A System For Automated Vision-Guided Suturing

by

Santosh Iyer

A thesis submitted in conformity with the requirements for the degree of Master of Applied Science in Biomedical Engineering
Graduate Department of IBBME
University of Toronto

© Copyright 2012 by Santosh Iyer
Abstract

A System For Automated Vision-Guided Suturing

Santosh Iyer
Master of Applied Science in Biomedical Engineering
Graduate Department of IBBME
University of Toronto
2012

Suturing in laparoscopic surgery is a challenging and time-consuming task that presents haptic, motor and spatial constraints for the surgeon. As a result, there is variability in surgical outcome when performing basic suturing tasks such as knot tying, stitching and tissue dissection (as large as 50%).

This goal of this thesis is to develop a standardized, proof-of-concept, automated robotic suturing system that performs a side-to-side anastomosis with image guidance and dynamic trajectory control. A passive alignment tool is created for rigidly constraining needle pose, and robust computer vision algorithms are used to track surface features and the suture needle. A robotic system integrates these components to autonomously pass a curved suture needle through sequential loops in a tissue pad phantom.
Dedication

Dedicated to my loving parents, who have provided me with the support to pursue this work.
Acknowledgements

I would like to thank SickKids for providing me with the facilities to pursue my research and the CIGITI lab (Thomas, Brian, Hamid, Karl) for providing me with motivation and support. I would like to thank my supervisor Dr. Drake and my committee for providing me with the superior guidance and direction to make this novel work a success.
Contents

1 Introduction ........................................ 1
  1.1 Thesis Outline .................................. 2

2 Background ......................................... 4
  2.1 Challenges In Laparoscopic Suturing ............... 4
    2.1.1 Trocar Constrained Robotic Laparoscopic Operation .... 5
  2.2 Rapid Laparoscopic Stitching Tools .................. 7
  2.3 Autonomous Modes For Suturing ..................... 7
  2.4 Stitching Planning In Laparoscopic Suturing .......... 8
  2.5 Needle and Surface Feature Tracking For 3D Navigation . 11

3 System Overview ..................................... 12
  3.1 Task Modelling For Laparoscopic Suturing .......... 12
    3.1.1 Coordinate Systems ........................... 12
    3.1.2 A Typical Robotically-Guided Stitching Task .......... 14
  3.2 System Architecture ................................ 17
    3.2.1 System Setup .................................. 17
    3.2.2 Workflow Diagram .............................. 19
    3.2.3 Subsystem Communication Protocol ............... 20

4 Mechanical Subsystem ............................... 24
  4.1 7-DOF Inverse Kinematics For Needle Tip Manipulation . 24
    4.1.1 Background .................................... 24
    4.1.2 Design Methodology ............................ 31
    4.1.3 Implementation ................................ 32
  4.2 Passive Needle Alignment Tool ....................... 33
    4.2.1 Design Methodology ............................ 33
    4.2.2 Results ........................................ 33
  4.3 Trocar Constrained Operation ....................... 35
    4.3.1 Virtual RCM Constraints ....................... 35
    4.3.2 Optimal Trocar Placement ...................... 36
5 Vision Subsystem

5.1 Background - Filters For Feature Segmentation and Shape Estimation

5.2 Surface Feature Segmentation
  5.2.1 Methodology
  5.2.2 Results

5.3 Needle Segmentation
  5.3.1 Methodology
  5.3.2 Results

5.4 Curve Approximation and Ellipse Fitting
  5.4.1 Methodology
  5.4.2 Results

5.5 Circular Pose Estimation
  5.5.1 Background
  5.5.2 Methodology

5.6 Calibration and Registration
  5.6.1 Methodology
  5.6.2 Results

5.7 Depth and Positioning Accuracy

5.8 Touch Point Calibration

6 Trajectory Generation Subsystem || Curved Needle Path Planning

6.1 Planning A Minimal Deformation Path For Smooth Needle Insertion

6.2 Determining An Optimal Centre Point (OPC) of Rotation

6.3 Generating Circular Needle Trajectories

7 System Performance

7.1 Lack of Suture Thread
  7.1.1 Orientation Change In The Needle When Releasing (After Insertion)

7.2 Challenges With Re-Aligning At The Second Suture Insertion Point

7.3 Quantifying Needle Positioning Accuracy

7.4 Procedure Repeatability
  7.4.1 Consistency For Controlled Lighting/Suture Point Locations
  7.4.2 Sensitivity To Lighting
  7.4.3 Sensitivity To Suture Point Locations
List of Figures

2.1 RCM Implementations .................................................. 6
2.2 The 5 Parameters That Define Needle Pose During Laparoscopic Suturing ............. 8

3.1 Laparoscopic Setup For Suturing ........................................ 13
3.2 Trocar-Tool Frame .......................................................... 13
3.3 Tool-Needle Frame .......................................................... 14
3.4 Needle-Tissue Frame ........................................................ 15
3.5 Modelling Of The Stitching Task .......................................... 16
3.6 Experimental Setup Integration ............................................ 18
3.7 Workflow Diagram For an Automated Suturing Procedure ................................. 21
3.8 Communication Protocol Layout For Subsystem Components ............................ 23

4.1 A Block Diagram Of The Mechanical Subsystem ........................................ 25
4.2 Pieper's Theorem For Three Intersecting Axes .......................................... 30
4.3 Independent Joint PID Control ............................................... 30
4.4 Modified Passive Needle Alignment Tool-Tip for Robotic-Guided Suturing ............. 34
4.5 Visualization of Remote-centre-Of Motion in Robotic Laparoscopic Procedures ........................ 36
4.6 Port Placement Optimization -RCM Points with the largest reachable workspace ....... 38
4.7 Optimized trocar Placement for Maximizing Robotic Workspace In Laparoscopic Surgery 38
5.1 A Block Diagram Of The Vision Subsystem ............................................. 41
5.2 Skeletonized Image ................................................................................. 49
5.3 Distance Transform ................................................................................ 50
5.4 Robust Surface Feature Thresholding ...................................................... 56
5.5 Individual Surface Feature Detection ...................................................... 57
5.6 Tool-Invariant Surface Feature Segmentation ......................................... 57
5.7 Background Subtraction ......................................................................... 58
5.8 High Needle Specificity - Subtractive Intensity Based Thresholding .......... 59
5.9 Hough Lines Approximation Mask For Tool Based Occlusions .............. 60
5.10 Occlusion Invariant Needle Segmentation and Ellipse Estimation .......... 61
5.11 The Ellipse Fitting Module in OpenCV .................................................... 62
5.12 Inconsistent Ellipses .............................................................................. 62
5.13 Surface Feature Ellipse Fitting For Lighting-Invariant Tracking ............ 64
5.14 Lighting-Invariant RANSAC Constrained Needle Ellipse Fitting ............ 65
5.15 Workflow Diagram For Ellipse-To-Circle Backprojection Algorithm ....... 68
5.16 Circular Grid Pattern Used For Calibration ............................................ 70
5.17 Distortion Correction Using Circular Grid Patterns For Calibration ......... 71
5.18 $N^{th}$ Order Fits To Determine Calibration Accuracy - RMS Error ........ 72
5.19 4-Point Circular Grid To Quantify Depth Resolution ............................. 73
5.20 Synthetic Circular Grid Varied In Magnification To Reflect Placing the Grid Closer/Farther From The Camera ......................................................... 73
5.21 Depth Resolution Results - Synthetic Images ......................................... 74
5.22 Depth Resolution Results - Real World Images ...................................... 75
5.23 Synthetic Image Setup: Circle Rotated About the Major (Horizontal Axis), To Quantify Angle Resolution ................................................................. 76
5.24 Angle Resolution Results - Synthetic Images ........................................................................ 77
5.25 Real World Image Setup: Coloured Needle Actuated By Robotic Tool, To Quantify Angle
Resolution. .......................................................................................................................... 78
5.26 Angle Resolution Results - Real World Images ................................................................. 79

6.1 A Block Diagram Of The Trajectory Generation Subsystem ............................................. 81
6.2 Infinite Solutions For An Optimal Centre Point Calculation ........................................... 83
6.3 Calculating the true “insertion” angle $\alpha$ as a deviation from the tissue surface normal . 85

7.1 Accurate Insertion and Pose Estimation Of A Curved Needle ........................................... 89
7.2 Accurate Pick Up Of a Curved Needle .................................................................................. 90
7.3 A Tapered VISI Black Needle ............................................................................................... 91
7.4 Procedure Repeatability - Consistency, Sensitivity To Lighting, and Sensitivity To Suture
Point Locations .................................................................................................................... 94

A.1 DH Parameter Table For the 7-DOF Denso System .......................................................... 104

B.1 Secant Line Intersections For Accurate Circle Centre Detection ...................................... 107
Chapter 1

Introduction

Minimally invasive surgery (MIS), when compared to open-surgery, is a promising cost-effective, safe alternative that could potentially reduce patient recovery times. Ultimately, however, MIS has spatial, motor and haptic constraints that ultimately limit the surgeon in his/her ability to perform a surgery accurately and effectively [17]. Surgeons operate long, rigid surgical tools through trocars, which limit MIS procedures to four degrees of freedom. For visual feedback, they are guided by two-dimensional endoscopic camera images, with lack of depth information. As a result, these challenges introduce variability amongst many inexperienced surgeons when performing basic suturing tasks, such as knot tying, stitching, and tissue dissection (as large as 50% ) [17].

The DaVinci surgical robotic system has been widely adopted in laparoscopic cardiac and gynecological surgeries, used for suturing through teleoperative control (surgeons remotely manipulating laparoscopic tools through a console)[18]. The system provides 7 degree of freedom laparoscopic EndoWrist tools, 3D Visualization, and tremor filtration to facilitate easier, more accurate suturing procedures (compared to manual, rigid tool, laparoscopic suturing) [18].

Fundamentally however, 1) manipulating curved needles, 2) dealing with a flexible suture, 3) getting a trajectory for knot placement and 4) applying proper tension [17] in a 2D endoscopic, deformable tissue environment still make teleoperated MIS suturing a challenging technical task.

As a result, there is a need for an automated suturing tool that addresses the dexterity and perception constraints associated with conventional MIS, while being versatile enough to suture various deformable tissue types in a rapid and reliable manner.

To address speed in suturing, active research in semi-automated suturing tool design have shown promising preliminary results in terms of consistency and time-efficiency in suturing procedures. Circumventing
knot tying and suture/needle manipulation, these devices close vessel conduits in a rapid-firing fashion. Unfortunately, success has been limited to extensive prep work by the surgeon (manual wall eversion), limited case-specific functionality (due to spatial design constraints), and manual control to position the device [17, 33]. Robotic systems driving these tools have also been experimented with to add speed, precision and control to the procedure [17]. Functionally however, these devices only operate with idealized (often preset) points for stitching and knot tying, using encoders to simulate rigid aspects of suturing.

To improve robustness, image-guidance for MIS suturing tasks have been explored with the **DaVinci** using feature-based estimation models to guide a robot in assisted stitching, knot-tying, and cutting [27]. Gesture-based needle and tool tracking have been used to build complex trajectory templates for automated knot tying and stitching tasks[17], while B-spline flexible suture approximations have been used to guide automated scissors to cut a suture during a suturing task [27]. These approaches use standard curved needles to perform suturing, and show promising results in various simulated and clinical experimental scenarios [27]. However, integrating these assistive algorithms for an end-to-end cohesive automated suturing task has not yet been pursued.

The purpose of this research project is to develop a versatile robotically-controlled suturing tool that autonomously performs a continuous side-to-side anastomosis in a safe and efficient manner. This is accomplished using robust vision algorithms, innovative tool design, and RCM-compensated motion planning to laparoscopically perform a continuous suturing procedure. Shape and feature estimation algorithms dynamically track the suture needle and surface markers, and a modified needle driver passively aligns a curved suture needle for easy insertion. A robotic system integrates these vision and passive rigidity features to generate needle trajectories that accurately pass through deformable entry and exit points. Needle positioning accuracy has to be within 1-2mm, and the system has to perform the suturing task at comparable human speeds - within seconds to minutes, depending on the complexity of the tissue geometry.

### 1.1 Thesis Outline

This thesis is divided into 7 sections:

Chapter 2 provides a broad literature survey focusing on topics pertaining to trocar constrained operation and laparoscopic suturing.

Chapter 3 provides a broad overview of the automated suturing system.

Chapter 4, 5 and 6 detail the mechanical, vision and trajectory generation subsystems critical to realizing the task of automated suturing. Relevant background information, methodology/design, and implement-
tation/results are categorically presented for each functional component within each subsystem to aid in easy user navigation.

Chapter 7 addresses system-level issues and provides automated suturing performance results for a variety of lighting, positioning and repeatability conditions.

Chapter 8 discusses some of the current and future challenges that need to be addressed before this system is clinically feasible.
Chapter 2

Background

The literature survey presented here explores challenges in port constrained operation within minimally invasive surgery, motion planning for laparoscopic suturing, and surgical feature tracking to enhance motion task accuracy in a 3D environment.

2.1 Challenges In Laparoscopic Suturing

Laparoscopic tools in minimally invasive surgery (MIS) operate through surgical ports (small incisions made on the patient’s body) to access various internal structures. This pivotal constraint is defined as a Remote Center-of-Motion (RCM) [19], and is an important component for MIS procedures. Fundamentally, RCM constrained MIS procedures introduce kinematic constraints and severe motor limitations for the surgeon [19]. A flat and restricted bidimensional field of view, along with lack of visual and (minimal) tactile feedback make depth, position and angle estimation extremely challenging [19].

Suturing is one of the most difficult gestures in laparoscopic surgery [25], which is performed using two needle holders. A circular needle is manipulated with these instruments to pass a suture thread back and forth, until 1) stitching closes a gap and 2) the suture is secured by way of knot-tying [25].

