):465–474, June 1999. ISSN 1042-296X. doi: 10.1109/70.768179.
- P. Aigner and B. McCarragher. Human integration into robot control utilising potential fields. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 291–296, Apr. 1997. doi: 10.1109/ROBOT.1997.620053.
- J. Aleotti, S. Caselli, and M. Reggiani. Evaluation of virtual fixtures for a robot programming by demonstration interface. *IEEE Trans. Syst., Man, Cybern. A*, 35(4):536–545, July 2005. ISSN 1083-4427. doi: 10.1109/TSMCA.2005.850604.
- J. Allard, S. Cotin, F. Faure, P.-J. Bensoussan, F. Poyer, C. Duriez, H. Delingette, and L. Grisoni. SOFA - an open source framework for medical simulation. In *Medicine Meets Virtual Reality*, Long Beach, USA, Feb. 2007.
- M. Ammi and A. Ferreira. Robotic assisted micromanipulation system using virtual fixtures and metaphors. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 454–460, Apr. 2007. doi: 10.1109/ROBOT.2007.363828.

- J. Arata, H. Kozuka, H. Kim, N. Takesue, B. Vladimirov, M. Sakaguchi, J. Tokuda, N. Hata, K. Chinzei, and H. Fujimoto. Open core control software for surgical robots. *Int. J. Comp. Assist. Radiol. Surg.*, 5:211–220, 2010. ISSN 1861-6410.
- K. K. Badani, S. Kaul, and M. Menon. Evolution of robotic radical prostatectomy. *Cancer*, 110(9): 1951–1958, 2007. ISSN 1097-0142. doi: 10.1002/cncr.23027.
- D. Baraff. Analytical methods for dynamic simulation of non-penetrating rigid bodies. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, pages 223–232, 1989.
- J. Barbic and D. James. Six-dof haptic rendering of contact between geometrically complex reduced deformable models. *IEEE Trans. Haptics*, 1(1):39–52, Jan 2008. ISSN 1939-1412. doi: 10.1109/TOH.2008.1.
- R. A. Beasley and R. D. Howe. Increasing accuracy in image-guided robotic surgery through tip tracking and model-based flexion correction. *IEEE Trans. Robot.*, 25(2):292–302, Apr. 2009. ISSN 1552-3098. doi: 10.1109/TRO.2009.2014498.
- B. Becker, R. MacLachlan, L. Lobes, and C. Riviere. Vision-based retinal membrane peeling with a handheld robot. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1075–1080, May 2012. doi: 10.1109/ICRA.2012.6224844.
- B. Becker, R. MacLachlan, L. Lobes, G. Hager, and C. Riviere. Vision-based control of a handheld surgical micromanipulator with virtual fixtures. *IEEE Trans. Robot.*, 29(3):674–683, June 2013. ISSN 1552-3098. doi: 10.1109/TRO.2013.2239552.
- B. C. Becker, R. A. MacLachlan, G. D. Hager, and C. N. Riviere. Handheld micromanipulation with vision-based virtual fixtures. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 4127–4132, May 2011. doi: 10.1109/ICRA.2011.5980345.
- M. Benallegue, A. Escande, S. Miossec, and A. Kheddar. Fast C1 proximity queries using support mapping of sphere-torus-patches bounding volumes. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 483–488, May 2009. doi: 10.1109/ROBOT.2009.5152722.
- J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, Sep. 1975. ISSN 0001-0782. doi: 10.1145/361002.361007.
- A. Bettini, S. Lang, A. M. Okamura, and G. D. Hager. Vision assisted control for manipulation using virtual fixtures. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1171–1176, 2001.
- A. Bettini, P. Marayong, S. Lang, A. M. Okamura, and G. D. Hager. Vision-assisted control for manipulation using virtual fixtures. *IEEE Trans. Robot.*, 20(6):953–966, 2004.
- G. Borghesan, B. Willaert, and J. De Schutter. A constraint-based programming approach to physical human-robot interaction. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 3890–3896, May 2012. doi: 10.1109/ICRA.2012.6224943.
- S. A. Bowyer and F. Rodriguez y Baena. Dynamic frictional constraints for robot assisted surgery. In *Proc. IEEE World Haptics Conf.*, pages 319–324, 2013.
- S. A. Bowyer and F. Rodriguez y Baena. Dynamic frictional constraints in translation and rotation. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 2685–2692, 2014.
- S. A. Bowyer, B. L. Davies, and F. Rodriguez y Baena. Active constraints/virtual fixtures: A survey. *IEEE Trans. Robot.*, 30(1):138–157, Feb. 2014.

- S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- A. Brander, P. Viikinkoski, J. Tuuhea, L. Voutilainen, and L. Kivisaari. Clinical versus ultrasound examination of the thyroid gland in common clinical practice. *Journal of Clinical Ultrasound*, 20(1): 37–42, 1992. ISSN 1097-0096. doi: 10.1002/jcu.1870200107.
- B. Brogliato, R. Lozano, B. Maschke, and O. Egeland. *Dissipative Systems Analysis and Control: Theory and Applications*. Springer-Verlag, 2007.
- J. Brown, S. Sorkin, C. Bruyns, J.-C. Latombe, K. Montgomery, and M. Stephanides. Real-time simulation of deformable objects: Tools and application. In *Proc. Comp. Ani.*, pages 228–258, Nov. 2001. doi: 10.1109/CA.2001.982397.
- H. Bruyninckx. Open robot control software: the orocos project. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 3, pages 2523–2528, 2001. doi: 10.1109/ROBOT.2001.933002.
- C. Burghart, J. Keitel, S. Hassfeld, U. Rembold, and H. Woern. Robot controlled osteotomy in cranio-facial surgery. In *Int. Workshop Haptic Devices in Med. Appl.*, Paris, France, June 1999.
- D. Burschka, J. J. Corso, M. Dewan, W. Lau, M. Li, H. Lin, P. Marayong, N. Ramey, G. D. Hager, B. Hoffman, D. Larkin, and C. Hasser. Navigating inner space: 3-d assistance for minimally invasive surgery. *Robotics and Autonomous Systems*, 52(1):5–26, 2005. ISSN 0921-8890. doi: 10.1016/j.robot.2005.03.013.
- C. Canudas de Wit, H. Olsson, K. Astrom, and P. Lischinsky. A new model for control of systems with friction. *IEEE Trans. Autom. Control*, 40(3):419–425, Mar. 1995. ISSN 0018-9286. doi: 10.1109/9.376053.
- R. Castillo-Cruces and J. Wahrburg. Virtual fixtures for autonomous error compensation for human-robot cooperative tasks. *Robotica*, 28:267–277, 2010.
- W. J. Catalona and S. W. Bigg. Nerve-sparing radical prostatectomy: evaluation of results after 250 patients. *The Journal of Urology*, 143(3):538–543, 1990.
- T. C. P. Chau, K.-W. Kwok, G. C. T. Chow, K. H. Tsoi, K.-H. Lee, Z. Tse, P. Y. K. Cheung, and W. Luk. Acceleration of real-time proximity query for dynamic active constraints. In *Proceedings of the International Conference on Field-Programmable Technology*, pages 206–213, 2013.
- N. Y. Chong, T. Kotoku, K. Ohba, and K. Tanie. Virtual repulsive force field guided coordination for multi-telerobot collaboration. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 1013–1018, 2001. doi: 10.1109/ROBOT.2001.932682.
- A. Chowriappa, R. Wirz, Y. Seo, A. Reddy, T. Kesavadas, P. Scott, K. Guru, and T. Kesavadas. A predictive model for haptic assistance in robot assisted trocar insertion. In *Proc. IEEE World Haptics Conf.*, pages 121–126, Apr. 2013. doi: 10.1109/WHC.2013.6548395.
- K. L. Clarkson. Nearest neighbour searching and metric space dimensions. In G. Shakhnarovich, T. Darrell, and P. Indyk, editors, *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice*, pages 15–59. MIT Press, Cambridge, MA, USA., 2005.
- J. D. Cohen, M. C. Lin, D. Manocha, and M. Ponamgi. I-collide: an interactive and exact collision detection system for large-scale environments. In *Proceedings of the symposium on Interactive 3D graphics*, pages 189–218, 1995. doi: 10.1145/199404.199437.