Stitching is extremely sensitive to the kinematic constraints present in laparoscopic surgery. In an open scenario, it requires six DOFs to position and orient the needle, where as knot tying only requires three [25]. Furthermore, needles can assume any position and orientation in the jaws of the needle holder (an unconstrained system), and as a result, the stitching task has an extremely large number of parameters (mainly angles, at the trocar and site of operation). Surgeons cannot accurately estimate or predict
these parameters with 2D endoscopic images, due to the uncontrolled grabbing of the needle with the tool, and unpredictable motion of the needle in the tissue.

Kinematic constraints within the abdomen have been somewhat alleviated by advances in robotics and flexible wrist manipulators, such as the DaVinci from Intuitive Surgical, Zeus from Computer Motion Inc., Black Falcon and the UCSF/UCB surgical robot [25], which have been guided in design to target particular surgical gestures like suturing.

2.1.1 Trocar Constrained Robotic Laparoscopic Operation

MIS robotic systems are constrained to operate about an RCM in 3 ways:

**Mechanically constrained kinematic structures** (Figure 2.1, Left), such as the DaVinci, use a bar parallelogram mechanism to physically constrain the device to only operate about a fixed point [19]. Even if software control is faulty, the rigidity of the mechanism prevents excessive translation about a surgical trocar (thereby minimizing risk in patient safety). This makes parallel linkage systems the preferred method for robotic MIS procedures, albeit their specificity for laparoscopic interventions prevents use for other, more generic tasks.

**Passively constrained tools** (Figure 2.1, Left), such as those employed by a guided prostate seed-insertion system[29], pivot with two degrees of freedom (linear positioning and insertion) and passively maintain an RCM[19]. Like mechanically constrained systems, these are also preferred clinically due their restricted range of motion (and minimizing patient complications in the event of system failure).

**Virtual, trocar constraining control software** (Figure 2.1, Right) is used by general-purpose robotic manipulators to ensure a tool is constrained to an RCM all times. Clinically however, this is problematic as a software fault can lead to violation of RCM (and compromise patient safety). Despite these drawbacks, virtual control can still achieve high levels of pivotal constraining and provide a realistic laparoscopic surgical experience. For these reasons, such a system is popular in research settings [19] and was implemented in this setup.
Figure 2.1: RCM implemented through: Mechanically constrained kinematic structures and tools passively maintaining RCM (Left), and active, software-based virtual RCM systems with flexible manipulators (Right).


2.2 Rapid Laparoscopic Stitching Tools

Instruments designed to simplify laparoscopic stitching (such as staples, clip applicators, or needle-shuttling devices), while promising, have several limitations.

Staple-based anastomotic tools were developed as early as the 1950s by Androsov [33]. The idea was to provide greater speed and accuracy without compromising patient safety. The Cardica C-port is a modern laparoscopic stapler that performs an interrupted suture technique to close conduits as small as 1mm. While experimental and clinical results were positive (patency rates of 97%), wall eversion and graft manipulation still limit this technique to end-to-side anastomosis procedures [33]. 2-3 additional stitches are required to close any remaining holes.

Suture shuttle devices such as the Endostitch (by Covidien)[33] transfer a needle between two jaws and lock the tapered tip of the needle to one of the jaws at a time. Clinically, this device has been successful for procedures such as gastrocystoplasties [33], which involve intracorporeal suturing. Limitations of the Endostitch include its 10mm width and short dull needle that cannot pass through thick tissue, resulting in more trauma than a similarly sized wedged-on suture [33]. The needle can only be passed perpendicularly from jaw to jaw and may require excess tissue manipulation for proper suture placement. Finally, the device is disposable and reloads are costly, adding unnecessary expense (standard needle drivers are reusable)[21].

2.3 Autonomous Modes For Suturing

As a means of relieving the surgeon from repetitive and difficult tasks in laparoscopic suturing, autonomous modes have been proposed.

Taylor et al., Casals et al. and Wang et. al proposed automated control schemes which moved the endoscope to follow the motions of the surgical or to automatically position the instruments.

Kang and Wen et al. proposed autonomous modes for knot tying using robotics. An EndoStitch needle device was used to autonomously stitch tissue phantoms, and a geometric knot tying module (with two 4DOF “Endobots”) was used to accomplish automated knot tying.

While promising, lack of vision feedback, case-specific stitching success, non-laparoscopic (free-space operation) and lack of compensation for organ movement (i.e. anastomosis on a beating heart) make the Endobot systems rather limited in application, and proof-of-concept, at best. Furthermore, studies [10] have suggested that knot tying is a major time-consuming and task-limiting step during the process of suturing, so any attempt to reduce the time for surgical anastomosis should minimize or ideally avoid the knot-tying step.


## 2.4 Stitching Planning In Laparoscopic Suturing

Nageotte *et al.* presents a comprehensive stitching planning methodology for laparoscopic applications, addressing challenging kinematic and pose constraints through concise geometric formulations that can be applied to any keyhole surgical suturing task [25]. Their work focuses on optimal path planning techniques for needles to pass through tissues (with any geometry). It was designed to be used in augmented reality environments, or as a path planning tool for robotic guidance/assistance in surgery.

They divide the stitching task into 5 steps, that accurately guide a curved suturing needle from entry point $I^*$ to exit point $O^*$, while minimizing tissue deformation. 5 parameters define the state of the needle: the 3D position of the intersection point $I$, the angle of the part of the needle under the tissue $\beta_n$, and a tool pose angle $\rho$ (Figure 2.2).

![Figure 2.2: The 5 Parameters ($I_x, I_y, I_z, \beta_n, \rho$) That Define Needle Pose During Laparoscopic Suturing. $I$ corresponds to the 3D needle/tissue intersection point, $\beta_n$ corresponds to the angle of the part of the needle under the tissue, and $\rho$ corresponds to the tool pose angle.](image)

**Modelling Tissue Deformations**

During normal stitching, the contact between the needle and tissue can be considered as as a point
contact. During insertion or pulling out, the deformation caused by the needle at ($I^*$ or $O^*$) can be represented by the difference between the current position of these points and their rest positions. This can be decomposed along two directions: tangential to the surface of the tissue (shear deformation) and normal to the tissue (longitudinal deformation). Longitudinal and shear deformations depends on the dynamic behaviour of tissues, through multiple parameters such as elasticity, stiffness, tissue-needle contact, and the motion of the needle.

Longitudinal deformations arise when piercing into the insertion point, but can be limited if the needle is normal to the tissue ($90^\circ$) during insertion, and no frictional forces (quasi-static) between the needle and tissue are present [25]. The latter is a reasonable assumption, since most curved suturing needles are made of stainless steel (with no rough surfaces), so there is no pull on the tissue during piercing.

Shear deformations arise when the motion of the needle does not rotate about a fixed axis of rotation (some motion is off-plane) from the entry point $I^*$ to the exit point $O^*$.

In laparoscopic surgery, the ideal motion (positioning the needle approach normal to the tissue, and rotating about a plane) is generally not feasible since it requires free positioning and orientation of the needle in Euclidean space [25]. As a result, surgeons are often forced to position the needle non-orthogonally at the insertion point, compensate for shear tissue deformation forces, and iteratively change the pose of the needle midway to ensure it exits through $O^*$. Often, the needle is (unintuitively) not directed towards the exit point, due to deformation and awkward laparoscopic constraints.

Given these limitations, the following path planning methodology attempts to define a “best-case scenario” for a given set of RCM constraints, to accomplish laparoscopic suturing.

**Choice of Initial Position**

The needle tip is constrained to intersect the entry point, $I^*$ by ensuring $I = I^*$ and $\beta_n = 0$. The value of $\rho$ is not imposed, due to laparoscopic constraints. Practically, however, the best value for $\rho$ is chosen by sampling from a range. For each value $\rho_j$, an initial configuration cost $ICC(\rho_j)$ is calculated for a ($I^*, 0, \rho_j$) state:

$$ ICC(\rho_j) = \begin{cases} 
\alpha + \frac{\lambda}{dHT} & \text{if } dHT < dist_{lim}, \\
\alpha & \text{otherwise}
\end{cases}$$  

(2.1)

where $\alpha$ is the angle between the tangent to the tip of the needle and normal to the tissue. The second term of the cost function depends on $dHT$, the distance between the tissues and the handling point (the
Chapter 2. Background

needle/needle holder intersection), and it is used to stop the needle holder from being brought too close to the tissue.

The limit distance $\text{dist}_{\text{lim}}$ is chosen as a function of the needle to needle tool tip distance, $b$. Nageotte et al. chose $\text{dist}_{\text{lim}} = 2b$ [25]. The chosen initial position is then chosen to be: $(I^*, 0, \arg \min_{\rho_j} ICC(\rho_j))_c$.

Choice of Final Position

The final position, similar to the initial position, can be constrained by ensuring $I = O^*$ and $\beta_n = 0$. A simple way to choose the final position can be by finding a value of $\rho$ which minimizes the distance between $I^*$ and $I_{\text{fin}}$ [25].

Planning a Minimal Deformation Path

Assume a suitable initial and final position have been chosen. Then the maximum allowable tissue depth angle $\beta_n \text{ max}$ at the exit can be defined by:

$$\beta_n \text{ max} = 2\sin^{-1}\left(\frac{\|I_{\text{fin}}O^*\|}{2r_a}\right)$$

(2.2)

This will be accomplished if and only if the deformation remains lower than some upper limit, which can be described by a surface region $A_{\text{def}}$ around the desired entry point $I^*$. This patch is geometry specific, and may not necessarily be symmetric around $I^*$ [25]. The surgeon usually specifies an acceptable amount of deformation, which defines the size of the surface region $A_{\text{def}}$, and a minimal deformation path is chosen such that $I_{\text{fin}} - I^*$ does not fall outside this region.

An optimal insertion trajectory is calculated with these constraints in mind, based on finding a minimal distance path (generated from a cost function) in an oriented graph (with obstacles) from the entry point to the exit point:

$$C(n_{i,j}) = \lambda_E C_E(n_{i,j}) + \lambda_H C_H(n_{i,j}) + \lambda_L C_L(n_{i,j})$$

(2.3)

Where $C_E$ describes a repulsive function that prevents the needle from being too close to the surface of the tissue, $C_H$ is another repulsive function that prevents the needle holder from being too close/colliding to the tissue during insertion, and $C_L$ describes the shortest and smoothest motion path of the needle.

Note, the cost-function optimized path trajectory permits the needle to be reoriented while it is in the tissue, to ensure path deformation is minimal and the tip comes as close to $O^*$ as possible.
Chapter 2. Background

Thick Tissue Considerations

The stitching planning method presented by Nagoette et al. assumes that tissues are thin, which allows needle-reorientation during the procedure [25]. While circular needles are extremely stiff relative to the tissue (and will not deform during insertion), during thick tissue insertion, complex motions induced by optimal-graph based path planning techniques will result in tissue damage.

To prevent these issues, the ideal motion should follow the arc of the needle (i.e. rotate about its centrepoint) while it is in the tissue. An additional fast increasing cost function \( C_T \) can be added to equation 2.3, describing the distance \( d \) between the actual path (for the current needle configuration) \((I, \beta_n, \rho)\) and the ideal path, \(IP(\beta_n)\) along the arc of the needle [25].

2.5 Needle and Surface Feature Tracking For 3D Navigation

Small object tracking is an active area of interest in MIS procedures due to loss of depth perception [36] with monocular endoscopic cameras. As a result, manipulating needles, screws and plates is a technically challenging task. Furthermore, partial occlusion of the object (due to laparoscopic tools and anatomical obstructions) complicates the surgery, adding inaccuracy, inconsistency and guesswork on the surgeons part.

Several approaches have focused on surface-mounted fiducials to optically track an object of interest [36]. Magnetic tracking is not used due to interference from other surgical accessories (clips, needles, surgical tools, etc). Depth is artificially computed based on the pose of the object, and relayed to the surgeon for easier manipulation. Line of sight is key to ensure this method works, and any form of occlusion destroys tracking capability. Depending on object size (i.e. a needle), this method is quite restrictive.

Marker-free alternatives, such as structured light endoscopes and ultrasound probes have also been investigated, but were only validated against larger surgical instruments [36].

The most promising, occlusion free method for small object tracking is based on de Ipinia et al.’s TRIP tag algorithm, designed to provide real-time pose information from circular features [20]. Wengert et al.’s algorithm provides a simple and computationally efficient algorithm to track small circular objects (such as needles) in an endoscopic view, with no additional changes to the surgical scene (expensive visualization equipment, or markers) [20].
Chapter 3

System Overview

3.1 Task Modelling For Laparoscopic Suturing

Figure 3.1 illustrates the RCM-constrained setup for laparoscopic suturing. The tool and needle are placed through a trocar, along with the endoscope camera system. The goal is to suture through a series of input/output entry/exit points ($I^*/O^*$) by rotating on a fixed plane about the needle centre axis, to minimize longitudinal and transverse tissue deformation (i.e. on the pink tissue phantom) [25].

Figure 3.2 - 3.4 define camera, robot/world, tool, needle, tissue and trocar frames, orienting the user with a geometric/mathematical description of the robotic suturing setup. These frames are extensively used in mechanical, vision and trajectory subsystem formulation.

3.1.1 Coordinate Systems

**Position of The Needle Holder w.r.t The Patient:** The tool in Figure 3.2 (attached to frame $F_{to}$) manipulates the needle through a trocar (attached to frame $F_{tr}$). $F_c$ corresponds to the camera coordinate system frame. Pose estimation and perspective transformations are expressed through this camera frame, and converted to the $F_w$ robot world coordinate frame through a registration scheme (Section 3.5.3, Calibration and Registration) for vision-guided robotic suturing.

Unless otherwise mentioned, all frames are expressed in terms of the robot world coordinate frame, $F_w$.

**Position of The Needle w.r.t The Needle Holder:** The contact frame between the needle and needle holder, $F_{nt}$, changes during automated suturing due to release/pick up of the needle (Figure 3.3).
Chapter 3. System Overview

Figure 3.1: Laparoscopic Setup For Suturing with Entry Point $I$ and Exit Point $O$.

Figure 3.2: Trocar-Tool Frame
Chapter 3. System Overview

Figure 3.3: Tool-Needle Frame

The vision system detects pose changes in the needle (by extracting the needle centre point axis, $F_n$), and, variably changes the relative position of $F_{nt}$ based on this information, and updates the robot kinematics model to ensure accurate needle contact during various phases of the suturing sequence.

Position of The Needle w.r.t The Tissue Surface: For ideal needle insertion, the needle centre axis $F_n$ has to be positioned and oriented such that the needle tip is normal (90°) to the $F_{tis}$ plane (Figure 3.4).

Tissue Deformation Considerations

Given that the needle tip can orient normal to the surface of insertion ($F_{tis}$ in Figure 3.4), and the needle can rotate about its centrepoint axis ($F_n$) on a fixed plane, one can expect minimal shear and longitudinal deformation forces (assuming no friction between the needle tip and tissue is present) on simulated/real tissue during automated suturing[25].

3.1.2 A Typical Robotically-Guided Stitching Task

The stitching task can be described by a succession of simple movements (Cao et al. 1996), as illustrated in Figure 3.5. It is assumed the surgeon selects a series of entry $I$ and exit $O$ stitch points for the suturing task.
1. **Needle-Tool Deployment**: The surgeon manually inserts a curved needle in the needle driver, ensuring the needle is oriented at 90° relative to the tool-tip base and the non-biting end of the tool coincides with $F_{nt}$. This is important, as the kinematics of the robotic system assume that the tool tip frame ($F_{nt}$) and the needle centre frame ($F_n$) have the same orientation (see Figure 3.3). Furthermore, not ensuring the needle is orthogonal to the tool-tip will introduce shear deformations in the tissue, as the needle will be rotating off-plane during insertion.

2. **Approaching $I_i$**: The needle bite tip has to reach $I_i$ while avoiding surrounding critical structures, and approach $I_i$ in the tissue region of interest. It is assumed the abdominal cavity in which laparoscopic suturing is taking place is inflated with CO$_2$, so the local tissue ROI should be free of critical structures and an approach should be possible.

At $I_i$, the needle biting tip has to be aligned to the surface normal (at 90°), so longitudinal deformations are minimized.

3. **Piercing about $C_i$**: The needle rotates about its centre point axis, $C_i$, on a fixed plane, passing the needle tip along a circular curvature (of needle radius, $R$) from $I_i$ to $O_i$ (assuming $I_i - O_i <= 2R$). This is the ideal insertion trajectory for thick tissue anastomosis [27], and minimizes shear tissue deformation.

After the tool tip passes $O_i$ from needle centrepoint rotation, the tool tips release the needle and pulls back.

4. **Reaching $O_i$**: The tool approaches the biting tip of the needle with open jaws, and moves along the curvature of the needle until it makes contact with exit point $O_i$. 
5. Pulling Out Of $O_i$: The tool iteratively pulls the needle out by $\delta \theta$, loosens grip, goes back $-\delta \theta$, re-grabs, and pulls out $\delta \theta$ until 50% of the needle ($90^\circ$, out of a total $180^\circ$) is exposed on the exit end. The tool then vertically pulls the needle and suture thread out of the tissue (i.e along the z axis of $F_w$ in Figure 3.2), until the exposed end of the suture thread is close to $I_i$.

6. Suture Length Exhaustion: The tool vertically pulls up (along the z axis of the world frame $F_w$) to simulate suture length exhaustion.

7. Approaching and Piercing At $I_{i+1}$: The needle bite tip approaches $I_{i+1}$ similar to $I_i$ (Step 2), but rotates an additional $90^\circ$ to ensure contact with $I_{i+1}$ (this is a by-product of pulling out and holding the needle at the centre, not at its non-biting end). The needle then pierces the tissue, rotating about $C_{i+1}$ (similar to Step 3) until half the needle is within the tissue.

The methodology presented above describes a continuous side-to-side anastomosis procedure, but can be adapted for tubular stitching (end-to-end or end-to-side) by tweaking needle approach.

Suture tracking and suture management are not explicitly addressed in this project due to vision system limitations (See Section 8.1.1 for more discussion).

3.2 System Architecture

3.2.1 System Setup

Figure 3.6 illustrates the system setup for the automated robotic suturing tool.

Multi-Purpose Surgical Tool Motor Mount

A dual-motor driver mount actuates a needle driver with passive alignment jaws with axial rotation and grasping functionality.

Tissue Pad Phantoms

A silicone tissue pad is used to provide a controlled (and repeatable) testing environment for automated suturing validation.

Olympus Endoscopic Camera System and Adjustable Mount

An Olympus Exera II CLV-180 Endoscope is used to visualize internal procedures through a trocar on the dome. This is used for suture needle tracking and real-time surface tracking, so the surgeon and robot can visualize feature points used for autonomous robotic guidance during an suturing. The adjustable mount is designed to move the camera around the dome, so any major obstruction in the surgical scene (i.e. the entire needle being blocked by the tool or tissue pad) can be circumvented.

There are three major subsystems - the Mechanical Subsystem, Vision Subsystem, and Trajectory Generation Subsystem - that drive feature acquisition, geometrically-constrained trajectory processing and low-level robotic joint control to realize an automated suturing task.

Mechanical Subsystem

The Mechanical subsystem deals with constrained joint-level Denso control, and tool-tip manipulation for automated suturing. Joint trajectories and tool-tip commands are sent to the Quanser Controller and EPOS-Maxon Tool Motor APIs to actuate a 7-DOF system in real-time.
Figure 3.6: Experimental Setup Integration. A Denso VP-6 Robot laparoscopically operates through an optimally-selected trocar (which maximizes reachable manipulator workspace). An Olympus Exera II CLV-180 Endoscope visualizes the surgical scene, relaying surface feature and needle information for vision processing and trajectory generation. The mechanical subsystem relays these motion trajectories to the robot through low-level (Quanser and EPOS-Maxon) joint control. This allows for autonomous curved-needle suturing through the tissue pad phantom, by way of complex insertion, pick-up and realignment motions with the laparoscopic tool. **Note:** Trocar-based operation was not implemented (no dome present) in the final iteration of automated suturing due to dexterity limitations with a rigid tool.
Vision Subsystem

The Vision subsystem intelligently interprets circular features and objects, such as surface markers and a suturing needle in Open CV, and uses perspective geometry for depth estimation and infer an object’s pose in 3D space.

Trajectory Generation Subsystem

A Trajectory Generation Subsystem serves as a communication bridge that processes 3D information from the vision subsystem and generates a set of geometrically constrained joint trajectories that the mechanical subsystem can utilize to actuate the surgical tool and Denso system for an automated suturing.

The Olympus endoscope relays surface feature and needle images for vision subsystem processing. The trajectory subsystem intelligently combines this information to generate joint-level insertion, pick-up and realignment needle trajectories for robotic guidance. These are then fed to the mechanical subsystem, which actuates the DENSO VP-6 robot and a custom needle driver to autonomously suture through a tissue pad phantom.

The robot is geometrically constrained to operate through a virtual RCM trocar, optimally selected to maximize manipulator workspace dexterity and reachability. For the task of laparoscopic suturing, pose constraints imposed by the RCM prevent optimal suture needle insertion and result in tissue tearing. For this reason, RCM virtual fixtures are removed for validating the automated suturing task.

3.2.2 Workflow Diagram

Figure 3.7 details a workflow diagram for an automated suturing procedure.

1. **Entry $I_n$/Exit $O_n$ Point Pair Selection**: The surgeon begins by selecting entry $I$ and exit $O$ suture points on a 2D image of the suture scene.

2. **Surface Feature Pose Estimation** Surgeon selected points are fed to the Vision subsystem, which then tracks 3D circular features on the surface of the tissue, and interpolates surgeon-selected points relative to these features.

   - Pose information for surface features are obtained via an ellipse-to-circle back projection algorithm. This algorithm exploits known geometric parameters (such as the radius of a circular object) to find a unique solution for elliptical projections of objects that appear in a camera image.
3. **Optimal Centre Point Calculation** \((I_i/O_i \text{ pair})\) Surgeon Selected Points (Converted to 3D equivalent coordinates by the vision subsystem) are then passed to the Trajectory generation subsystem. An optimal centre point (OCP) is calculated for each \(I_i\) and \(O_i\) pair.

4. **Trajectory Generation - Insertion** Inverse kinematics are employed to generate a joint insertion trajectory that rotates the needle about its OCP, to pass it through \(I_i\) and \(O_i\).

5. **Quanser Communication - Needle Insertion**: Joint trajectories are then relayed to the Quanser robot controller, which directs the robot tool-tip (and the suture needle) to physically suture through the tissue for insertion.

6. **Needle Feature Pose Estimation** The tool releases the needle, and moves out of the surgical scene, so the suture needle pose can be obtained. Shape estimation and pose acquisition for the needle utilizes the same ellipse-to-circle back projection algorithm used for surface features.

7. **Needle Centrepoint Extraction/Verification** To verify needle centre coordinates are extracted accurately, the tool touches this point in physical space. Note: This is an optional validation step and isn’t always utilized in a standard automated suture sequence.

8. **Trajectory Generation - Pick Up** A pick-up trajectory (at \(O_i\)) is generated from needle pose data, with the robot tool-tip moving along the needle curvature to approach \(O_i\).

9. **Iterative Pulling Out The Needle** The tool then iteratively pulls the needle out through release/regrab sequences, until the midpoint of the needle is grabbed.

10. **Trajectory Generation - OCP \((I_{i+1}/O_{i+1} \text{ pair})\) and Realignment** A new optimal centre point is calculated for the next entry/exit pair, based on surgeon selected points. The tool realigns to this entry point and rotates about the new OCP to insert the needle through the tissue.

See Figure 3.5 for information on insertion, pick-up and realignment phases.

### 3.2.3 Subsystem Communication Protocol

The Vision subsystem, Trajectory generation and Quanser Robot Control module (in the Mechanical subsystem) relay data to each other via network socket protocols (Figure 3.2.3). Communication is broken into two distinct channels - one for vision acquisition and processing (extracting surface features and needle pose) and the other for robot control (geometrically-constrained joint trajectory generation).

This separation allows the system to potentially improve processing performance by decoupling vision processing and robotic control to two different PCs. For this iteration of the project however, vision and robotics control are driven through socket protocols within the same PC.
Vision Communication Channel

The vision communication channel (brown, Figure 3.2.3) provides surface feature and needle pose acquisition from the Vision subsystem (vision server) when the Trajectory Generation Subsystem (vision client) requires new/updated pose data. The WinSock API facilitates high bandwidth TCP/IP communication for transferring 3D (x-y-z) data points (position and orientation vectors for needle/surface features) through a standard socket. Throughput is sufficient, as the channel operates at standard Ethernet speeds of 1 GBps.

Robot Communication Channel

The robot communication channel (purple, Figure 3.2.3) relays joint-level trajectory generation data from the Trajectory Generation subsystem (robot server) to the Quanser Robot Controller (robot client), which moves the robot to a new set of joint values (when they are fed through the channel).

The Quanser Socket API facilitates high bandwidth TCP/IP, Quanser Compatible, communication for transferring $n \times m$ MATLAB matrices for Simulink-based robot control. Similar to the vision communication channel, throughput is sufficient, as the channel operates at standard Ethernet speeds of 1 GBps.
Figure 3.8: Communication Protocol Layout for Subsystem Components
Chapter 4

Mechanical Subsystem

Figure 4.1 provides a block diagram for the mechanical subsystem. Optimal trocar-constrained RCM restricts motion around a surgical port (to simulate laparoscopic operation), the passive needle alignment tool/EndoPath actuates the needle with rigid geometry (to aid in kinematics and vision), and the 7-DOF Inverse Kinematics model drives the robot in joint space for precise needle tip control and singularity avoidance.

4.1 7-DOF Inverse Kinematics For Needle Tip Manipulation

4.1.1 Background

In order to facilitate formulation of a 7-DOF inverse kinematics model optimized for robotic suturing, Kinematics modelling and linear Control schemes need to be defined adequately.

Forward Kinematics

Kinematics deals with the geometrical and time-based properties of manipulator motion, without regard to the forces and torques that cause it (i.e. dynamics). These include the position, velocity, acceleration, and all higher derivatives of the position variables [9].

A serial manipulator is as a set of rigid bodies (links) connected together in series by motor-actuated joints. Joints are either revolute (rotational) or prismatic (sliding).
MECHANICAL SUBSYSTEM

Figure 4.1: Functional Components of The Mechanical Subsystem: 1. A custom 7-DOF inverse kinematics model for joint-level control, 2. a passive alignment tool for needle actuation and 3. an optimal trocar-constrained RCM for laparoscopic robotic operation.
Forward kinematics uses equations describing various joint frames of the robot to compute the position and orientation of the end-effector, from joint value parameters \cite{9}. This is mathematically described by a rigid transformation matrix $T$:

$$
T = T_0^0 = T_1^0 \cdot T_2^1 \cdot \ldots \cdot T_n^{n-1}
$$

(4.1)

for an $n$-degree of freedom (DOF) serial manipulator. Each rigid transformation matrix, $T_{i-1}^i$, characterizes the relative movement allowed at joint $i$, along with the dimensions of the link from frame $i-1$ to $i$. Denavit and Hartenberg devised a systematic procedure to compute these rigid transformation matrices, by a set of parameters (DH Parameters) that characterize link lengths and joint type (revolute/prismatic), describing them through a coordinate transformation between frames $i-1$ and $i$ \cite{9}.

**Inverse Kinematics**

Inverse kinematics deals with solving the converse problem to forward kinematics - for a given position and orientation of the end-effector (tool), what are the corresponding joint angles to achieve the desired configuration? That is, for a given $T_n^0$, find the corresponding set of joint values $\theta_1...\theta_n$.