- J. Colgate, M. Peshkin, and S. Klostermeyer. Intelligent assist devices in industrial applications: a review. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, volume 3, pages 2516–2521, Oct. 2003. doi: 10.1109/IROS.2003.1249248.
- D. Constantinescu, S. Salcudean, and E. Croft. Haptic rendering of rigid body collisions. In *International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 2–8, Mar. 2004. doi: 10.1109/HAPTIC.2004.1287171.
- D. Constantinescu, S. E. Salcudean, and E. A. Croft. Haptic rendering of rigid contacts using impulsive and penalty forces. *IEEE Trans. Robot.*, 21(3):309–323, June 2005. ISSN 1552-3098. doi: 10.1109/TRO.2004.840906.
- H. Courtecuisse, H. Jung, J. Allard, C. Duriez, D. Y. Lee, and S. Cotin. GPU-based real-time soft tissue deformation with cutting and haptic feedback. *Progress in Biophysics and Molecular Biology*, 103(2): 159–168, 2010. ISSN 0079-6107. Special Issue on Biomechanical Modelling of Soft Tissue Motion.
- J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Pearson Education Inc., NJ, USA, 3rd edition, 2005. ISBN 0201095289.
- A.-M. Cretu, E. M. Petriu, and G. G. Patry. Neural network architecture for 3d object representation. In *Proc. IEEE Int. Workshop Haptic, Audio & Visual Env. & their Appl.*, pages 31 – 36, Sep. 2003. doi: 10.1109/HAVE.2003.1244721.
- R. L. Crumley. Unilateral recurrent laryngeal nerve paralysis. *Journal of Voice*, 8(1):79 – 83, 1994. ISSN 0892-1997. doi: 10.1016/S0892-1997(05)80323-6.
- P. Dahl. A solid friction model. Tech. Rep. TOR-0158(3107-18)-1, The Aerospace Corporation, 1968.
- E. B. Dam, M. Koch, and M. Lillholm. Quaternions, interpolation and animation. Technical Report DIKU-TR-98/5, Department of Computer Science, University of Copenhagen, 1998.
- G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.
- B. Davies, M. Jakopc, S. J. Harris, F. Rodriguez y Baena, A. Barrett, A. Evangelidis, P. Gomes, J. Henckel, and J. Cobb. Active-constraint robotics for surgery. *Proc. IEEE*, 94(9):1696–1704, 2006.
- B. L. Davies. A discussion of safety issues for medical robots. In R. H. Taylor, S. Lavalée, G. C. Burdea, and R. Mösges, editors, *Computer-Integrated Surgery - Technology and Clinical Applications*, chapter 19, pages 287–296. MIT Press, 1996.
- B. L. Davies, R. D. Hibberd, M. J. Coptcoat, and J. E. A. Wickham. A surgeon robot prostatectomy - a laboratory evaluation. *J. Med. Eng Tech.*, 13(6):273–277, 1989.
- B. L. Davies, R. D. Hibberd, W. S. Ng, A. G. Timoney, and J. E. A. Wickham. A surgeon robot for prostatectomies. In *Proc. IEEE Int. Conf. Adv. Robot.*, volume 1, pages 871–875, 1991.
- H. Davis and W. Book. Torque control of a redundantly actuated passive manipulator. In *Proc. American Control Conf.*, volume 2, pages 959–963, June 1997. doi: 10.1109/ACC.1997.609669.
- M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational geometry: algorithms and applications*. Springer-Verlag, 1997.
- C. Deck and R. Willinger. Improved head injury criteria based on head FE model. *International Journal of Crashworthiness*, 13(6):667–678, 2008.

- L. Di Gaspero. QuadProg++. URL <http://quadprog.sourceforge.net/>. [Online] Available: <http://quadprog.sourceforge.net/>.
- E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. ISSN 0029-599X. doi: 10.1007/BF01386390.
- Y. D. Dubbelman, G. R. Dohle, and F. H. Schroder. Sexual function before and after radical retropubic prostatectomy: A systematic review of prognostic indicators for a successful outcome. *European Urology*, 50(4):711–720, 2006. ISSN 0302-2838. doi: <http://dx.doi.org/10.1016/j.eururo.2006.06.009>.
- P. Dupont, V. Hayward, B. Armstrong, and F. Altpeter. Single state elastoplastic friction models. *IEEE Trans. Autom. Control*, 47(5):787–792, May 2002. ISSN 0018-9286. doi: 10.1109/TAC.2002.1000274.
- E. Dyllong and W. Luther. Verified convex hull and distance computation for octree-encoded objects. *Journal of Computational and Applied Mathematics*, 199(2):358–364, 2007. ISSN 0377-0427. doi: 10.1016/j.cam.2005.08.043.
- C. Ericson. *Real-Time Collision Detection*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. ISBN 1558607323.
- R. A. Finkel and J. L. Bentley. Quad trees: A data structure for retrieval on composite keys. *Acta Informatica*, 4(1):1–9, 1974.
- N. K. Francis, G. B. Hanna, A. B. Cresswell, F. J. Carter, and A. Cuschieri. The performance of master surgeons on standard aptitude testing. *American journal of surgery*, 182(1):30–33, July 2001.
- J. V. Frangioni. In vivo near-infrared fluorescence imaging. *Current Opinion in Chemical Biology*, 7(5): 626–634, 2003. ISSN 1367-5931. doi: <http://dx.doi.org/10.1016/j.cbpa.2003.08.007>.
- H. Fuchs, Z. M. Kedem, and B. F. Naylor. On visible surface generation by a priori tree structures. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, pages 124–133, 1980.
- J. Funda, R. H. Taylor, B. Eldridge, S. Gomory, and K. G. Gruben. Constrained cartesian motion control for teleoperated surgical robots. *IEEE Trans. Robot. Autom.*, 12(3):453–465, June 1996.
- Y. C. Fung. *Biomechanics: Mechanical Properties of Living Tissues*. Springer-Verlag New York, 2nd edition, 1993.
- E. Garcia, M. A. Jimenez, P. G. de Santos, and M. Armada. The evolution of robotics research. *IEEE Robot. Autom. Mag.*, 14(1):90–103, Mar. 2007. ISSN 1070-9932. doi: 10.1109/MRA.2007.339608.
- A. Ghanbari, H. Abdi, B. Horan, S. Nahavandi, X. Chen, and W. Wang. Haptic guidance for microrobotic intracellular injection. In *Proc. IEEE RAS EMBS Int. Conf. Biomed. Robot. Biomech.*, pages 162–167, Sep. 2010. doi: 10.1109/BIOROB.2010.5627016.
- T. L. Gibo, L. N. Verner, D. D. Yuh, and A. M. Okamura. Design considerations and human-machine performance of moving virtual fixtures. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 671–676, May 2009. doi: 10.1109/ROBOT.2009.5152648.
- E. Gilbert, D. Johnson, and S. Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Trans. Robot. Autom.*, 4(2):193–203, Apr. 1988. ISSN 0882-4967. doi: 10.1109/56.2083.

- R. B. Gillespie, J. E. Colgate, and M. A. Peshkin. A general framework for cobot control. *IEEE Trans. Robot. Autom.*, 17(4):391–401, Aug. 2001. ISSN 1042-296X. doi: 10.1109/70.954752.
- D. Goldfarb and A. Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27(1):1–33, 1983. ISSN 0025-5610. doi: 10.1007/BF02591962.
- S. Gottschalk, M. C. Lin, and D. Manocha. OBBTree: a hierarchical structure for rapid interference detection. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, pages 171–180. ACM, 1996. ISBN 0-89791-746-4. doi: 10.1145/237170.237244.
- G. S. Guthart and J. K. Salisbury, Jr. The Intuitive Telesurgery System: overview and application. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 618–621, 2000. doi: 10.1109/ROBOT.2000.844121.
- B. Hagag, R. Abovitz, H. Kang, B. Schmitz, and M. Conditt. RIO: Robotic-arm interactive orthopedic system MAKOpasty: User interactive haptic orthopedic robotics. In J. Rosen, B. Hannaford, and R. M. Satava, editors, *Surgical Robotics: Systems, Applications and Visions*, pages 219–246. Springer Science+Business Media, 2011. doi: 10.1007/978-1-4419-1126-1.
- G. Hager, A. Okamura, P. Kazanzides, L. Whitcomb, G. Fichtinger, and R. Taylor. Surgical and interventional robotics: part III. *IEEE Robot. Autom. Mag.*, 15(4):84–93, Dec. 2008. ISSN 1070-9932. doi: 10.1109/MRA.2008.930401.
- G. D. Hager. Human-Machine Cooperative Manipulation With Vision-Based Motion Constraints. In *Workshop Visual Servoing, IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2002.
- U. Hagn, R. Konietschke, A. Tobergte, M. Nickl, S. Jorg, B. Kubler, G. Passig, M. Groger, F. Frohlich, U. Seibold, L. Le-Tien, A. Albu-Schaffer, A. Nothhelfer, F. Hacker, M. Grebenstein, and G. Hirzinger. DLR MiroSurge: a versatile system for research in endoscopic telesurgery. *Int. J. of Computer Assisted Radiology and Surgery*, 5(2):183–193, 2010. ISSN 1861-6410. doi: 10.1007/s11548-009-0372-4.
- T. Haidegger, L. Kovacs, B. Benyo, and Z. Benyo. Spatial accuracy of surgical robots. In *International Symposium on Applied Computational Intelligence and Informatics*, pages 133–138, May 2009. doi: 10.1109/SACI.2009.5136227.
- B. Hannaford, J. Rosen, D. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. Kosari, and L. White. Raven-II: An open platform for surgical robotics research. *IEEE Trans. Biomed. Eng.*, 60(4):954–959, Apr. 2013. ISSN 0018-9294. doi: 10.1109/TBME.2012.2228858.
- R. I. Hartley and F. Kahl. Global optimization through rotation space search. *Int J. Comput Vis.*, 82: 64–79, 2009. doi: 10.1007/s11263-008-0186-9.
- V. Hayward, B. S. R. Armstrong, F. Altpeter, and P. E. Dupont. Discrete-time elasto-plastic friction estimation. *IEEE Trans. Control Syst. Technol.*, 17(3):688 –696, May 2009. ISSN 1063-6536. doi: 10.1109/TCST.2008.2001710.
- D. Hennekens, D. Constantinescu, and M. Steinbuch. Continuous impulsive force controller for forbidden-region virtual fixtures. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 2890–2895, May 2008.
- S. C. Ho, B. L. Davies, R. D. Hibberd, and J. Cobb. Robotic knee surgery - implicit force control strategy with active motion constraint. In *European Robot. and Intell. Sys. Conf.*, volume 3, pages 1235–1248, 1994.