For a manipulator with $m$ degrees of freedom in an $n$ degree of freedom workspace, where $m \leq n$, it cannot attain all positions and orientations. For a 6+ DOF manipulator, however, all position and orientation configurations can be achieved in 3D space, sometimes with multiple solutions for joint values. A solution which minimizes the amount each joint is required to move is selected \cite{9}.

**Solving For Joint Parameters In A 6-DOF Serial Manipulator**

For a 6-DOF robotic manipulator, we have 12 equations and 6 unknowns in $T$:

Of these 12, 9 correspond to the translation matrix component of $T_6^0$ ($r_{11} - r_{33}$), of which only 3 are independent. These, combined with the 3 independent equations corresponding to the translation-component of $T_6^0$ ($p_x, p_y, p_z$) give 6 equations and 6 unknowns (Equations 4.2 - 4.5):

$$
\begin{align*}
    r_{11} &= c_1[c_{23}(c_4c_5c_6 - s_4s_5) - s_23s_5c_6] + s_1(s_4c_5c_6 + c_4s_6) \\
    r_{21} &= s_1[c_{23}(c_4c_5c_6 - s_4s_5)] - c_1(s_4c_5c_6 + c_4s_6) \\
    r_{31} &= -s_23(c_4c_5c_6 - s_4s_6) - c_23s_5c_6 \\
    r_{12} &= c_1[c_{23}(-c_4c_5s_6 - s_4c_6) + s_23s_5s_6] + s_1(c_4c_6 - s_4c_5s_6) \\
    r_{22} &= s_1[c_{23}(-c_4c_5s_6 - s_4c_6)] + s_1(c_4c_6 - s_4c_5s_6) \\
    r_{32} &= -s_23(-c_4c_5s_6 - s_4c_6) + c_23s_5s_6
\end{align*}
$$

(4.2)
\[ r_{13} = -c_1(c_{23}c_4s_5 + s_{23}c_5) - s_1s_4s_5 \]
\[ r_{23} = -s_1(c_{23}c_4s_5 + s_{23}c_5) - c_1s_4s_5 \]
\[ r_{23} = c_{23}c_4s_5 - c_{23}c_5 \]

\[ p_x = c_1[a_2c_2 + a_3c_{23} - d_4s_{23}] - d_3s_1 \]
\[ p_y = s_1[a_2c_2 + a_3c_{23} - d_4s_{23}] - d_3c_1 \]
\[ p_z = -a_3s_{23} - a_2s_2 - d_4c_{23} \]

Unfortunately, these equations are nonlinear, transcendental equations that are difficult to solve. Proposed manipulator solution strategies are divided into two broad categories: **closed-form solutions** and **numerical solutions** [9].

**Closed Form Solutions**

Pieper’s theorem states that closed-form solutions exist for a 6-DOF manipulator, provided that three consecutive axes (i.e. the frames corresponding to three consecutive joints) intersect at a common point [9] (see Figure 4.2). In practise, most industrial manipulators with six degrees of freedom usually have the last 3 *wrist* (dictating orientation) joints intersecting. Pieper’s theorem broadly applies to prismatic and revolute manipulator configurations [9].

When the last three axes intersect, the origins of joint frames 4, 5 and 6 are all located at this point of intersection. This allows us to decouple the first 3 joints from the last 3, expressing them as a function of the intersecting origin coordinates, \( P_0^4 \):

\[ P_0^4 = T_1^0 T_2^1 T_3^2 P_3^3 = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \]

if \( g_1, g_2 \) and \( g_3 \) denote the x, y and z components of \( P_0^4 \), and we denote \( r \) as the square magnitude of the three components \( (r = g_1^2 + g_2^2 + g_3^2) \), then, along with the z component equation \( g_3 \), we can eliminate \( \theta_1 \) dependence and solve for a system of two equations, in terms of joints \( \theta_2 \) and \( \theta_3 \). \( \theta_1 \) can then be obtained by substituting into the \( r \) or \( g_3 \)

Now, since \( \theta_1 \) to \( \theta_3 \) are known, the last three joints values \( \theta_4 \) to \( \theta_6 \) can then be solved by:

\[ R_6^4 = R_4^0 \cdot R_6^0 \]
where $R_0^{-1}$ represents a known system of equations *only* in terms of $\theta_1$ to $\theta_3$ and $R_0^0$ is given (the desired position and orientation). $\theta_4$, $\theta_5$ and $\theta_6$ can be solved by the Z-Y-Z Euler angle formulation [9]. Note: there are always two solutions for each of the last three joints, so the ones closest to the previous known configuration (which minimizes change in joint values) are selected.

For real time applications where inverse kinematic computation at fairly high rates are desired, look-up tables for the $\tan^{-1}$ transcendental function are often used [9]. Furthermore, when possible, sets of joint angle equations that are linearly independent can be solved in parallel via multi-threading [9].

**Numerical Solutions**

An alternative (commonly used) solution to the inverse kinematics problem is through the Newton-Raphson method[9]. That is, for a function $f(x)$, its roots are iteratively solved through a recursive function:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (4.8)$$

$x_{n+1}$ is used to approximate the true root (zeros) of the function $f(x)$, and is used as the initialization condition for the next cycle until $f(x_n) = 0$.

For our project, closed form solutions are employed for inverse kinematics computations.

**Control**

Control schemes for robotic applications are important to realize desired motion trajectories in a stable, accurate manner. They can broadly be divided into three categories: **Independent Joint Control**, **Cartesian control** and Multi-variable non-linear control [30].

Independent joint and Cartesian control are commonly employed for slow motion applications. Due to the low-frequency nature of these tasks, the dynamics of the robotic manipulator (i.e. unwanted motion due to the weight/inertia of the links) are ignored, treating any non-linear instabilities as minor disturbances (which are modelled through linear operational modifiers, such as a multiplier, integrator, and/or derivative).

For high frequency, fast motion applications (i.e. pick and place robots in an industrial manufacturing facility), multi-variable non-linear control models account for the dynamics of the robotic system in complex path planning applications, so trajectories are realized accurately and efficiently.
In our application, the suturing task is relatively slow - needle insertion through the tissue has to be an accurate and articulate process, minimizing tissue deformations and ensuring the needle tip exits at a desired surgeon-defined exit point. For this reason, we will focus on design implementations of independent joint and Cartesian control.

**Independent Joint Control**

Independent joint control is the simplest control strategy for driving a manipulator. Each joint axis of the manipulator is controlled as a single-input/single output (SISO) system. Any coupling effects due to motion of other links is treated as a disturbance [30]. Figure 4.3 illustrates a (commonly used) Proportional-Integrator-Derivative (PID) control scheme that achieves zero steady state error, while rejecting disturbances with small (stable) gains [30].

**Cartesian Control**

*Joint-based schemes*

Often, manipulator end-effectors are required to follow straight lines or other path shapes described in Cartesian space. These simple motions are often translated to complex, non-linear motions in joint space, which complicate path planning strategies for robotic applications.

*Trajectory conversion* schemes are often used to convert desired Cartesian trajectories into joint-equivalent trajectories, and then employ joint-control schemes (discussed earlier) to achieve the desired motion [30].

Joint position, acceleration and velocities are computed through these conversion schemes via their cartesian analogs:

\[
\theta_d = InvKin(X_d) \\
\dot{\theta}_d = J^{-1}(\theta)\dot{X}_d \\
\ddot{\theta}_d = J^{-1}(\theta)\ddot{X}_d + J^{-1}(\theta)\dot{X}_d
\]

\(\theta_d, \dot{\theta}_d, \text{and } \ddot{\theta}_d\) represent desired joint, joint velocity, and joint acceleration vectors (for an \(N\)-DOF manipulator) respectively. \(InvKin\) is a trajectory conversion scheme representing an inverse kinematics model, which computes joint values for a desired Cartesian pose \((X_d, \text{which describes the position and orientation of the end-effector}). \ J^{-1}\) represents the Jacobian inverse (which is computed via the pseudo-jacobian inverse method for a redundant manipulator with a non-square Jacobian matrix).

*Cartesian-based schemes*
Figure 4.2: Pieper’s Theorem For Three Intersecting Axes. Left: Link frames for the last three joints (4, 5 and 6). Right: Frames moved along each joint axis to commonly intersect at joint 5.

Figure 4.3: Independent Joint PID Control. $K_P$, $K_I$ and $K_D$ are tuned with the system to achieve zero steady state error.
Alternatively, joint values (which are provided through joint encoders) are converted to Cartesian coordinates via a forward kinematics model. This Cartesian description is then compared to the desired Cartesian position to form an error \( \delta X \). This error is mapped into joint space by an inverse Jacobian, and the resulting error \( \delta \theta \) are then multiplied by gains to compute torques that will minimize these errors (Inverse Jacobian Control) [30].

Another cartesian control scheme, known as Transpose Jacobian Control multiplies the Cartesian error vector by some gain to compute the Cartesian force vector. The Jacobian transpose then maps this to equivalent joint torques, which are minimized to reduce observed errors.

Unfortunately, Jacobian transpose and inverse schemes do not work over the entire workspace, though they can be made stable with appropriate gain selection (and some form of velocity feedback).

Furthermore, Joint and Cartesian based control schemes have some form of coordinate conversion process (position to joint value parameters, or vice versa) which is within the control loop. This may affect performance for real-time applications [30].

### 4.1.2 Design Methodology

The VP-6242E Denso robot is driven using Cartesian control, based on an inverse kinematics model (trajectory conversion scheme) of the robot, laparoscopic tool and needle. This custom model allows for accurate needle-tip based positioning and rotation about the needle centre plane. This on-plane rotation minimizes shear deformations and allows for smooth needle insertion through tissue during an suturing procedure [25].

The Denso robot provides 6 degrees of freedom (DOF), and the tool provides an additional rotational DOF. 7 DOFs are redundant in an unconstrained 3D workspace (3 Position DOFs, 3 Orientation DOFs), so all roll motions are offloaded to rotational Joint 7 (on the tool), thereby relieving Joints 1-6 on the Denso and circumventing potential singularities. This enables the robot to dexterously operate within its reachable workspace during a suturing procedure.

Using DH parameter formulation [9] (see Appendix A.1), a forward kinematics model was generated. The Robotics Modelling Toolbox (by Peter Corke) was used to verify the correctness of this model. It is a comprehensive robotics package that facilitates 1. DH-Parameter visualization of robot joint/link frames in 3D space, 2. Trajectory Planning visualization and 3. Numerical Inverse Kinematics in MATLAB[8]. DH parameter and trajectory visualization is useful to verify if frames are assigned properly, and if end-effector positions and orientations physically make sense for a variety of configurations.
An inverse kinematic model was then manually computed using standard closed-form techniques [9]. The MATLAB environment allowed us to prototype mathematical models with ease and evaluate the correctness of these solutions through visual validation with the Robotics Modelling toolbox (i.e. for a given pose, joint solutions generated from the inverse kinematics model were validated on a movable, visual DH-parameter representation of the robot).

The forward kinematics and inverse kinematics formulations derived in MATLAB have the last frame positioned at the needle centre, $F_n$ (see Figure 3.3). This allows relatively complex needle motions in Cartesian space to be realized through trivial motions in joint space (i.e. rotation about the needle centre corresponds to a rotation about joint 7, which, through the 7-DOF redundant inverse kinematics model, results in smooth tissue insertion).

During an automated suturing procedure, the needle is grabbed at various points during insertion, pick-up, and realignment, creating variable needle-tip offsets relative to the laparoscopic tool tip. The vision subsystem updates the inverse kinematics model with new needle offsets in real-time, so if the needle is rigidly attached to the tool or in the tissue, needle re-grabbing, positioning and insertion is accurate (i.e. $F_{nt}$ in Figure 3.3 variably changes based on the vision system).

### 4.1.3 Implementation

Due to the real-time performance requirements for a robust automated suturing procedure, the 7-DOF inverse kinematics model formulated in MATLAB was ported and optimized to run in C++. As a point of comparison, one time performance test for single needle insertion (i.e. joint trajectories passing the needle through a pair of surgeon specified entry/exit points) indicated the MATLAB routine taking 5.2 minutes, as opposed to 17 milliseconds with the C++ counterpart.

RobOp, a C++ based kinematics package, computes inverse kinematics solutions numerically (and could have potentially circumvented time-intensive, manual closed-form computation), but due to lack of redundant manipulator support, could not be exploited (its lightweight interface was ideal for real-time applications).

Cartesian control was implemented by integrating the C++ inverse kinematics package in the Trajectory Generation Subsystem, and relaying joint values to the Quanser Client for independent joint-level (PID) control. This works ideally for the task of suturing, as motion has to be slow and accurate, minimizing deformations in the tissue pad during needle insertion, pulling-out, and realigning.


4.2 Passive Needle Alignment Tool

4.2.1 Design Methodology

For autonomous modes of suturing, Kang et al. described some of the technical challenges in manipulating curved needles with a robotic setup, and adapting a suture shuttle device (the EndoStitch) as an alternative [17]. This device is optimized to rapidly pass straight double-sided needles back and forth, simply mounted to the end of a robotic manipulator and performing majority of the suturing task independently.

Despite issues in robotically manipulating curved suturing needles, they are the accepted norm in standard laparoscopic surgery, more readily available, and cheaper. Furthermore, they are versatile enough to be used in different suturing scenarios.

In an attempt to address consistency in picking up and actuating curved suturing needles, a passive needle alignment tool was developed (Figure 4.4). The design is adapted from a standard laparoscopic needle driver, with concave groves on the bottom tip and a convex surface on the top tip, to ensure an orthogonal lock-in of the needle when the tips close. This ensures that any mathematical modelling and vision algorithms can use this rigid relation to ensure smooth needle insertion through two points (via robotic control). This modified needle driver is mounted onto an existing dual motor driver frame, which rotates and actuates the gripper (opens and closes the tool) (Figure 4.4). The design is similar to tool-tips commercially available by Olympus for auto-aligning needle-based laparoscopic procedures [27].