- S. C. Ho, R. D. Hibberd, and B. L. Davies. Robot assisted knee surgery. *IEEE Eng. Med. Biol. Mag.*, 14 (3):292–300, May/June 1995.
- N. Hogan. Impedance control: An approach to manipulation: Part i—theory. *J. Dyn. Sys., Measure., and Cont.*, 107(1):1–7, 1985. doi: 10.1115/1.3140702.
- V. Horsley and R. H. Clarke. The structure and functions of the cerebellum examined by a new method. *Brain*, 31:45–124, 1908.
- P. Hubbard. Collision detection for interactive graphics applications. *IEEE Trans. Vis. Comput. Graphics*, 1(3):218–230, Sep. 1995. ISSN 1077-2626. doi: 10.1109/2945.466717.
- G. Hubens, H. Coveliers, L. Balliu, M. Ruppert, and W. Vaneerdeweg. A performance study comparing manual and robotically assisted laparoscopic surgery using the da vinci system. *Surgical Endoscopy And Other Interventional Techniques*, 17(10):1595–1599, 2003. ISSN 0930-2794. doi: 10.1007/s00464-002-9248-1. URL <http://dx.doi.org/10.1007/s00464-002-9248-1>.
- J. F. Hughs, A. V. Dam, M. McGuire, D. F. Sklar, J. D. Foley, S. K. Feiner, and K. Akeley. *Computer graphics: principles and practice*. Addison-Wesley, 3rd edition, 2013.
- M. Ikits, J. D. Brederson, C. D. Hansen, and C. R. Johnson. A constraint-based technique for haptic volume exploration. In *Proc. IEEE Int. Conf. Vis.*, pages 263–269, Oct. 2003. doi: 10.1109/VI-SUAL.2003.1250381.
- Intuitive Surgical, Inc. The da vinci surgery experience, 2013. [Online] PN 1005195 Rev B 12/13.
- M. Jakopc, S. J. Harris, F. Rodriguez y Baena, P. Gomes, and B. L. Davies. The acrobot system for total knee replacement. *Industrial Robot: An Int. J.*, 30(1):61–66, 2003a. doi: 10.1108/01439910310457724.
- M. Jakopc, F. Rodriguez y Baena, S. Harris, P. Gomes, J. Cobb, and B. L. Davies. The hands-on orthopaedic robot ”Acrobot”: Early clinical trials of total knee replacement surgery. *IEEE Trans. Robot. Autom.*, 19(5):902–911, Oct. 2003b. ISSN 1042-296X. doi: 10.1109/TRA.2003.817510.
- D. L. James and D. K. Pai. BD-tree: output-sensitive collision detection for reduced deformable models. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, SIGGRAPH ’04, pages 393–398, New York, NY, USA, 2004. ACM. doi: 10.1145/1186562.1015735.
- J. Jang, H. W. Kim, and Y. S. Kim. Efficient neurosurgical monitoring method for operative safety on image-guided brain tumor resection. In *Int. J. Comp. Assist. Radiol. Surg.*, volume 7, pages (Suppl 1):S128–S130, June 2012. doi: 10.1007/s11548-012-0708-3.
- G. R. Jatzko, P. H. Lisborg, M. G. Muller, and V. M. Wette. Recurrent nerve palsy after thyroid operations—principal nerve identification and a literature review. *Surgery*, 115:139–144, 1994.
- J.-P. Jeannon, A. A. Orabi, G. A. Bruch, H. A. Abdalsalam, and R. Simo. Diagnosis of recurrent laryngeal nerve palsy after thyroidectomy: a systematic review. *International Journal of Clinical Practice*, 63(4):624–629, 2009. ISSN 1742-1241. doi: 10.1111/j.1742-1241.2008.01875.x.
- P. Jimenez, F. Thomas, and C. Torras. 3d collision detection: A survey. *Comp. and Graph.*, 25:269–285, 2000.
- P. Joli, S. Payandeh, M. Chan, and B. Bayart. A new approach to solve constraint forces of virtual fixtures in haptic rendering. In *12th World Congress in Mechanism and Machine Sci. in conj. with IFToMM*, 2007.

- L. D. Joly and C. Andriot. Imposing motion constraints to a force reflecting telerobot through real-time simulation of a virtual mechanism. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 357–362, May 1995. doi: 10.1109/ROBOT.1995.525310.
- S.-W. Kang, S. C. Lee, S. H. Lee, K. Y. Lee, J. J. Jeong, Y. S. Lee, K.-H. Nam, H. S. Chang, W. Y. Chung, and C. S. Park. Robotic thyroid surgery using a gasless, transaxillary approach and the da vinci s system: The operative outcomes of 338 consecutive patients. *Surgery*, 146(6):1048 – 1055, 2009. ISSN 0039-6060. doi: <http://dx.doi.org/10.1016/j.surg.2009.09.007>.
- A. Kapoor and R. H. Taylor. A constrained optimization approach to virtual fixtures for multi-handed tasks. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 3401–3406, May 2008.
- A. Kapoor, M. Li, and R. H. Taylor. Spatial motion constraints for robot assisted suturing using virtual fixtures. In *Proc. Int. Conf. Med. Img. Comp. & Comp. Asst. Int.*, volume 8, pages 89–96. Springer-Verlag, 2005.
- A. Kapoor, M. Li, and R. H. Taylor. Constrained control for surgical assistant robots. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 231 – 236, 15-19, 2006. doi: 10.1109/ROBOT.2006.1641189.
- O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.*, 5: 90–98, Apr. 1986. ISSN 0278-3649. doi: 10.1177/027836498600500106.
- I. Kienzie, T.C., S. Stulberg, M. Peshkin, A. Quaid, and C.-H. Wu. An integrated CAD-robotics system for total knee replacement surgery. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 1609–1614, Oct. 1992. doi: 10.1109/ICSMC.1992.271508.
- R. Kikuuwe, N. Takesue, A. Sano, H. Mochiyama, and H. Fujimoto. Admittance and impedance representations of friction based on implicit euler integration. *IEEE Trans. Robot.*, 22(6):1176–1188, Dec. 2006. ISSN 1552-3098. doi: 10.1109/TRO.2006.886262.
- R. Kikuuwe, N. Takesue, and H. Fujimoto. A control framework to generate nonenergy-storing virtual fixtures: Use of simulated plasticity. *IEEE Trans. Robot.*, 24(4):781–793, Aug. 2008.
- J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE Trans. Vis. Comput. Graphics*, 4(1):21–36, 1998. ISSN 1077-2626. doi: 10.1109/2945.675649.
- D. Knuth. *The Art of Computer Programming: Sorting and Searching*, volume 3. Addison-Wesley, 1973.
- K. Kosuge, T. Itoh, T. Fukuda, and M. Otsuka. Tele-manipulation system based on task-oriented virtual tool. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 351–356, May 1995. doi: 10.1109/ROBOT.1995.525309.
- D. Kragic, P. Marayong, M. Li, A. M. Okamura, and G. D. Hager. Human-machine collaborative systems for microsurgical applications. *Int. J. Robot. Res.*, 24(9):731–741, 2005. doi: 10.1177/02783649050507059.
- A. B. Kuang, S. Payandeh, B. Zheng, F. Henigman, and C. MacKenzie. Assembling virtual fixtures for guidance in training environments. In *Proc. Symp. Haptic Interf. Virt. Env. and Teleop. Sys.*, pages 367–374, Mar. 2004.
- R. Kumar, A. Kapoor, and R. H. Taylor. Preliminary experiments in robot/human cooperative microin-

- jection. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, volume 3, pages 3186–3191, Oct. 2003. doi: 10.1109/IROS.2003.1249647.
- Y. S. Kwoh, J. Hou, E. A. Jonckheere, and S. Hayati. A robot with improved absolute positioning accuracy for CT guided stereotactic brain surgery. *IEEE Trans. Biomed. Eng.*, 35(2):153–160, 1988. doi: 10.1109/10.1354.
- K.-W. Kwok, V. Vitiello, and G.-Z. Yang. Control of an articulated snake robot under dynamic active constraints. In *Proc. Int. Conf. Med. Img. Comp. & Comp. Asst. Int.*, volume 6363 of *Lecture Notes in Computer Science*, pages 229–236. Springer Berlin / Heidelberg, 2010.
- K.-W. Kwok, K. H. Tsoi, V. Vitiello, J. Clark, G. Chow, W. Luk, and G.-Z. Yang. Dimensionality reduction in controlling articulated snake robot for endoscopy under dynamic active constraints. *IEEE Trans. Robot.*, 29(1):15–31, 2013. ISSN 1552-3098. doi: 10.1109/TRO.2012.2226382.
- D.-S. Kwon, J.-J. Lee, Y.-S. Yoon, S.-Y. Ko, J. Kim, J.-H. Chung, C.-H. Won, and J.-H. Kim. The mechanism and registration method of a surgical robot for hip arthroplasty. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 2, pages 1889–1894, 2002. doi: 10.1109/ROBOT.2002.1014816.
- E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha. Fast proximity queries with swept sphere volumes. Technical Report TR99-018, Department of Computer Science, UNC Chapel Hill, 1999.
- T. Larsson and T. Akenine-Möller. Collision detection for continuously deforming bodies. In *Eurographics*, pages 325–333, 2001.
- C. Lauterbach, Q. Mo, and D. Manocha. gProximity: Hierarchical GPU-based operations for collision and distance queries. In *Proceedings of Eurographics*, volume 29, pages 419–428, 2010.
- C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Society for Industrial and Applied Mathematics, 1995.
- M. Li and A. M. Okamura. Recognition of operator motions for real-time assistance using virtual fixtures. In *Proc. Symp. Haptic Interf. Virt. Env. and Teleop. Sys.*, pages 125 – 131, Mar. 2003. doi: 10.1109/HAPTIC.2003.1191253.
- M. Li and R. H. Taylor. Optimum robot control for 3D virtual fixture in constrained ENT surgery. In *Proc. Int. Conf. Med. Img. Comp. & Comp. Asst. Int.*, volume 2878, pages 165–172. Springer Berlin / Heidelberg, 2003. ISBN 978-3-540-20462-6.
- M. Li and R. H. Taylor. Performance of surgical robots with automatically generated spatial virtual fixtures. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 217–222, Apr. 2005. doi: 10.1109/ROBOT.2005.1570122.
- M. Li, A. Kapoor, and R. H. Taylor. A constrained optimization approach to virtual fixtures. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1408 – 1413, Aug. 2005. doi: 10.1109/IROS.2005.1545420.
- M. Li, M. Ishii, and R. H. Taylor. Spatial motion constraints using fixtures generated by anatomy. *IEEE Trans. Robot.*, 23(1):4–19, Feb. 2007.
- M. Lin and J. Canny. A fast algorithm for incremental distance calculation. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 2, pages 1008–1014, Apr. 1991. doi: 10.1109/ROBOT.1991.131723.
- M. C. Lin and S. Gottschalk. Collision detection between geometric models: A survey. In *IMA Conference on Mathematics of Surfaces*, pages 37–56, 1998.

- M. C. Lin and D. Manocha. *Handbook of Discrete and Computational Geometry*, chapter Collision and proximity queries, pages 787–808. Chapman & Hall/CRC, 2004.
- E. Lopez, K.-W. Kwok, C. Payne, P. Giataganas, and G.-Z. Yang. Implicit active constraints for robot-assisted arthroscopy. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 5390–5395, May 2013. doi: 10.1109/ICRA.2013.6631350.
- M. M. Malhotra, R. Subramanian, P. S. Gahlot, and B. S. Rathore. *Textbook in Applied Mechanics*. New Age International Publishers, 1994.
- P. Marayong and A. M. Okamura. Speed-accuracy characteristics of human-machine cooperative manipulation using virtual fixtures with variable admittance. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 46(3):518–532, 2004. doi: 10.1518/hfes.46.3.518.50400.
- P. Marayong, M. Li, A. M. Okamura, and G. D. Hager. Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1954–1959, 2003.
- P. Marayong, G. D. Hager, and A. M. Okamura. Effect of hand dynamics on virtual fixtures for compliant human-machine interfaces. In *Proc. Symp. Haptic Interf. Virt. Env. and Teleop. Sys.*, pages 109–115, Mar. 2006. doi: 10.1109/HAPTIC.2006.1627075.
- P. Marayong, G. D. Hager, and A. M. Okamura. Control methods for guidance virtual fixtures in compliant human-machine interfaces. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1166–1172, Sep. 2008. doi: 10.1109/IROS.2008.4650838.
- L. Marchal-Crespo and D. J. Reinkensmeyer. Review of control strategies for robotic movement training after neurologic injury. *J. of NeuroEng. and Rehab.*, 6(20), 2009. doi: 10.1186/1743-0003-6-20.
- M. Matinfar, C. Baird, A. Batouli, R. Clatterbuck, and P. Kazanzides. Robot-assisted skull base surgery. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 865 –870, Nov. 2007. doi: 10.1109/IROS.2007.4399012.
- W. A. McNeely, K. D. Puterbaugh, and J. J. Troy. Six degree-of-freedom haptic rendering using voxel sampling. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, 1999.
- D. Meagher. Geometric modelling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, 1982. doi: 10.1016/0146-664X(82)90104-6.
- U. Meier, O. Lopez, C. Monserrat, M. Juan, and M. Alcaniz. Real-time deformable models for surgery simulation: a survey. *Computer Methods and Programs in Biomedicine*, 77(3):183–197, 2005. ISSN 0169-2607. doi: 10.1016/j.cmpb.2004.11.002.
- F. Mémoli and G. Sapiro. Fast computation of weighted distance functions and geodesics on implicit hyper-surfaces. *J. of Comp. Phys.*, 173(2):730–764, 2001. ISSN 0021-9991. doi: 10.1006/jcph.2001.6910.
- J. Mezger, S. Kimmerle, and O. Etmub. Hierarchical techniques in collision detection for cloth animation. *J. of WSCG*, 11(1):322–329, Feb. 2003.
- A. Micaelli, C. Bidard, and C. Andriot. Decoupling control based on virtual mechanisms for telemanipulation. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 3, pages 1924 –1931 vol.3, May 1998. doi: 10.1109/ROBOT.1998.680594.

- B. Mirtich. V-clip: Fast and robust polyhedral collision detection. *ACM Trans. on Graph.*, 17:177–208, 1998.
- B. Mirtich and J. Canny. Impulse-based simulation of rigid bodies. In *Symposium on Interactive 3D graphics*, pages 181–217, 1995.
- G. Moustris, A. Mantelos, and C. Tzafestas. Enhancing surgical accuracy using virtual fixtures and motion compensation in robotic beating heart surgery. In *Mediterranean Conference on Control Automation*, pages 1254–1260, June 2013. doi: 10.1109/MED.2013.6608880.
- N. Navkar, Z. Deng, D. Shah, and N. Tsekos. A framework for integrating real-time mri with robot control: Application to simulated transapical cardiac interventions. *IEEE Trans. Biomed. Eng.*, 60(4): 1023–1033, Apr. 2013. ISSN 0018-9294. doi: 10.1109/TBME.2012.2230398.
- N. V. Navkar, Z. Deng, D. J. Shah, K. E. Bekris, and N. V. Tsekos. Visual and force-feedback guidance for robot-assisted interventions in the beating heart with real-time MRI. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 689–694, May 2012. doi: 10.1109/ICRA.2012.6224582.
- J. T. Nolin, P. M. Stemniski, and A. M. Okamura. Activation cues and force scaling methods for virtual fixtures. In *Proc. Symp. Haptic Interf. Virt. Env. and Teleop. Sys.*, pages 404–409, 2003.
- M. Novotni and R. Klein. Computing geodesic distances on triangular meshes. In *Proc. of WSCG*, pages 341–347, 2002.
- M. Ortega, S. Redon, and S. Coquillart. A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. *IEEE Trans. Vis. Comput. Graphics*, 13(3):458–469, May 2007. ISSN 1077-2626. doi: 10.1109/TVCG.2007.1028. URL <http://dx.doi.org/10.1109/TVCG.2007.1028>.
- T. Ortmaier, H. Weiss, U. Hagn, M. Grebenstein, M. Nickl, A. Albu-Schaffer, C. Ott, S. Jorg, R. Konietzschke, L. Le-Tien, and G. Hirzinger. A hands-on-robot for accurate placement of pedicle screws. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 4179–4186, May 2006. doi: 10.1109/ROBOT.2006.1642345.
- C. O’Sullivan and J. Dingliana. Realtime collision detection and response using sphere-trees. In *Proc. Spring Conf. on Comp. Graph.s*, pages 83–92, 1999.
- M. Otaduy and M. Lin. A modular haptic rendering algorithm for stable and transparent 6-dof manipulation. *IEEE Trans. Robot.*, 22(4):751–762, Aug. 2006. ISSN 1552-3098. doi: 10.1109/TRO.2006.876897.
- J. W. Park, J. Choi, Y. Park, and K. Sun. Haptic virtual fixture for robotic cardiac catheter navigation. *Artificial Organs*, 35(11):1127–1131, 2011. ISSN 1525-1594. doi: 10.1111/j.1525-1594.2011.01373.x.
- S. Park, R. D. Howe, and D. F. Torchiana. Virtual fixtures for robotic cardiac surgery. In *Proc. Int. Conf. Med. Img. Comp. & Comp. Asst. Int.*, pages 1419–1420. Springer-Verlag, 2001. ISBN 3-540-42697-3.
- C. Passenberg, R. Groten, A. Peer, and M. Buss. Towards real-time haptic assistance adaptation optimizing task performance and human effort. In *Proc. IEEE World Haptics Conf.*, pages 155–160, June 2011. doi: 10.1109/WHC.2011.5945478.
- H. A. Paul, W. L. Bargar, B. Mittlestadt, B. Musits, R. H. Taylor, P. Kazanzides, J. Zuhars, B. Williamson, and W. Hanson. Development of a surgical robot for cementless total hip arthroplasty. *Clinical Orthopaedics and Related Research*, 285:57–66, 1992.