The 5mm port size of the passive alignment tool offers additional benefits (smaller incisions) over the 10mm Endostitch tool, with the added versatility of manipulating curved needles in the tissue (which allows for a plethora more of suturing procedure to be performed). This tool is designed for 2-0 and 3-0 suture needles, but can be modified to accommodate 5-0 needles by reducing groove size to accommodate smaller curvature profiles.

4.2.2 Results

Initial (manual) experimentation with this tool was quite positive, orthogonally aligning a needle lying on the surface at various starting positions and orientations. The passive alignment modification was designed to accommodate needles with 22mm diameters sizes (for easy segmentation with vision algorithms), but needles with slightly smaller (16-20mm) diameters also performed reasonably well. This makes the tool versatile for various suturing tasks with diverse tissue geometries.
Challenges

Motor-based torque exertion on passive alignment tool jaws was enough to maintain needle alignment during insertion and re-grabbing, but realignment and re-insertion was problematic as the needle would only be rigidly held in place if positioned at the very back of the tool jaws (i.e. the farthest groove back). Due to minor positioning errors in the vision system, this was not always possible, and inevitably, the needle was occasionally grabbed closer to the middle of the tool jaws. This caused slippage of the needle about the groove (therefore violating the 90° rigidity constraint for on-plane needle insertion), causing bad insertion for any entry/exit point pairs following the 1st point pair. The tool jaws cannot handle excessive forces on the pull shaft when closing. Closing tighter was not an option, as the pull shaft could not handle larger stress (before failing). Using a stiffer shaft is also not possible, since holes on the top jaw (connecting to the pull shaft) have little material around them to withstand excessive forces (and could tear as a result).

An Alternate Approach - The EndoPath

To avoid both these issues (and given the fact that the needle would be preloaded, 90° relative to the tool tip jaws), a standard Ethicon Endopath 5mm Laparoscopic Needle Driver was used. The tool tip, while not grooved like the passive alignment tool, provided superior grip anywhere along the tool jaws and could sustain larger torque exertion by the motors on the pull shaft actuating the jaws. This was due to the perforated rough surface design on the top and bottom jaws, providing good traction with stainless steel needles.

Given the fact the surgeon would preload the needle on the Endopath tool (at 90° relative to the jaw), and the fact that the vision system would alter tool pose (and approach) to pick up the needle orthogonally during the re-grabbing phase, any small errors in the 90° needle alignment constraint would be negligible (and the benefit of having a strong tool tip to grip the needle would far outweigh these minor constraint violations).
This being said, the value of the passive alignment tool should not be underestimated, as it helped validate the insertion and re-grabbing steps for the first entry/exit pair of an automated suturing procedure. Mechanical constraints on the tool ensured that the inverse kinematics model facilitated proper rotation about the needle centre.

4.3 Trocar Constrained Operation

4.3.1 Virtual RCM Constraints

Methodology

In clinical practise, robotically guided laparoscopic procedures are either mechanically or virtually constrained about a trocar, which dramatically reduces the available workspace for navigation and operation [19]. Our 7-DOF system is virtually constrained to operate about an RCM point, driven in joint space. RCM is implemented geometrically through a spherical coordinate system (Figure 4.5), determining pitch ($\theta$) and yaw ($\phi$) angles through the insertion depth, $r$:

$$r = \sqrt{x^2 + y^2 + z^2}$$  \hspace{1cm} (4.9)

$$\theta = \arctan\frac{-z}{x}$$  \hspace{1cm} (4.10)

$$\phi = \arcsin\frac{y}{r}$$  \hspace{1cm} (4.11)

The VP-6 Denso operates tools through a 5mm trocar. Figure 3.2 shows the origin of the spherical coordinate system $F_{tr}$ at this trocar, with the parameter $r$ determining RCM pitch and yaw angles.

Results

Extensive tests with software based RCM joint control were performed by operating the tool through various ports around the virtual patient abdomen (transparent dome). The end effector (tool tip) was able to touch desired points within 0.1 mm accuracy and +/- 0.5mm translation about the trocar during movement. Practically, this is reasonable for our surgical application (where mm level accuracies are sufficient). Any errors present in RCM implementation may stem from mechanical inaccuracies with prototyped tools.
4.3.2 Optimal Trocar Placement

Background

In conventional MIS procedures, trocar placement is governed by easy access and enhanced operability and dexterity on the surgical site of interest, while avoiding critical structures [34]. When robotics are introduced, factors such as arm collisions, manipulator dexterity, reachability, and visibility are equally important. Guidelines for robotically assisted endoscopic procedures have been developed to address these additional constraints, but heavily depend on the assumption that external landmarks reflect access (in real-time) to corresponding internal anatomical structures of interest [6]. Surgical simulation platforms were used to validate trocar selection based on these guidelines, but did not account for patient movement between imaging and surgery [34].

Adhami et al. describes an optimization procedure that weighs surgeon defined trocar placement criteria (based on desired internal anatomical access) against an exhaustive list of ports that can maximize reachability with robotically guided tools during surgery [1]. This criteria was evaluated by Trejos et al. in a cardiac trocar placement study, which determined a set of ports that were surgically relevant and robotically compatible (i.e. large range of motion in the operating volume). They an optimization metric
-the global isotropic index (GII) - to minimize manipulator singularities (i.e. robotic configurations which lock the system, or are not physically reachable) [34].

Despite advances in flexible RCM-constrained robotic wrist manipulators and considerations for optimal robotically-compensated trocar ports that maximize reachability within the abdomen, there are several drawbacks with robotic systems.

High cost, extensive training (these systems are operated by surgeons through master-slave setups), and joint limitations (which result in manipulator singularities) [19] make robotic manipulators challenging to use for surgeons, and particularly demanding for an unconstrained kinematic task such as laparoscopic suturing [19].

Methodology

During RCM experimentation, it was observed that depending on trocar of insertion, the robot could touch a varying range of points with the laparoscopic tool. In an effort to quantify reachable workspace as a function of trocar placement, a simulation (based on the same inverse kinematics virtual model used for geometric RCM constraining) was conducted with ports placed throughout the dome. The goal is to find an optimal port which maximizes the reachable volume with the robot tool tip.

The robotic model, while constrained to one of these ports, was tasked to touch points in a square fashion, on the X-Y plane (see bottom of Figure 4.5, right, for axes) starting from the inner edge of the dome. All intermediate points in this square were also touched, to ensure consistency within the workspace. If one of these points yielded an invalid joint configuration (from the inverse kinematics model and RCM constraints), the square size was reduced and the procedure was repeated again.

If all points within a square were successfully touched, then the square height was increased (in the Z-plane) and the procedure was repeated again. All X, Y and Z movements between intermediate points are done in increments of 5mm. Furthermore, ports are restricted from 32-40 degrees in the vertical plane (rotation about Y) and greater than +/-20 degrees in the horizontal plane (rotation about X), as values outside this range experimentally yielded non-existent workspaces (i.e. collision of tool with the robot, or the robot hitting a singularity during trocar insertion).

In this manner, a bounding box was established for each trocar, and ports with the largest workspace volume were identified (this process takes 4-5 minutes). Using the maximum and minimum X,Y,Z values from each box, a Jacobian is calculated, from which a Global Isotropy Index (GII) is obtained [34]. This value describes a robots general tendency to hit a singularity within the reachable workspace (the closer the GII is to 1, the lower the tendency).
Results

4 RCM points with the largest (and identical) workspaces were identified (Table 1, Figure 4.7) from which the point with a GII value closest to 1 was selected. This RCM point was 36 degrees from the vertical plane and 20 degrees on the horizontal plane, lying on the exterior of the dome, with a reachable workspace of 4 x 14 x 3 cm.

<table>
<thead>
<tr>
<th>RY</th>
<th>RX</th>
<th>Xmin</th>
<th>Xmax</th>
<th>Ymin</th>
<th>Ymax</th>
<th>Zmin</th>
<th>Zmax</th>
<th>GII</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>20</td>
<td>0</td>
<td>40</td>
<td>-70</td>
<td>70</td>
<td>0</td>
<td>30</td>
<td>0.554</td>
</tr>
<tr>
<td>34</td>
<td>20</td>
<td>0</td>
<td>40</td>
<td>-70</td>
<td>70</td>
<td>0</td>
<td>30</td>
<td>0.612</td>
</tr>
<tr>
<td>36</td>
<td>20</td>
<td>0</td>
<td>40</td>
<td>-70</td>
<td>70</td>
<td>0</td>
<td>30</td>
<td>0.912</td>
</tr>
<tr>
<td>38</td>
<td>20</td>
<td>0</td>
<td>40</td>
<td>-70</td>
<td>70</td>
<td>0</td>
<td>30</td>
<td>0.716</td>
</tr>
</tbody>
</table>

Figure 4.6: RCM Points with largest reachable workspace. **GII Values here are normalized w.r.t the workspace.** The RCM point with the largest workspace (36,20) (highlighted in yellow) has a GII value closest to 1. Through this point, it is easier for the manipulator to manoeuvre within the 4 x 14 x 3 cm workspace, and has a lower tendency to hit joint singularities. RX, RY in degrees, all max/min values in mm. RX: Rotation about X.

Figure 4.7: RCM points 32, 20 38,20 superimposed on all RCM points tested. RCM points with largest GII in varying intensities of red. Largest GII value (Most red point) is 36, 20.

RCM and optimal trocar placement constraints were imposed for a variety of simple motion tasks with the robot (i.e. following a straight line and tracing the contour of phantom organs through a trocar- see
Laparoscopic Suturing Considerations

RCM constraints and Optimal trocar placement planning were not retained for the final task of automated suturing. The reason for this is because the optimal trocar (36,20) does not facilitate multiple needle approaches for any given position within the workspace (only a reachable workspace analysis was done, not a dexterous one). During automated suturing, approach needs to be varied during insertion, pick-up, and realignment. Wrongly orienting a suture needle (due to geometric RCM trocar constraints) during insertion can cause unwanted deformations in the tissue, resulting in tissue tearing and possibly inaccurate suturing.

Of course, an alternate dexterity and reachability-based optimization task - one that finds a trocar through which the robot can approach with desired insertion/pick-up/and realigning approach angles, while physically being able to reach the suture points - could be pursued. Assuming such a trocar exists, the robot’s reachable workspace may not be as large as (36,20), but dexterous enough to realize the given suturing task.

Due to the scope of this thesis, dexterity-based analysis was not pursued for laparoscopic automated suturing. Instead, a larger focus was given on realizing sub motion tasks for the stitching sequence, with the eventual goal that an RCM layer would be applied on top (to constrain approach, as is conventionally done laparoscopically). This RCM layer would operate through a port that is optimized for dexterity and reachability of the tool tip and needle at their required positions and orientations, for various phases of the suturing task.
Chapter 5

Vision Subsystem

Figure 5.1 provides a block diagram for the vision subsystem. Surface and needle features are captured by the Olympus Endoscope, processed and shape estimated for pose extraction. These are then fed through a calibration module which transform pose coordinates to tool-tip equivalent robot world coordinates.

The primary task of the vision subsystem is to provide 3D (depth) information for robotic guidance during automated suturing. Conventionally, depth in vision is usually obtained with stereo cameras, and triangulating perspective images into a 3D cloud of feature points. However, this method is inaccurate in a surgical context, due to limitations with endoscopic stereovision systems. The millimetre level distance separation between endoscopic stereo lenses results in poor disparity and depth perception. As a result, tracking millimetre level changes in tissue feature deformation and surgical instrument motion (such as suturing needles) is inviable during operation[22]. To compensate for disparity inaccuracies in endoscopic stereovision, another trocar for a second camera instrument (to provide an alternate, unique perspective image), which would introduce unnecessary patient trauma.

The objective is to find an alternative, light-weight algorithm which can provide comparable levels of accuracy as conventional stereo-vision, while increasing computational speed and using existing clinical visualization equipment for robust feature tracking.

Wengert et. al’s monocular pose estimation algorithm (based on the TRIP algorithm) exploits known circular geometries (rigid objects with well-defined radii) and their elliptical projections in a camera image to infer pose information about these objects in 3D space.

Open CV has computationally inexpensive pre-processing feature segmentation filters, and an ellipse-fitting module, which, when combined with Wengert et. al’s pose estimation method, can be adapted to
Figure 5.1: Functional Components Of The Vision Subsystem: 1. Surface Feature And Needle Segmentation/Shape Estimation, 2. Pose Estimation and 3. Distortion Correction (Camera Calibration) and Robot Tool-tip Coordinate Conversion.
estimate the position and orientation of a semi-circular needle (9mm radius) and surface features (3mm green stickers, glued onto the tissue surface) in 3D Cartesian space. This gives depth context to the tissue environment being sutured on, estimates needle pose, and facilitates accurate robotically guided suturing.

5.1 Background - Filters For Feature Segmentation and Shape Estimation

In order to ensure surface features and the needle are robustly and accurately segmented for ellipse fitting and pose estimation, colour-based segmentation, background subtraction, image smoothing, and skeletonization filters need to be employed to give noise-free, accurate binary images of the features being tracked.

Shape estimation techniques are employed once features are well segmented in an image. Line, circle and ellipse estimation methods are used in this project for masking applications (to discriminate specific features) and shape fitting applications for pose estimation.

Colour Image Segmentation

Colour Space Properties

Colour image segmentation is the process of extracting one or more connected regions that satisfy a uniformity criterion (based on features derived from spectral components) [32]. These spectral components are defined by a colour space model.

Three commonly employed color space schemes for vision applications include: RGB, CMY (Cyan-Magenta-Yellow), and HSV (Hue-Saturation Value) [32]. RGB is an additive colour model (creating colours by mixing bands of spectral light in varying combinations), CMY is a subtractive one (retaining colours by only absorbing certain wavelengths of white light while reflecting the rest), and HSV is a cylindrical-coordinate analog of RGB (With hue defining raw colour, saturation indicating the degree of raw colour differentiation from neutral gray, and value indicating the level of illumination).

HSV provides superior transparency, specular (reflection), and shadow invariance [32] and was used for colour feature segmentation of surface features in this project.
Colour Segmentation Algorithms

Colour image segmentation algorithms can broadly be divided into 2 categories: Pixel Based and Area Based Segmentation [32]. Pixel-based approaches segment images based on individual pixel elements, while area-based approaches partition clusters of image pixels, segment the result.

Pixel Based Segmentation

Pixel-based segmentation is broadly divided into histogram and cluster-based techniques.

Histogram Based Techniques

This subclass of pixel-based segmentation identifies one or more peaks, and surrounding areas are used in the pixel classification process [32].

Lin et al. used an HSV colour space model for road following applications. The image was separated into road and non-road regions based on distinctive histogram peaks, from which a segmented (discontinuous) road could be identified. They compared the algorithm to histogram partitioning in RGB space, and found the RGB-based model became unstable when shadowed parts of the road were present. The HSV-model, on the other hand performed robustly and consistently to these variations [32].

Segmentation By Clustering Data

This subclass of pixel-based segmentation collects like pixel values into groups (randomly, or through a heuristic like Lolyd’s algorithm [32]) , which are then used in the pixel classification process [32]. Practically, this cluster segmentation process is equivalent to colour thresholding [32].

Umbaugh et al. used normalized RGB/HSV colour space models for automatically identifying skin tumour features based on cluster-data colour segmentation. Pixels are classified by minimum distance to single representative of classes. These classes are identified by a median splitting process (i.e. identifying a box which encompasses pixel clusters) [32].

Area Based Segmentation

Area-based segmentation is divided into two main groups: Region growing, and split-and-merge [32].

Region Growing

This subclass of area-based segmentation uses a (given) basic set of uniform regions (seeds) and various strategies are employed to join surrounding neighbourhoods.
Ismaili et al. used region-growing methods and regression analysis in the RGB and HSV space for automated removal of fluid during an endoscopy. Small uniform blocks are defined by recursively tuning histogram and regression parameters. These are merged in a quadtree (see Fast Hough Transforms, described in Shape Estimation) structure [32] to discriminate fluid from other similar features.

**Split and Merge**

This subclass of area-based segmentation starts from nonuniform regions, subdivide them until uniform regions are obtained (split), and apply heuristic methods to fit them to the maximum uniform area (merge). The maximum uniform area corresponds to the area sum of all nonuniform regions (i.e. the original image).

Panjwani et al. used Markov random field models in RGB colour space to automatically segment textured colour images. Non-uniform fixed size blocks were split until uniform blocks were obtained, and agglomerative clustering (employing Gaussian Markov random fields) was used to conservatively re-merge the data into quasi-uniform clusters [32].

**Background Subtraction Filters**

Background subtraction filters are used in this project for accurate needle segmentation, so pose estimation information can be reliably used for robotic guidance. For a given frame sequence (obtained from a fixed camera), background filters takes the difference between the current frame and a static background image, and thresholds the result. This shows which areas of the picture changed:

\[ |frame_i - background_i| > T_h \]  \hspace{1cm} (5.1)

The background image is sensitive to:

1. Illumination changes
   
   (a) Gradual
   
   (b) Sudden

2. Motion changes
   
   (a) Camera Oscillations
   
   (b) High-frequency background objects (such as tree branches moving with the wind, waves, etc)
3. Changes in background geometry (i.e. parked cars)

and as a result, $T_h$ in Equation 5.1 needs to be tuned to filter out these inconsistencies during a background difference operation (although, this is not always possible).

Background subtraction filters can be broadly classified into **predictive** and **non-predictive** models.

**Predictive methods** model the scene as a time series and develop a dynamic model to recover the current input based on past observations. The magnitude of the deviation between the predicted and actual observation can then be used as a measure of change.

**Non-Predictive, density based methods** neglect the order of observations and build a probabilistic representation of the observations at a particular pixel.

Both predictive and non-predictive models were evaluated in this project for effective needle segmentation.

### Predictive Models

**Running Gaussian Average**

Wren et al. proposed a running average technique to model the background independently at each $(x,y)$ pixel location [23]. A Gaussian is fit to a histogram from the first image frame, and a running average iteratively adjusts $\mu_t$ for subsequent frames:

$$
\mu_t = \alpha \times I_t + (1 - \alpha) \times \mu_{t-1} \tag{5.2}
$$

where $I_t$ is the pixel’s current value and $\mu_{t-1}$ previous running average. $\alpha$ is the empirical weight chosen as a trade off between stability and quick update. This running average can then be used to classify a pixel as a foreground pixel if it satisfies the following inequality:

$$
|I_t - \mu_t| > k\sigma_t \tag{5.3}
$$

Otherwise, it is a background pixel.

Koller et al. indicated that $\mu_t$ in equation 5.2 gets unduly updated (i.e. error accumulation) when foreground values are present (which will skew the classification process in equation 5.3 and prevent robust foreground segmentation). To compensate, Koller updated the model [23] to:

$$
\mu_t = M\mu_{t-1} + (1 - M)(\alpha I_t + (1 - \alpha) \times \mu_{t-1}) \tag{5.4}
$$
where the binary value $M$ corresponds to 1 if $I_t$ is a foreground value, and 0 otherwise. This selective backgrounding keeps the model robust to a range of foreground pixels, than biasing it towards one end of the range (as was the case with equation 5.2).

**Mixture of Gaussians**

For high frequency changes in the background scene (i.e. leaves on a tree, snow, rain, waves), Mixture of Gaussians is a robust background subtraction algorithm that provides superior results to the gaussian running average method (which yields an incorrect $u_t$ for high frequency changes in the scene).

Stauffer and Grimson proposed a probabilistic model to describe observing a pixel value, $x$, at time $t$ by mixing $K$ Gaussian distributions [23]:

$$P(x_t) = \sum_{i=1}^{K} \omega_{i,t} \times \eta(x_t, \mu_{i,t}, \sum_{i,t}) \quad (5.5)$$

Where $K$ is the number of distributions, $\omega_{i,t}$ is an estimate of the weight (what portion of the data is accounted for by this Gaussian) of the $i^{th}$ Gaussian in the mixture at time $t$, $\mu_{i,t}$ is the mean value of the $i^{th}$ Gaussian in the mixture at time $t$, and $\eta$ is a Gaussian probability density function. $K$ is determined by the available memory and computation power (typically a value of 3-5 is used [23]).

The discrimination between the foreground and background are achieved as such: first, all distributions are ranked based on the ratio between their peak amplitude, $\omega_i$, and the standard deviation, $\sigma_i$. The assumption is the more compact and narrow the distribution, the more likely it is to belong to the background. Then, the first $B$ distributions in ranking order satisfying

$$\sum_{i=1}^{B} \omega_i > T \quad (5.6)$$

where $T$ is the assigned threshold, are accepted as background[23].

At each time frame $t$, two problems must be simultaneously solved: a) assigning the new observed value, $x_t$, to be the best matching distribution, and b) estimating the updated model parameters (from equation 5.5). These problems can be simultaneously solved by employing an expectation-maximization algorithm (EM)[23]. However, since this algorithm is computationally costly, the matching is approximated by all distributions satisfying:

$$\frac{(x_t - \mu_{i,t})}{\sigma_{i,t}} > 2.5 \quad (5.7)$$
The first in ranking order is accepted as a match for $x_t$. Only then are the Gaussian parameters (for $\eta$ in equation 5.5) updated for these matching distributions. If no match is found, the last ranked distributions are replaced by a new one centred in $x_t$, with low weight and high variance.

Non-Predictive Models

Kernel Density Estimation

Elgammal et al. proposed a non-parametric density based model, known as the Kernel Density Estimation (KDE) model which guarantees a smoothed, continuous version of the histogram of a background image [23]. The purpose of this background model is to capture recent information about the image sequence, continuously capturing and updating fast changes in the background scene.

For a given sample pixel intensity, $x_i$, the probability that this pixel will have an intensity $x_t$ at time $t$ can be non-parametrically estimated with a kernel estimator $K$ by [23]:

$$P(x_t) = \frac{1}{n} \sum_{i=1}^{n} K(x_t - x_i)$$ (5.8)

If $K$ approximates a Normal distribution, and independence amongst the red, green and blue channels are assumed, then one can use equation 5.8 to estimate if a pixel is a foreground pixel if $P(x_t) < T_h$, where $T_h$ is a global image threshold that can be adjusted to achieved to minimize false positives (which may appear due to noise).

To quantitatively determine $T_h$ and minimize false positive detection rates, Elgammal et al. created a short term and long term model (by changing sample size) of equation 5.8, comparing results, and taking the intersection of the two detection results [23]. True positives in the short term model may be suppressed by the long term model, but if an adjacent constraint is imposed - i.e. intersection pixels augmented by short-term true positive pixels - then this issue is circumvented [23].

Ultimately, a predictive, running Gaussian average background subtraction method was utilized for needle segmentation, due to relatively static background conditions.

Smoothing Filters

Smoothing filters are vital in removing background noise and other high frequency components, which is useful for shape estimation algorithms (such as ellipse fitting). The following discussion examines a variety of image smoothing techniques that were investigated for this project, weighing robust filtering benefits (i.e. maintaining detail while removing noise) versus computational expense.
Gaussian Smoothing Filter

The Gaussian smoothing operator convolves a Gaussian kernel with a 2D image to produce a "blurred" image - one that removes detail and noise. A kernel (a discrete element) of width $\sigma$ specifies the sampling interval for generating a discretized Gaussian function (from its continuous counterpart).

The convolution is relatively simple and computationally inexpensive, making it ideal for real-time applications [14]. The 2D kernel is separable - meaning the image can first be convolved with a 1D horizontal filter, and then subsequently convolved with a 1D vertical filter. Computationally, this equates to $2k$ operations per pixel (where the Gaussian kernel is $k \times k$ in size), compared to $k^2$ in a standard 2D convolution scenario [14].

Apart from time performance advantages, the Gaussian filter is rotationally symmetric (which is ideal for object tracking applications where pose is constantly varying) [14]. Furthermore, the size of the kernel can be tuned to ensure edges of an object are retained in an image.

Bilateral Filter

The bilateral filter employs a pixel-based approach to smooth the image while preserving its edges - for any given pixel, it takes a Gaussian-weighted average of neighboring pixels. As a result, it is used in noise reduction and HDR compression applications [14]. The weighted averages are not only based on Euclidean distances, but also on radiometric differences (such as colour intensity) [14]. However, Gaussian-weighted averaging is unstable when there are few similar neighboring pixels. Furthermore, extremely sharp edges lose detail as they are smoothed to some degree. To address the latter, Chaudhury et al. proposed a new "trilateral" filter which tilts and skews the filter window using bilateral filtered image gradients, but at the cost of high computation [14].

From a real-time performance perspective, the bilateral filter is computationally inefficient (operating in $O(n^2)$ time) as kernel size is increased. However, recent work by Adams et al. suggest faster filtering algorithms operating on colour images, based on histogram data [14]. These trade off increased filter speed for image quality degradation.

Robust Smoothing Filter

This filter is a simple and fast nonlinear filter that, like the bilateral filter, employs a pixel-based approach to preserve edges. However, unlike the bilateral filter, it does this 1) computing the maximum
and minimum gray values of neighbouring pixels and 2) comparing the gray value of the centre pixel to
the maximum/minimum.

If the centre pixel gray value is larger than the maximum, the maximum is the output. If the gray value
is smaller than the minimum, the minimum is the output, and for anything in between, the gray value
is the retained [14].

A Gaussian smoothing filter was ultimately employed for surface feature and needle segmentation shape
pre-processing, due to rotationally symmetry (for a given object), and time-performance advantages,
and no need for edge detail preservation.

**Skeletonization**

Skeletonization is the process of *thinning* an object, while preserving shape topology and component
connectivity between adjacent regions [7] (Figure 5.2). Mathematically, this operation is described by a
**medial-axis transform** (MAT), which extracts the centreline of the object. This transform is extremely
useful for shape reconstruction and motion planning applications in computer vision [7], as the user can
extract core structural descriptors of an object (by abstracting thickness away), and use it for tracking
and pose estimation. In our application, skeletonization was used to thin a segmented needle image
(obtained from background subtraction) for accurate ellipse shape estimation.

![Skeletonized Image](image)

Figure 5.2: Skeletonized Image. The white centreline and branch structures protruding from it preserve
the structural features of the original “B” shape, which can assist in shape estimation algorithms.

For a given binary image, a medial-axis transformation reduces foreground regions (of a segmented
object) to a skeletal remnant (thin, connected regions). As an analogy, if one assumes the foreground
pixels are made of slow, uniformly burning material, and a fire is lit simultaneously on all parts of the
segmented object, the fire burns away toward the interior. At points where this fire is travelling from
two different parts of the segmented object and intersect, the fire is extinguished and a *quench line* is
formed [5]. This is the centreline skeleton of a segmented binary object.
Conventionally, a medial-axis transform is applied on a continuous polygonal object. With a digitized image however, using a geometric approach will usually fail, as pixel-based segmented objects are non-continuous, discrete objects that yield visually incorrect inconsistencies in the output skeleton image. To account for this, digital medial axis operators, such as the distance transform, address skeletonization from a pixel-based approach. The algorithm iteratively progresses toward the centreline of continuous binary pixel clusters, until the remaining sections are one pixel wide.

Discretized MAT - Singular Solutions To A Distance Transform

A distance transform converts a binary image to a grayscale image, with intensities corresponding to distances of internal points to boundary points. The points along the centreline correspond to curvature discontinuities and are singular solutions to the distance transform (Figure 5.3). Correspondingly, these solutions lie along the skeleton from a medial-axis transformation.

For real-time performance applications, distance-transform based solutions perform reasonably well (linearithmic, in $O(n \log n)$ time) [2] but are extremely sensitive to noise [5]. Furthermore, the extremities of the skeleton drop in intensity, relative to the skeletal centre.