- S. Payandeh and Z. Stanisic. On application of virtual fixtures as an aid for telemanipulation and training. In *Proc. Symp. Haptic Interf. Virt. Env. and Teleop. Sys.*, pages 18–23, 2002. doi: 10.1109/HAPTIC.2002.998936.
- C. Payne, K.-W. Kwok, and G.-Z. Yang. An ungrounded hand-held surgical device incorporating active constraints with force-feedback. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 2559–2565, Nov. 2013. doi: 10.1109/IROS.2013.6696717.
- F. Pearson. *Map Projections: Theory and Applications*. CRC Press Inc., 1990.
- A. R. Peon, C. Pacchierotti, and D. Prattichizzo. Vibrotactile stimuli for augmented haptic feedback in robot-assisted surgery. In *Proc. IEEE World Haptics Conf.*, pages 473–478, Apr. 2013. doi: 10.1109/WHC.2013.6548454.
- M. A. Peshkin, J. E. Colgate, W. Wannasuphoprasit, C. A. Moore, R. B. Gillespie, and P. Akella. Cobot architecture. *IEEE Trans. Robot. Autom.*, 17(4):377–390, Aug. 2001. ISSN 1042-296X. doi: 10.1109/70.954751.
- J. G. Petersen and F. Rodriguez y Baena. A dynamic active constraints approach for hands-on robotic surgery. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1966–1971, 2013.
- Z. Pezzementi, A. M. Okamura, and G. D. Hager. Dynamic guidance with pseudoadmittance virtual fixtures. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1761–1767, Apr. 2007. doi: 10.1109/ROBOT.2007.363577.
- R. Prada and S. Payandeh. A study on design and analysis of virtual fixtures for cutting in training environments. In *Proc. IEEE World Haptics Conf.*, pages 375–380, Mar. 2005.
- R. Prada and S. Payandeh. On study of design and implementation of virtual fixtures. *Virtual Reality*, 13(2):117–129, June 2009. ISSN 1359-4338. doi: 10.1007/s10055-009-0115-4.
- S. Quinlan. Efficient distance computation between non-convex objects. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 4, pages 3324–3329, May 1994. doi: 10.1109/ROBOT.1994.351059.
- J. Rao, A. Dragulescu-Andrasi, and H. Yao. Fluorescence imaging in vivo: recent advances. *Current Opinion in Biotechnology*, 18(1):17–25, 2007. ISSN 0958-1669. doi: <http://dx.doi.org/10.1016/j.copbio.2007.01.003>. Analytical biotechnology.
- S. Redon, A. Kheddar, and S. Coquillart. Gauss’ least constraints principle and rigid body simulations. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 517–522, 2002. doi: 10.1109/ROBOT.2002.1013411.
- J. Ren, R. V. Patel, K. A. McIsaac, G. Guiraudon, and T. M. Peters. Dynamic 3-D virtual fixtures for minimally invasive beating heart procedures. *IEEE Trans. Med. Imag.*, 27(8):1061–1070, 2008.
- L. B. Rosenberg. The use of virtual fixtures as perceptual overlays to enhance operator performance in remote environments. Technical Report AL/CF-TR-1994-0089, USAF Armstrong Laboratory, 1992.
- L. B. Rosenberg. Virtual fixtures: Perceptual tools for telerobotic manipulation. In *Proc. IEEE Virt. Real. Annual Int. Symp.*, pages 76–82, Sep. 1993.
- D. C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. In *24th Annual Conf. Comp. Graph. & Inter. Tech.*, pages 345–352. ACM Press/Addison-Wesley Publishing Co., 1997. ISBN 0-89791-896-7. doi: 10.1145/258734.258878.

- F. Rydén and H. Chizeck. Forbidden-region virtual fixtures from streaming point clouds: Remotely touching and protecting a beating heart. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 3308–3313, Oct. 2012. doi: 10.1109/IROS.2012.6386012.
- F. Rydén and H. Chizeck. A proxy method for real-time 3-dof haptic rendering of streaming point cloud data. *IEEE Trans. Haptics*, 6(3):257–267, July 2013. ISSN 1939-1412. doi: 10.1109/TOH.2013.20.
- J.-H. Ryu, D.-S. Kwon, and B. Hannaford. Stable teleoperation with time-domain passivity control. *IEEE Trans. Robot. Autom.*, 20(2):365–373, Apr. 2004. ISSN 1042-296X. doi: 10.1109/TRA.2004.824689.
- E. Sacco, T. Prayer-Galetti, F. Pinto, S. Fracalanza, G. Betto, F. Pagano, and W. Artibani. Urinary incontinence after radical prostatectomy: incidence by definition, risk factors and temporal trend in a large series with a long-term follow-up. *BJU International*, 97(6):1234–1241, 2006. ISSN 1464-410X. doi: 10.1111/j.1464-410X.2006.06185.x.
- J. Sackier and Y. Wang. Robotically assisted laparoscopic surgery. *Surgical Endoscopy*, 8(1):63–66, 1994. ISSN 0930-2794. doi: 10.1007/BF02909496.
- H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Publishing Company, Inc., 1989.
- C. Sayers and R. Paul. Synthetic fixturing. In *Proc. ASME Winter Annual Meeting*, volume 49, pages 37–46, 1993.
- C. Sayers, R. Paul, and M. Mintz. Operator interaction and teleprogramming for subsea manipulation. In *Proc. 4th IARP Workshop on Underwater Robot.*, Genova, Italy, Nov. 1992.
- H. J. Seddon. A classification of nerve injuries. *British Medical Journal*, 2(4260):237–239, 1942.
- J. Serpell, S. Woodruff, M. Bailey, S. Grodski, and M. Yeung. Recurrent laryngeal nerve diameter increases during thyroidectomy. *Annals of Surgical Oncology*, 18(6):1742–1747, 2011. ISSN 1068-9265. doi: 10.1245/s10434-010-1531-0.
- S. Seung, B. Kang, W. Kim, J. Park, and S. Park. Development of image guided master-slave system for minimal invasive brain surgery. In *41st Int. Symp. Robot. & 6th German Conf. on Robot.*, pages 1–6, June 2010.
- T. B. Sheridan. Telerobotics. *Automatica*, 25(4):487 – 507, 1989. ISSN 0005-1098. doi: 10.1016/0005-1098(89)90093-9.
- M. Shoham, I. H. Lieberman, E. C. Benzel, D. Togawa, E. Zehavi, B. Zilberstein, M. Roffman, A. Bruskin, A. Fridlander, L. Joskowicz, S. Brink-Danan, and N. Knoller. Robotic assisted spinal surgery-from concept to clinical practice. *Computer Aided Surgery*, 12(2):105–115, 2007. doi: 10.3109/10929080701243981.
- W. Siebert and S. Mai. One year clinical experience using the robot system caspar for TKR. In *Annual North American Program on Computer Assisted Orthopaedic Surgery*, pages 141–142, 2001.
- J. Spillmann, M. Becker, and M. Teschner. Efficient updates of bounding sphere hierarchies for geometrically deformable models. *J. Vis. Comm. & Image Rep.*, 18(2):101–108, Apr. 2007. ISSN 1047-3203. doi: 10.1016/j.jvcir.2007.01.001.

- G. Srimathveeravalli, V. Gourishankar, and T. Kesavadas. Comparative study: Virtual fixtures and shared control for rehabilitation of fine motor skills. In *Proc. IEEE World Haptics Conf.*, pages 304–309, Mar. 2007. doi: 10.1109/WHC.2007.31.
- M. A. Srinivasan and C. Basdogan. Haptics in virtual environments: Taxonomy, research status and challenges. *Computers & Graphics*, 21(4):393–404, 1997.
- Stanmore Implants Worldwide Ltd. Stanmore Sculptor RGA (Robotic Guidance Arm), 2012. URL <http://www.stanmoreimplants.com/>.
- R. G. D. Steel, J. H. Torrie, and D. A. Dickey. *Principles and Procedures of Statistics: A Biometrical Approach*. McGraw-Hill, Inc., 3rd edition, 1997.
- V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler, and H. Hoppe. Fast exact and approximate geodesics on meshes. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, volume 24, pages 553–560, New York, NY, USA, July 2005. ACM. doi: 10.1145/1073204.1073228.
- A. Tang, Q. Cao, and T. Pan. Spatial motion constraints for a minimally invasive surgical robot using customizable virtual fixtures. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 2013. ISSN 1478-596X. doi: 10.1002/rcs.1551.
- R. Taylor and D. Stoianovici. Medical robotics in computer-integrated surgery. *IEEE Trans. Robot. Autom.*, 19(5):765–781, Oct. 2003. ISSN 1042-296X. doi: 10.1109/TRA.2003.817058.
- R. H. Taylor, H. A. Paul, B. D. Mittelstadt, W. Hanson, P. Kazanzides, J. F. Zuhars, E. Glassman, B. L. Musits, B. Williamson, and W. L. Bargar. An image-directed robotic system for hip replacement surgery. *J. Joint Robot. Soc. Japan*, 8(5):111–116, 1990.
- R. H. Taylor, C. B. Cutting, Y. Kim, A. D. Kalvin, D. Larose, B. Haddad, D. Khoramabadi, M. Noz, R. Olyha, N. Bruun, and D. Grimm. A model-based optimal planning and execution system with active sensing and passive manipulation for augmentation of human precision in computer-integrated surgery. In *Int. Symp. on Exp. Robot.*, pages 177–195, 1991a.
- R. H. Taylor, H. A. Paul, P. Kazanzides, B. D. Mittelstadt, W. Hanson, J. F. Zuhars, B. Williamson, B. L. Musits, E. Glassman, and W. L. Bargar. Taming the bull: Safety in a precise surgical robot. In *Proc. IEEE Int. Conf. Adv. Robot.*, 1991b.
- R. H. Taylor, H. A. Paul, C. B. Cutting, B. D. Mittelstadt, W. Hanson, P. Kazanzides, B. L. Musits, Y.-Y. Kim, A. Kalvin, B. Haddad, D. Khoramabadi, and D. Larose. Augmentation of human precision in computer-integrated surgery. *Innovations and Tech. in Biol. and Med.*, 13(4):450–468, 1992.
- D. Terzopoulos and H. Qin. Dynamic NURBS with geometric constraints for interactive sculpting. *ACM Trans. on Graph.*, 13(2):103–136, Apr. 1994.
- M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. Collision detection for deformable objects. *Computer Graphics Forum*, 24(1):61–81, Mar. 2005. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2005.00829.x.
- A. Toniato, R. Mazzarotto, A. Piotto, P. Bernante, C. Pagetta, and M. R. Pelizzo. Identification of the nonrecurrent laryngeal nerve during thyroid surgery: 20-year experience. *World Journal of Surgery*, 28(7):659–661, 2004. ISSN 0364-2313. doi: 10.1007/s00268-004-7197-7.

- J. Troccaz and Y. Delnondedieu. Semi-active guiding systems in surgery. a two-dof prototype of the passive arm with dynamic constraints (PADyC). *Mechatronics*, 6(4):399–421, 1996. ISSN 0957-4158. doi: 10.1016/0957-4158(96)00003-7.
- J. Troccaz, M. Peshkin, and B. Davies. Guiding systems for computer-assisted surgery: introducing synergistic devices and discussing the different approaches. *Medical Image Analysis*, 2(2):101–119, 1998. ISSN 1361-8415. doi: 10.1016/S1361-8415(98)80006-6.
- S. Troyan, V. Kianzad, S. Gibbs-Strauss, S. Gioux, A. Matsui, R. Oketokoun, L. Ngo, A. Khamene, F. Azar, and J. Frangioni. The FLARE intraoperative near-infrared fluorescence imaging system: A first-in-human clinical trial in breast cancer sentinel lymph node mapping. *Annals of Surgical Oncology*, 16(10):2943–2952, 2009. ISSN 1068-9265. doi: 10.1245/s10434-009-0594-2. URL <http://dx.doi.org/10.1245/s10434-009-0594-2>.
- N. Turro, O. Khatib, and E. Coste-Maniere. Haptically augmented teleoperation. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 1, pages 386 – 392, 2001. doi: 10.1109/ROBOT.2001.932582.
- S. G. Unruh, T. N. Faddis, and B. G. Barr. Position-assist shared control of a force-reflecting telerobot. In H. Das, editor, *Proc. SPIE*, volume 1833, pages 113–121, 1993. doi: 10.1117/12.142102.
- G. van den Bergen. Efficient collision detection of complex deformable models using AABB trees. *J. Graph. Tools*, 2(4):1–13, Jan. 1998. ISSN 1086-7651.
- V. Vitiello, K.-W. Kwok, C. Payne, and G.-Z. Yang. DOF minimization for optimized shape control under active constraints for a hyper-redundant flexible robot. In *Information Processing in Computer-Assisted Interventions*, volume 6689, pages 67–78. Springer Berlin / Heidelberg, 2011. ISBN 978-3-642-21503-2.
- H. E. Wagner and C. Seiler. Recurrent laryngeal nerve palsy after thyroid gland surgery. *British Journal of Surgery*, 81(2):226–228, 1994. doi: 10.1002/bjs.1800810222.
- P. C. Walsh and J. L. Mostwin. Radical prostatectomy and cystoprostatectomy with preservation of potency. results using a new nerve-sparing technique. *British Journal of Urology*, 56(6):694–697, 1984. ISSN 1464-410X. doi: 10.1111/j.1464-410X.1984.tb06149.x.
- P. C. Walsh, H. Lepor, and J. C. Eggleston. Radical prostatectomy with preservation of sexual function: Anatomical and pathological considerations. *The Prostate*, 4:473–485, 1983.
- O. M. Wazni, C. Barrett, D. O. Martin, M. Shaheen, K. Tarakju, B. Baranowski, A. Hussein, T. Callahan, T. Dresing, M. Bhargava, M. Kanj, P. Tchou, A. Natale, and W. Saliba. Experience with the hansen robotic system for atrial fibrillation ablation-lessons learned and techniques modified: Hansen in the real world. *Journal of Cardiovascular Electrophysiology*, 20(11):1193–1196, 2009. ISSN 1540-8167. doi: 10.1111/j.1540-8167.2009.01539.x.
- O. Weede, H. Monnich, B. Muller, and H. Worn. An intelligent and autonomous endoscopic guidance system for minimally invasive surgery. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 5762–5768, May 2011. doi: 10.1109/ICRA.2011.5980216.
- D. R. Wilhelmsen. A nearest point algorithm for convex polyhedral cones and applications to positive linear approximations. *Mathematics of Computation*, 30(133):48–57, Jan. 1976.
- J. C. Willems. Dissipative dynamical systems. *Euro. J. Control*, 13:134–151, 2007.

- J. P. Williams, R. H. Taylor, and L. B. Wolff. Augmented k-d techniques for accelerated registration and distance measurement of surfaces. In *Computer Aided Surgery: Computer-Integrated Surgery of the Head and Spine*, Linz, Austria, 1997.
- T. Xia, C. Baird, G. Jallo, K. Hayes, N. Nakajima, N. Hata, and P. Kazanzides. An integrated system for planning, navigation and robotic assistance for skull base surgery. *Int. J. Med. Robot. Comp. Ass. Surg.*, 4(4):321–330, 2008. ISSN 1478-596X. doi: 10.1002/rcs.213.
- T. Xia, A. Kapoor, P. Kazanzides, and R. Taylor. A constrained optimization approach to virtual fixtures for multi-robot collaborative teleoperation. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 639–644, Sep. 2011. doi: 10.1109/IROS.2011.6095056.
- T. Xia, S. Leonard, A. Deguet, L. Whitcomb, and P. Kazanzides. Augmented reality environment with virtual fixtures for robotic telemanipulation in space. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 5059–5064, Oct. 2012. doi: 10.1109/IROS.2012.6386169.
- T. Yamamoto, N. Abolhassani, S. Jung, A. M. Okamura, and T. N. Judkins. Augmented reality and haptic interfaces for robot-assisted surgery. *Int. J. Med. Robot. Comp. Ass. Surg.*, 8(1):45–56, 2012. ISSN 1478-596X. doi: 10.1002/rcs.421.
- W. Yu, R. Alqasemi, R. Dubey, and N. Pernalet. Telemanipulation assistance based on motion intention recognition. In *Proc. IEEE Int. Conf. Robot. Autom.*, pages 1121–1126, Apr. 2005. doi: 10.1109/ROBOT.2005.1570266.
- G. Zachmann. Rapid collision detection by dynamically aligned dop-trees. In *Proc. IEEE Virt. Real. Annual Int. Symp.*, pages 90–97, 1998. doi: 10.1109/VRAIS.1998.658428.
- D. Zhang, L. Wang, J. Gu, Z. Li, and K. Chen. Realization of spatial compliant virtual fixture using eigenscrews. In *Proc. IEEE Int. Conf. Eng. Med. Biol. Soc.*, pages 1506–1509, 2012.
- X. Zhang and Y. Kim. Interactive collision detection for deformable models using streaming AABBs. *IEEE Trans. Vis. Comput. Graphics*, 13(2):318–329, Mar. 2007. ISSN 1077-2626. doi: 10.1109/TVCG.2007.42.
- Y. Zhuang and J. Canny. Haptic interaction with global deformations. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 3, pages 2428–2433, 2000. doi: 10.1109/ROBOT.2000.846391.
- C. B. Zilles and J. K. Salisbury. A constraint-based god-object method for haptic display. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, volume 3, pages 146–151, Aug. 1995. doi: 10.1109/IROS.1995.525876.