Recursive Open Operations for Contrast and Noise Invariant Skeletonization

An open operation is a mathematical morphological operator defined as the dilation of an eroded image [11]. A common structuring element is used for both erosion and dilation operations. This element describes the shape and size of the morphological operator being applied to an image, and can affect the
resolution (coarse/fine) and isolate desired features of interest (i.e. using rectangular versus elliptical structuring elements).

In the context of skeletonization, recursively applying the union of an erosion and open morphological operator on an image eliminates noise (1-pixel wide), while robustly providing uniform, high contrast skeletons of segmented binary objects [11]. Recursive discrete morphological skeletonization is computationally inexpensive (performing in linear $O(n)$ time) [11], robust to noise (recursive thinning removes any pixel clusters that are less than 2 pixels wide) [11] and contrast invariant (i.e. a binary skeleton, unlike a grayscale skeleton obtained from a discretized medial-axis transform). As a result, this method was employed for needle skeletonization in our application.

**Skeletal Pruning**

Skeletonization is an effective shape-preserving, thinning method for shapes with simple geometry. However, when more complex objects are thinned, several redundant branches also appear [5]. This occurs due to minor noise and object boundary deformation, and introduces unnecessary (added) skeletal topological information and incorrectly skews shape estimation.

To compensate for these inconsistencies, skeletal pruning is utilized [5]. Pruning attempts to remove redundant branches in a skeleton, and give a cleaner centreline. In practise, however, vital topological information is also removed in the process.

Bai et al. attempts to address this issue by using Discrete Curve Evolution (DCE), a contour partitioning method which is robust to noise and yields visually consistent skeletal remnants of complex shape geometries. DCE reduces boundary noise without displacing boundary points, and uses global contour information to generate an accurate centreline representation of an object. By doing this, it removes skeletal instability and variation, and provides a reliable method for shape estimation. The DCE pruning algorithm has a time complexity of $O(n \log n)$ [5], which is reasonable (though not ideal) for real-time applications.

**Shape Estimation For Vision Guidance**

Accurate shape estimation is an important component of vision-guided applications (such as robotics). Shape detection algorithms estimate a "descriptor" (i.e. parameters which describe the shape uniquely) based on features, colour, and texture over a series of images, to build an accurate and concise model [4].
Line, Circle And Ellipse Shape Detection

Lines, Circles and Ellipses are commonly used in many object (pose) tracking applications in computer vision. The following discussion will examine robust (noise, occlusion invariant) estimation algorithms which track these shapes. All three revolve around using variations of the Hough transform.

The Hough Transform (HT) is a feature extraction technique used in vision to find imperfect instances of objects within a certain class of shapes [4]. For each point in image space, there is a corresponding parameter space representation of the point. The idea is to find the union of parameter values (an accumulator array [3]) which uniformly describe all (or most) image space points within a binary image. In line fitting, the hough transform estimates two parameters (the slope $k$ and intercept $c$), while in circle fitting, three parameters are estimated (the centrepoint $(x, y)$ and the radius, $R$).

Ellipse detection through a Hough transform is more computationally expensive (compared to line/circle fitting), due to a large (6 parameter) search space. To address this, least-square fitting techniques for robust parameter estimation have been proposed by a variety of authors as a faster, real-time alternative to Hough transforms [26, 13, 15].

More rigorous discussion of line, circle and ellipse estimation using hough transforms and least squared fitting is presented in Appendices B.1-B.4.

RANSAC - A Computationally Efficient Parameter Validation Technique For Ellipse Detection

Least-squares techniques for ellipse parameter estimation do not always produce ideal results. Often, ellipses are inaccurately estimated due to noise and low pixel information for fitting.

RANSAC (Random Sample Consensus), proposed by Fischler and Bolles in 1981, is an iterative method that estimates parameters of a mathematical model from a set of observed data which contains outliers. It is a stochastic process which produces a reasonable result with only a certain probability, which increases as more iterations are allowed [37].

It is ideal for the task of ellipse fit refinement and validation, as it can efficiently reject outliers and refine parameter estimation (even if they are initially invalid) in a computationally efficient manner, without compromising accuracy[37].

Yu et al. used RANSAC to refine a non-ellipse pixel rejection technique (based on a graph Laplacian) for robust ellipse fitting [37]. A graph Laplacian describes the extent of elliptical pixel connectivity in a
noisy, segmented image. Outliers that are external to the ellipse are filtered by the graph laplacian, but those that are within the ellipse are still retained. RANSAC removes these internal outliers in a recursive fashion, providing a well segmented ellipse image that can subsequently be used by least-squares fitting techniques (LSTs) for shape estimation [37].

Conveniently, RANSAC is supplied with a random set of points for model estimation. It iteratively refines this model until a certain classification criteria is satisfied. In Yu’s implementation, however, all inlier points from the graph Laplacian are used. The revised RANSAC technique operates as follows [37]:

1. Fit a model \( h \) to all inliers from the graph Laplacian
2. Test all data points with respect to \( h \), and any that deviate from \( h \) (above a certain threshold) are classified as outliers and removed from future test cases
3. Refit the model \( h \) based on updated inliers
4. Repeat steps 2) and 3) until the classification (i.e. number of inlier points) does not change anymore

With this modified RANSAC implementation, any further outliers are eliminated, and standard LSTs can then be used to generate an accurate ellipse fit for segmented ellipse pixel data.

**PRANSAC For Noise, Spurious Edge and Occlusion Invariant Ellipse fitting**

Kaewapichai and Kaewtrakulpong proposed an efficient and robust technique that detects ellipses that are highly occluded, noisy and have spurious edges (that are commonly present when using canny or sobel edge detection techniques) in a segmented ellipse image [16]. They implement a PRANSAC model (pseudo RANSAC) which randomly selects edge patches, calculates a *fitness* metric, and compares it against a fitness threshold.

The method first starts by randomly selecting two points in the input data. The distance between them must exceed \( 2r \), otherwise the points are reselected. The number of times, \( K \) random point pairs are selected are determined by:

\[
K = \frac{\log(1 - b)}{\log(1 - (1 - err)^s)}
\]  

(5.9)

Where \( b \) is the probability that at least one random point is free from outliers, \( err \) is the proportion of outliers, and \( s \) is the minimum points for the model fitting requirement [16].
Once an appropriate point pair is selected, all connected edge points to each of the selected points (within radius $r$) are chosen for least squares ellipse fitting. The authors utilize Fitzgibbon’s approach [16].

Then, a *fitness* metric is determined, to verify the parameters of the LST ellipse fit. This value is the ratio of non-occluded edge length to ellipse perimeter:

$$fitness = \frac{N}{P_{\text{ellipse}}}$$  \hspace{1cm} (5.10)

where $N$ is the number of data points with shortest (perpendicular) distances to the ellipse perimeter $P_{\text{ellipse}}$ (less than some distance, $d$). The fitness value is compared against some fitness threshold, $f_{th}$, which determines if the ellipse is acceptable, or not.

Experimental results indicated that the method could accurately trace ellipses that were up to 50% occluded, with almost 50% noise in the image. These were tested against synthetic and real-world images by pre-processing and segmenting noisy, occluded ellipse data (through smoothing, canny edge detection) and then applying their fitness threshold to refine LST fitted ellipse results.

### 5.2 Surface Feature Segmentation

#### 5.2.1 Methodology

Circular features were colour thresholded (pixel-based clustering) by converting an RGB-image to an HSV (Hue-Saturation-Value) image, and filtering out green pixels in the image (employing a pixel-based histogram approach) within the range of: $H=30-120$, $S=0-255$, $V=0-220$. HSV provides a better Signal-to-Noise (SNR) ratio than RGB [32], minimizing lighting and background shadow (from high intensity bright-light) based inconsistencies on the colour circles of interest. The HSV image was smoothed with a Gaussian filter to ensure smooth edges for the ellipse curve fitting module (used later).

Open CV’s ellipse-fitting module can only fit one ellipse for a given set of binary pixel points. Since multiple surface features are present in the image, additional processing is required to segment each feature out, and individually feed them for ellipse fitting.

Open CV’s Hough circle detection module can robustly detect multiple circular (or near-circular) features in a binary image. It is one common application of the more common Hough transform, which is a feature extraction technique that aims to find imperfect instances of objects within a certain class of shapes by
employing a voting procedure [3]. This imperfection works ideally for our application, as the features that are being tracked are elliptical, and the Hough circle module can enclose each feature’s elliptical pixels with an approximate, larger, concise circle. As a result, these circles can serve as a masking method to extract individual features from an image.

The centre point of a detected Hough circle (corresponding to one surface feature) and an arbitrary radius can be used as a local ROI and masked on top of the RGB/thresholded binary image, and fed to the ellipse fitting module for further processing. A large circle pixel radius (65 px) was used to detect all elliptical surface features, so they could be appropriately masked. A robust "tolerance" parameter \( Tl \) - i.e. the goodness of a Hough circle fit to a segmented circle-like feature - was experimentally determined for a variety of a lighting conditions and different tissue pad orientations (the higher this parameter, the less tolerant and more stringent/precise the Hough circle fit is). \( Tl=30 \) robustly encompassed all pixels from each elliptical feature (without allowing adjacent feature pixels to leak in) in a lighting and shift (rotation/translation) invariant fashion (see Figure 5.5).

\[ \text{5.2.2 Results} \]

Figure 5.4 demonstrates robust surface feature thresholding to low and high level lighting conditions.

Figure 5.5 illustrates effective Hough circle masking with a 65 pixel circle size constraint and a relatively low \( (Tl=30) \) tolerance level.

Figure 5.6 shows robustness to a reflective stainless steel surgical tool being introduced in the scene, with effective thresholding, ROI masking, and ellipse approximation accurately.

Together, these results look promising for robust surface feature (light and tool invariant) segmentation in automated suturing.
Figure 5.4: Robust Surface Feature Thresholding. Raw RGB images (top), HSV images (middle), and Binary Threshold Images (bottom) at various lighting conditions (neutral, high, low). Notice the range $H=30-120$, $S=0-255$, $V=0-220$ accommodating varying levels of lighting and producing relatively uniform, well-thresholded images.
Chapter 5. Vision Subsystem

Figure 5.5: Local Feature Detection: A preprocessing step for surface feature ellipse fitting. The circular ROI (left) masks all other surface features and enables feature-specific ellipse fitting for accurate pose estimation.

Figure 5.6: Tool-Invariant Surface Feature Segmentation. Notice pixels from the reflective, stainless steel tool do not leak into the colour thresholded surface feature image, allowing for robust segmentation.

5.3 Needle Segmentation

5.3.1 Methodology

A standard 18mm diameter C012D CR/8 Ethicon Suture needle was used for automated suturing. Segmenting a reflective, grey, stainless steel needle is a challenging task, as direct RGB colour-based (grey colour) thresholding is sensitive to background shadows and lighting inconsistencies.

While RGB-HSV image space conversion was successful in minimizing lighting/shadow inconsistencies in surface feature segmentation, this was primarily due to green colour thresholding and non-inclusion of grey/black values within the thresholding range. In the case of needle thresholding however, since the colour of the needle is grey-black and the shadows are also grey-black, HSV conversion will not help isolate the range of needle pixels needed for ellipse fitting.
Background Subtraction To Remove Non-Needle Background Pixel Elements

To remove background ridge shadows and other undesirable dark non-needle pixel elements, while maintaining a well-segmented image of the needle, predictive, running gaussian average background subtraction was utilized (Figure 5.7).

It was assumed that the background was relatively static, and that needle insertion in the tissue did not deform the surrounding surface significantly. This is a reasonable assumption, as the tissue pad is relatively elastic, so it would deform to its normal state even after needle insertion. High frequency background components, or camera oscillations were not presents, and illumination was relatively constant. As a result, background image subtraction was relatively accurate and robust to any needle position and orientation, and complex predictive or non-predictive models (which would increase computational cost) were no implemented. It is important to note, however, in an in-vivo scenario, the surgical site of operation has blood flowing and tissue deforming (medium-high frequency components), which makes background subtraction unfeasible. Section 7.1 addresses this issue by using colour thresholding (in HSV space) to segment a commercially available matte black curved suturing needle.

Assuming relatively static background conditions, an image without the needle was captured at the beginning of the procedure and a hard threshold at \( V = 252 \) was applied to obtain a binary image of just the ridges (Figure 5.7, 2\(^{nd}\) Panel). This was subtracted from the needle and background ridge image (Figure 5.7, 3\(^{rd}\) Panel) to obtain a well-segmented image of only the needle. This image was then skeletonized/thinned (Figure 5.7, 4\(^{th}\) Panel), and Gaussian smoothed, to aid in better ellipse fitting.

5.3.2 Results

Figure 5.8 illustrates effective background subtraction and high-specificity needle segmentation. Histograms for the background and background with the needle were superimposed on each other, illustrat-
Figure 5.8: High Needle Specificity, based on Subtractive Intensity Based Thresholding. Histograms for Blue: Background, Green: Average Background + Needle (50 sequences), and Red: Needle Only (Green minus Blue). Black-dotted line indicates a hard threshold at $IV=252$, retaining all pixels to the left of the line.

Adding additional pixel power in the latter (cyan), corresponding to needle pixels. The subtracted image (red) clearly shows two humps, which correspond to both sides of the needle. The right most mini-peak corresponds to highly reflective pixel elements in the centre of the needle, which were purposely discarded (as they severely skewed ellipse fitting and were highly variable). For this reason, an intensity value $IV = 252$ for binary images of the background and background with the needle were chosen, to retain the two ends of a well-segmented needle (in tissue).

**Real Time Tracking Considerations For Needle Segmentation**

Currently, the needle is captured when the laparoscopic tool is outside the endoscope camera view, and pose is acquired for accurate needle pick-up at the exit point.
For real-time tracking, needle pose would ideally be captured for error compensation and used as a redundant system to validate kinematics motions of the tool tip, as the robot is inserting, releasing, and picking up the needle at various phases of a suturing procedure. Currently, needle segmentation occurs through background subtraction, which only works well when the needle (and no new object) is introduced into the surgical scene.