Appendix A

Proofs

A.1 Simplification of the elasto-plastic friction model

When using the elasto-plastic model to represent Coulomb friction, without elastic pre-sliding, the discrete operations in equations (12) and (13) of [Hayward et al., 2009] reduce to a single saturation operation upon a candidate z value z_c which represents a fully elastic displacement.

Proof. By setting $f_{max} = f_C$ in the standard Coulomb model for elasto-plastic friction (equation (6) in [Hayward et al., 2009]), the steady-state z value $z_{ss}(\dot{x})$ (equation (9) in [Hayward et al., 2009]) becomes:

$$z_{ss}(\dot{x}) = \begin{cases} z_{css} & \dot{x} \geq 0 \\ -z_{css} & otherwise \end{cases} \quad (\text{A.1})$$

where $z_{css} = f_C / \sigma_0$.

This representation of $z_{ss}(\dot{x})$ allows the fractional inequality used in the elasto-plastic friction model's saturation operations (equations (12) and (13)) to be represented as follows:

$$\frac{z}{z_{ss}(\dot{x})} < 1 \Leftrightarrow (|z| < z_{css}) \vee (\text{sgn}(\dot{x}) \neq \text{sgn}(z)) \quad (\text{A.2})$$

Using this notation, a bi-directional step function, equivalent to that shown in figure 5 of [Hayward et al.,

2009] without the breakaway deflection, can be used to define $\alpha(z, \dot{x})$:

$$\alpha(z, \dot{x}) = \begin{cases} 0 & (|z| < z_{css}) \vee (\text{sgn}(\dot{x}) \neq \text{sgn}(z)) \\ 1 & \text{otherwise} \end{cases} \quad (\text{A.3})$$

By the equivalence in equation (A.2), when the expression for $\alpha(z, \dot{x})$ in equation (A.3) is substituted into the top subfunction of equation (12) in [Hayward et al., 2009] it is clear that $\forall z \forall \dot{x} \alpha(z, \dot{x}) = 0$. The function subsequently simplifies to:

$$y_k = \begin{cases} \Delta x_k & \frac{z_{k-1}}{z_{ss}(v_k)} < 1 \\ 0 & \text{otherwise} \end{cases} \quad (\text{A.4})$$

From equation (A.4) it is known that:

$$\frac{z_{k-1}}{z_{ss}(v_k)} \geq 1 \implies y_k = 0 \quad (\text{A.5})$$

Therefore it must also hold that:

$$\frac{z_{k-1}}{z_{ss}(v_k)} \geq 1 \implies \left(\frac{z_{k-1}}{z_{ss}(v_k)} \right) = \left(\frac{z_{k-1} + y_k}{z_{ss}(v_k)} \right) \quad (\text{A.6})$$

Consolidation of the two saturation operations gives:

$$z_k = \begin{cases} z_c & \left(\frac{z_{k-1}}{z_{ss}(v_k)} < 1 \right) \wedge \left(\frac{z_c}{z_{ss}(v_k)} < 1 \right) \\ z_{ss}(v_k) & \text{otherwise} \end{cases} \quad (\text{A.7})$$

where $z_c = z_{k-1} + \Delta x_k$ is the candidate z value representing a fully elastic displacement.

By expressing the conditions in equation (A.7) using the alternate form in equation (A.2), it can be seen that the first clause of the top subfunction is redundant and therefore this operation further simplifies to:

$$z_k = \begin{cases} z_c & (|z_c| < z_{css}) \vee (\text{sgn}(\dot{x}) \neq \text{sgn}(z_c)) \\ z_{ss}(v_k) & \text{otherwise} \end{cases} \quad (\text{A.8})$$

As the magnitude of z_{css} is constant, it can be proven by induction that:

$$|z_c| \geq z_{css} \implies \text{sgn}(\dot{x}) = \text{sgn}(z_c) \quad (\text{A.9})$$

Such that equation (A.8) reduces to:

$$z_k = \begin{cases} z_c & |z_c| < z_{css} \\ z_{ss}(v_k) & \text{otherwise} \end{cases} \quad (\text{A.10})$$

Substituting the definition for $z_{ss}(v_k)$ from equation (A.1), and using the implication in equation (A.9), the final simplified equation for z_k is:

$$z_k = \begin{cases} z_c & |z_c| < z_{css} \\ z_{css} & z_c \geq z_{css} \\ -z_{css} & z_c \leq -z_{css} \end{cases} \quad (\text{A.11})$$

□

A.2 Dissipativity of dynamic frictional constraints in $\mathbb{R}^3$

A system is said to be dissipative [Willems, 2007] if $\forall t_1 \geq 0$ and $\forall t_2 \geq t_1$:

$$\int_{t_1}^{t_2} s(u(t), y(t)) dt \geq V(a(t_2)) - V(a(t_1)) \quad (\text{A.12})$$

where $u(t)$ and $y(t)$ are the system inputs and outputs at time t respectively, $s(\cdot)$ is the supply rate of energy into the system, $V(\cdot)$ is a continuous, non-negative storage function and $a(t)$ is the system state variable. Dynamic frictional constraints conform to this definition of dissipativity.

Proof. Dissipativity can be proven by substituting the robot power for the supply rate, the spring energy for the storage function and the elastic displacement for the system state variable in equation (A.12). The inequality subsequently becomes:

$$\int_{t_1}^{t_2} \mathbf{f}(t) \cdot \dot{\mathbf{x}}(t) dt \geq \frac{1}{2} \sigma_0 \|\mathbf{z}(t_2)\|^2 - \frac{1}{2} \sigma_0 \|\mathbf{z}(t_1)\|^2 \quad (\text{A.13})$$

It is possible to prove this inequality by showing that $\forall t \geq 0$ the derivative of the storage function is never greater than the supply rate:

$$\mathbf{f}(t) \cdot \dot{\mathbf{x}}(t) \geq \frac{d}{dt} \frac{1}{2} \sigma_0 \|\mathbf{z}(t)\|^2 \quad (\text{A.14})$$

$$\mathbf{f}(t) \cdot \dot{\mathbf{x}}(t) \geq \sigma_0 \|\mathbf{z}(t)\| \frac{d\|\mathbf{z}(t)\|}{dt} \quad (\text{A.15})$$

Assuming that the damping elements of the constraint force are dissipative:

$$\sigma_0 \mathbf{z}(t) \cdot \dot{\mathbf{x}}(t) \geq \sigma_0 \|\mathbf{z}(t)\| \frac{d\|\mathbf{z}(t)\|}{dt} \quad (\text{A.16})$$

$$\sigma_0 \frac{\mathbf{z}(t)}{\|\mathbf{z}(t)\|} \cdot \dot{\mathbf{x}}(t) \geq \sigma_0 \frac{d\|\mathbf{z}(t)\|}{dt} \quad (\text{A.17})$$

$$\sigma_0 \hat{\mathbf{z}}(t) \cdot \dot{\mathbf{x}}(t) \geq \sigma_0 \frac{d\|\mathbf{z}(t)\|}{dt} \quad (\text{A.18})$$

$$\hat{\mathbf{z}}(t) \cdot \dot{\mathbf{x}}(t) \geq \frac{d\|\mathbf{z}(t)\|}{dt} \quad (\text{A.19})$$

where the overhat symbol identifies a normalised vector.

Discretising the model at time k , and making the assumption that the time-step Δt is small such that derivatives can be approximated using the implicit Euler method, the power inequality for translation in $\mathbb{R}^3$ becomes:

$$\hat{\mathbf{z}}_{t,k} \cdot \frac{\Delta \mathbf{x}_{t,k}}{\Delta t} \geq \frac{\|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k-1}\|}{\Delta t} \quad (\text{A.20})$$

$$\hat{\mathbf{z}}_{t,k} \cdot \Delta \mathbf{x}_{t,k} \geq \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k-1}\| \quad (\text{A.21})$$

The remainder of this section shows how this discrete power inequality holds true for all configurations of the dynamic frictional constraints in translation.

Equation (A.21) is initially proven for the first and second subfunctions of equation (5.16) where:

$$\mathbf{z}_{t,k} = \frac{\mathbf{z}_{t,k-1} + \Delta \mathbf{x}_{t,k}}{\beta} \quad \text{where} \quad \beta \geq 1 \quad (\text{A.22})$$

$$\Delta \mathbf{x}_{t,k} = \beta \mathbf{z}_{t,k} - \mathbf{z}_{t,k-1} \quad (\text{A.23})$$

Substituting this into equation (A.21) gives:

$$\hat{\mathbf{z}}_{t,k} \cdot (\beta \mathbf{z}_{t,k} - \mathbf{z}_{t,k-1}) \geq \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k-1}\| \quad (\text{A.24})$$

$$\hat{\mathbf{z}}_{t,k} \cdot \beta \mathbf{z}_{t,k} - \hat{\mathbf{z}}_{t,k} \cdot \mathbf{z}_{t,k-1} \geq \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k-1}\| \quad (\text{A.25})$$

$$\beta \|\mathbf{z}_{t,k}\| - \hat{\mathbf{z}}_{t,k} \cdot \mathbf{z}_{t,k-1} \geq \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k-1}\| \quad (\text{A.26})$$

$$\beta \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k}\| + \|\mathbf{z}_{t,k-1}\| \geq \hat{\mathbf{z}}_{t,k} \cdot \mathbf{z}_{t,k-1} \quad (\text{A.27})$$

Applying Cauchy's inequality to the right hand side:

$$\beta \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k}\| + \|\mathbf{z}_{t,k-1}\| \geq \|\mathbf{z}_{t,k-1}\| \quad (\text{A.28})$$

$$\beta \|\mathbf{z}_{t,k}\| - \|\mathbf{z}_{t,k}\| \geq 0 \quad (\text{A.29})$$

$$\beta \|\mathbf{z}_{t,k}\| \geq \|\mathbf{z}_{t,k}\| \quad (\text{A.30})$$

$$\beta \geq 1 \quad (\text{A.31})$$

□

Equation (A.21) is next proven for the third subfunction of equation (5.16) when $\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} \leq 0$. Given equation (5.17):

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} \leq 0 \implies \|\mathbf{z}_{t,k}\| = 0 \quad (\text{A.32})$$

By substituting $\|\mathbf{z}_{t,k}\| = 0$ into equation (A.16), both left and right hand sides become zero proving the inequality. □

Equation (A.21) is finally proven for the third subfunction of equation (5.16), when $\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} > 0$, where:

$$\mathbf{z}_{t,k} = \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) \hat{\mathbf{y}} \quad \text{where} \quad 1 \geq \gamma \geq 0 \quad (\text{A.33})$$

and

$$\Delta \mathbf{x}_{t,k} = \mathbf{z}_{t,c} - \mathbf{z}_{t,k-1} \quad (\text{A.34})$$

and given equation (5.17), such that:

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} > 0 \implies \|\mathbf{z}_{t,k}\| > 0 \implies \hat{\mathbf{z}}_{t,k} = \hat{\mathbf{y}} \quad (\text{A.35})$$

Giving:

$$\hat{\mathbf{y}} \cdot (\mathbf{z}_{t,c} - \mathbf{z}_{t,k-1}) \geq \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) - \|\mathbf{z}_{t,k-1}\| \quad (\text{A.36})$$

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} - \hat{\mathbf{y}} \cdot \mathbf{z}_{t,k-1} \geq \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) - \|\mathbf{z}_{t,k-1}\| \quad (\text{A.37})$$

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} - \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) + \|\mathbf{z}_{t,k-1}\| \geq \hat{\mathbf{y}} \cdot \mathbf{z}_{t,k-1} \quad (\text{A.38})$$

Applying Cauchy's inequality to the right hand side:

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} - \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) + \|\mathbf{z}_{t,k-1}\| \geq \|\mathbf{z}_{t,k-1}\| \quad (\text{A.39})$$

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} - \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) \geq 0 \quad (\text{A.40})$$

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c} \geq \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{t,c}) \quad (\text{A.41})$$

$$1 \geq \gamma \quad (\text{A.42})$$

□

A.3 Dissipativity of dynamic frictional constraints in SO(3)

Dynamic frictional constraints in SO(3) are dissipative, as defined in the criterion in equation (A.12).

Proof. For rotations in SO(3), the discrete power inequality derived from equation (A.19) (complementary to equation (A.21) for translation) is:

$$\hat{\mathbf{z}}_{r,k} \cdot \Delta \mathbf{x}_{r,k} \geq \|\mathbf{z}_{r,k}\| - \|\mathbf{z}_{r,k-1}\| \quad (\text{A.43})$$

The remainder of this section shows how this discrete power inequality holds true for all configurations of the dynamic frictional constraints in rotation.

Equation (A.43) is initially proven for the first and second subfunctions of equation (5.23), given equa-

tion (5.18), where:

$$\mathbf{z}_{r,k} = \frac{\text{Quat2RV}(\Delta \mathbf{x}_{q,k} * \mathbf{z}_{q,k-1})}{\beta} \quad \text{where } \beta \geq 1 \quad (\text{A.44})$$

$$\beta \mathbf{z}_{r,k} = \text{Quat2RV}(\Delta \mathbf{x}_{q,k} * \mathbf{z}_{q,k-1}) \quad (\text{A.45})$$

$$\text{RV2Quat}(\beta \mathbf{z}_{r,k}) = \Delta \mathbf{x}_{q,k} * \mathbf{z}_{q,k-1} \quad (\text{A.46})$$

$$\text{RV2Quat}(\beta \mathbf{z}_{r,k}) * \mathbf{z}_{q,k-1}^{-1} = \Delta \mathbf{x}_{q,k} \quad (\text{A.47})$$

$$\Delta \mathbf{x}_{q,k} = \text{RV2Quat}(\beta \mathbf{z}_{r,k}) * \mathbf{z}_{q,k-1}^{-1} \quad (\text{A.48})$$

$$\Delta \mathbf{x}_{r,k} = \text{Quat2RV}(\text{RV2Quat}(\beta \mathbf{z}_{r,k}) * \mathbf{z}_{q,k-1}^{-1}) \quad (\text{A.49})$$

where Quat2RV is a function which converts rotation representations from unit quaternions to rotation vectors and RV2Quat is its inverse.

As the magnitude of the elastic displacement $\mathbf{z}$ within dynamic frictional constraints is both small and bounded by the constant z_{css} , and as the magnitude of the user's displacement at each time-step is small, the above quaternion equation can be linearly approximated as follows:

$$\Delta \mathbf{x}_{r,k} = \beta \mathbf{z}_{r,k} - \mathbf{z}_{r,k-1} \quad (\text{A.50})$$

This formulation for the relationship between $\mathbf{z}_{r,k}$ and $\mathbf{z}_{r,k-1}$ in rotation is the same as that in equation (A.23) for translation. The proof for the first and second subfunctions of equation (5.23) therefore follows the same method as that in equations (A.24) to (A.31) where translation vectors are substituted for rotation vectors. $\square$

Equation (A.43) is next proven for the third subfunction of equation (5.23) when $\hat{\mathbf{y}} \cdot \mathbf{z}_{r,c} \leq 0$. Given equation (5.24):

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{r,c} \leq 0 \implies \|\mathbf{z}_{r,k}\| = 0 \quad (\text{A.51})$$

By substituting $\|\mathbf{z}_{r,k}\| = 0$ into equation (A.16), both left and right hand sides become zero proving the inequality. $\square$

Equation (A.43) is finally proven for the third subfunction of equation (5.23), when $\hat{\mathbf{y}} \cdot \mathbf{z}_{r,c} > 0$, where:

$$\mathbf{z}_{r,k} = \gamma(\hat{\mathbf{y}} \cdot \mathbf{z}_{r,c}) \hat{\mathbf{y}} \quad \text{where } 1 \geq \gamma \geq 0 \quad (\text{A.52})$$

Given equation (5.18), such that:

$$\mathbf{z}_{q,k} = \Delta \mathbf{x}_{q,k} * \mathbf{z}_{q,k-1} \quad (\text{A.53})$$

$$\Delta \mathbf{x}_{q,k} = \mathbf{z}_{q,k} * \mathbf{z}_{q,k-1}^{-1} \quad (\text{A.54})$$

$$\Delta \mathbf{x}_{r,k} = \text{Quat2RV} \left(\mathbf{z}_{q,k} * \mathbf{z}_{q,k-1}^{-1} \right) \quad (\text{A.55})$$

and given equation (5.24), such that:

$$\hat{\mathbf{y}} \cdot \mathbf{z}_{r,c} > 0 \implies \|\mathbf{z}_{r,k}\| > 0 \implies \hat{\mathbf{z}}_{r,k} = \hat{\mathbf{y}} \quad (\text{A.56})$$

As previously, the magnitude of the elastic displacement $\mathbf{z}$ within dynamic frictional constraints is both small and bounded by the constant z_{css} , and the magnitude of the user's displacement at each time-step is small, therefore the above quaternion equation can be linearly approximated as follows:

$$\Delta \mathbf{x}_{r,k} = \mathbf{z}_{r,c} - \mathbf{z}_{r,k-1} \quad (\text{A.57})$$

This formulation for the relationship between $\mathbf{z}_{r,c}$ and $\mathbf{z}_{r,k}$ in rotation is the same as that in equation (A.34) for translation. The proof for this case therefore follows the same method as that in equations (A.36) to (A.42) where translation vectors are substituted for rotation vectors. $\square$