This is method is not ideal for real-time applications, as the tool grabs the needle during insertion, and will skew background subtraction (leakage of tool-pixels), inaccurately bias ellipse estimation, and ultimately affect real-time pose estimation of the needle.

Colour-based thresholding methods cannot be employed to discriminate tool from needle, as both are made from stainless steel, and black/grey colour filters used to segment one will most likely results in pixel leakage from the other. To address this, Hough lines (similar to Hough circles) can be used to approximate the tool (which is rigid and described by a set of near-linear features) with a series of lines. This result can serve as a mask to remove tool pixels from the final segmented image (Figure 5.9).

![Hough Lines Approximation Mask For Tool Based Occlusions](image)

While Hough line based tool-masking was not implemented in the current iteration of this project, it is worth noting that with proper masking, tool-based occlusions of the needle still allow for accurate ellipse estimation (Figure 5.10), which can then facilitate real-time tracking of the needle (whether it is grabbed by the tool or in the tissue). For proof-of-concept purposes, the needle in Figure 5.10 was intentionally coloured green to ensure colour contrast from the tool. The final segmented image (Figure 5.10, right) is supposed to mimic thresholded, thinned images of the needle (such as Figure 5.14, 4th Panel), by using masks such as the Hough lines estimator to approximate segments of the tool. Notice that the tool-occluded needle segmented image (Figure 5.10, bottom) still facilitates accurate ellipse fitting. This shows, that in theory, with proper tool occlusion-invariant needle segmentation, real-time tracking of the needle (through subsequent ellipse fitting and pose estimation) is viable for all phases of an automated suturing procedure.
Chapter 5. Vision Subsystem

5.4 Curve Approximation and Ellipse Fitting

5.4.1 Methodology

Well-segmented surface feature and needle binary images are fed to Open CV’s ellipse fitting module for approximating an ellipse, so depth information can be inferred by back-projecting elliptical information to a 3D circular object.

Open CV’s ellipse fitting algorithm (EllipseFit2) fits a set of 2D points (fed from skeletonized needle images and thresholded surface features) to an ellipse (see Figure 5.11) with major (A) and minor (B) axis, centre point (H,K), and rotation angle (α) using a non-linear least-squares approach to enclose as many points on/within the ellipse as possible. The non-linear approach is based on the Gauss-Newton method, which breaks down the problem by solving a sequence of linear least square problems [13]. A good ellipse fit is one whose curvature intersects all external boundaries of the feature being tracked.

Figure 5.10: Occlusion Invariant Needle Segmentation and Ellipse Estimation for non-obstructed (top) and obstructed (bottom) cases.
For surface feature fitting, elliptical solutions were relatively consistent, as feature edges were well-defined and smooth and there was no background noise. For needle fitting however, solutions were relatively inconsistent, as lack of pixel information lead to a multitude of elliptical solutions [13]. Many of these solutions underestimated needle curvature (Figure 5.12), and on rare occasions, yielded off-centred ellipses that did not intersect needle curvature (from the thinned needle image).

As a means of rejecting these irregularities, a RANSAC algorithm was implemented to check for valid ellipses. For each pixel in the skeletonized image, the distance to the two focus points of the fitted ellipse
are calculated. If the total distance is less than or equal to $2^*$ the major axis ($a$), this point is inside the ellipse [37]. If and only if $95\%$ of pixel points fit within the fitted ellipse (based on the above criteria), the fit is deemed acceptable. For each capture sequence, RANSAC processes skeletonized pixel samples, applies ellipse constraining criteria and retains valid ellipse fits.

Unfortunately, due to noise and lack of pixel information (i.e. a discontinuous semi-circular arc of a circle), a high percentage of false (underestimated, off-centered) ellipses were present in many capture sequences (and were actively rejected by RANSAC). As a result, a large sample size (75 images) was required to obtain at least 5 or more good RANSAC elliptical fits. Of these successful fits, all were averaged and the resulting parameter values ($A_{av}, B_{av}, H_{av}, K_{av}, \alpha_{av}$) were sent for further processing.

5.4.2 Results

Robust Surface Feature Ellipse Fitting

Figure 5.13 shows the robustness of surface feature ellipse fitting to a variety of lighting conditions. Lighting was varied by adjusting the illumination power level on the Xenon lamp (onboard the endoscope) from -4 (low) to +4 (high). Neutral lighting corresponds to 0.

Robust Needle Ellipse Fitting

Figure 5.14 shows average ellipses obtained from OpenCV’s FitEllipse2 function for low and high lighting conditions, with the 5% RANSAC tolerance criteria imposed.

Together, these results look promising for robust, light invariant surface feature and needle shape estimation (ellipse fitting) for automated suturing.
Figure 5.13: Surface Feature Ellipse Fitting For Lighting-Invariant Tracking. Surface feature segmentation and ellipse fitting is robust to neutral (left), high (middle), and low (right) level lighting conditions.
Figure 5.14: Lighting-Invariant RANSAC Constrained Needle Ellipse Fitting. Needle ellipse fitting and shape estimation is robust to low (top) and high (middle) lighting conditions.
5.5 Circular Pose Estimation

5.5.1 Background

Challenges With Stereo Vision For Robotic Applications In Laparoscopic Surgery

In the field of computer vision, stereo image triangulation (perceiving depth by triangulating images from two cameras) is the defacto method for obtaining 3D information about an object [22]. However, computational cost, large memory allocation size, and inflexibility make it nonideal for real-time performance applications [22]. While there have been strides to address these computational disadvantages [31], vision-guidance in a laparoscopic surgery context presents an intrinsically greater problem.

The primary source of image-acquisition in laparoscopic surgery is a continuous live video feed from one endoscopic camera system, placed through a trocar inside the patient abdomen. To mimic a stereo-vision setup, one would have to introduce a secondary endoscope through another trocar, which would unnecessarily increase patient trauma and delay recovery.

Single-port stereo endoscopes are commercially available and could potentially circumvent adding an additional trocar. Intuitive Surgical and VisionSense both have commercial single chip stereo endoscopes designed for 3D visualization within the human body. While these tools can provide depth information for tissues, organs and blood vessels to a surgeon during laparoscopic surgery (without introducing an additional trocar), they are not suitable for robotic applications - the disparities between the camera lenses are too small (in the order of millimetres), and as a result, depth resolution is low (\(\leq 7\)mm) [22].

Due to these laparoscopic camera constrains and computational challenges in real-time stereo-depth map reconstruction, monocular pose estimation techniques may be a more viable alternative.

Single Image Perspective Projections Using Simple Shape Primitives

Monocular pose estimation for object tracking is a challenging vision problem - there is an inherent ambiguity in solutions for position and orientation estimation for an object in 3D space. To address this, many researches have exploited the unique projective properties of rectangles, parallel lines, orthogonal point, and circles [35] to aid in robust tracking applications (formulation is presented in Appendices C.1 and C.2).

In our project, we exploit known geometric properties of circles (and their elliptical projections in a single camera image) to uniquely estimate 3D circle pose for desired objects of interest.
5.5.2 Methodology

Figure 5.15 visualizes an adaption of de Ipina et al.'s TRIP tag monocular circle pose estimation algorithm for surface features and a semi-circular needle of known size [20].

1. Obtaining 2D Ellipse Parameters

1) An approximated ellipse from Open CV’s EllipseFit2 module (for individual surface features) or 2) an average of RANSAC-constrained ellipses (from background-subtracted, skeletonized needle segmented images) for needle shape estimation are first calculated.

This average/estimated ellipse has 6 parameters: $A,B,H,K,\alpha$ (Figure 5.11). $A$ and $B$ are half length axes of the ellipse, and $H$ and $K$ are centrepoint coordinates of the ellipse in the zoomed image (the image is zoomed to allow for better surface feature/needle segmentation).

Normalized (unzoomed) centrepoint coordinates $H^*$ and $K^*$, $A$, $B$ and $\alpha$ define $C$, the ellipse used for circle pose estimation.

2. Normalize Ellipse

To map the ellipse from a 2D image plane to a 3D camera coordinate system, $C$ is multiplied by the intrinsics matrix $K$ (which uses parameters such as focal length, effective pixel size and pixel skew to describe 2D points on a 3D plane) to obtain the normalized ellipse, $C_n$.

The camera intrinsics matrix $K$ is obtained from OpenCV’s calibration routine (discussed in the Calibration and Registration section below).

3. Map Ellipse To 3D Cone

The goal is to infer the position and orientation of a circle, $c$, by rotating the ellipse, $C_n$, so that the intersection of a 3D cone with the ellipse plane becomes a circle. This circular plane only describes true orientation (and not position) of the circle in 3D space. The rotation matrix, $R_c$ describes ellipse to circle orientation mapping by relying on the diagonalized eigenvalues of $C_n$.

4. Calculate 3D Circle Orientation

A normal vector, $\vec{n} = R_c \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ describes the orientation of the circular plane in 3D space. For needle tracking, this vector will help the tool approach a suture needle in tissue with the correct pose. For surface feature tracking, the tool will align the needle orthogonal to the normal vector $\vec{n}$ for minimal tissue tearing and optimal needle insertion.
5. Calculate 3D Circle Position

Geometric relations between the ellipse and circle plane are exploited to infer 3D position of the circle centre. The first geometric quantity, \( d \), is the distance between the camera and the target circle plane. The second quantity, \( \delta \), is the ratio of the physical radius of the circle, \( \rho \), to the radius of the the rotated circle (calculated in section 3), \( r_o \), such that \( \delta = \frac{\rho}{r_o} \).

Then, the position of the circle centre \( T = R_c \cdot [\delta \ 0 \ d]^T \)

Together, \( T \) and \( \vec{n} \) describe the pose of the circle in 3D space.

![Figure 5.15: Workflow Diagram For Ellipse-To-Circle Backprojection Algorithm](image)

**Pose Estimation Parameter Selection For Needle and Surface Features**

As illustrated in the formulation above, given a camera intrinsics matrix \( K \), an ellipse \( C \) and known circle radius \( \rho \), one can calculate the position and orientation of a circle in 3D space. For surface features, \( \rho = 3mm \) and for the needle \( \rho = 9mm \).

5.6 Calibration and Registration

5.6.1 Methodology

Due to curvature in the endoscope lens, images were distorted and affected accurate 3D ellipse-to-circle back projection data for robotically guided needle placement and insertion. Open CV provides a built-in camera calibration routine which corrects for lens distortion using a pinhole camera model, estimating camera intrinsic parameters. Unfortunately for image-guided applications, where sub-millimetre accuracy is required, this model is insufficient.
To correct for these deficiencies, a circular grid pattern was used (Figure 5.16) for ellipse acquisition, circle estimation, and ground-truth correction with robotic assistance.

A 3D volume of planar data sets (the circular grid oriented in random poses) were captured and processed with the ellipse-to-circle back projection algorithm. For each data set, corresponding ground-truth 3D points were recorded by using the robot tool-tip to touch corner points (at centre of each circle) on the circular grid grid. The base of the needle tool-tip (which was the most pointed surface on the tool) was used to touch these centre points.

These corner points were used to calculate a rigid transformation matrix mapping a flat circular grid matrix to one oriented in a specific pose in 3D space (Figure 5.6.2, 1st Panel). This matrix was used to infer the 3D position of all intermediate points in the circular grid.

The two 3D volume sets were then registered and fitted using a 3rd order polynomial (in terms of x,y,z). This model corrected for minor camera distortions, as well as provided registration from the camera space to the robot space.

5.6.2 Results

Figure 5.6.2 illustrates distortion correction for two different circular grid poses (blue corresponds to undistorted points, red corresponds to ground-truth corrected points). Circular grid points (Figure 5.6.2, 1st Panel) are transformed in 3D space based on grid pose (inferred from ground-truth corner point data). This is then used to correct for distorted vision data (Figure 5.6.2, 2nd, 3rd and 4th Panel), and serves as fitting data for the 3rd order polynomial.

Figure 5.18 illustrates rationale for selecting a 3rd order fit. 4 planar data points (measured at a ground truth height of 114mm, with the surgical tool) were used to test the accuracy of 1st to 4th order polynomial fits. The vision system then captured these points, and distorted camera coordinate data was passed through each fit and the registered/transformed coordinates were recorded. Error in X, Y and Z was calculated for each planar data point, and the magnitude of the RMS error was subsequently calculated. Figure 5.18 shows that the 3rd order fit has the lowest RMS error, and therefore corrects for distortion and other non-linear effects in the system most effectively.

For any surface or needle based elliptical features appearing in the camera image, as long as they were present within the calibrated 3D volume, corresponding robot-equivalent coordinates were calculated with the 3rd order model.

The positioning accuracy of circular features calibrated with the 3rd order model were validated by having the needle grasper tool tip touch the centre-points of (large) 20mm black circles (with white dots...
representing the centrepoints) lying on a flat plane within the calibrated volume. The distance from the tool-tip to the centre-point was measured, and averaged over 100 runs. The average positioning error was ±1.5mm. The base of the needle tool-tip (which was the most pointed surface on the tool) was used to measure positioning error.

Inconsistencies in the refined calibration procedure (touching the tool tip at the centre of corner grid circles) and minor variability in elliptical size feature estimation from projected circular features in a camera image may explain the presence of non-zero positioning error.
Figure 5.17: Distortion Correction For Two Circular Grid Poses (units in mm). Blue: Distorted Vision Data. Red: Corrected Ground-Truth Calibrated Vision Data. Note: Data is normalized (unitless) for visualization purposes so distorted and undistorted points can be superimposed on each other to demonstrate effective of non-linear distortion, and corrective effects.
5.7 Depth and Positioning Accuracy

de Ipná et al.’s circular pose estimation algorithm is employed in this project to calculate the position $(p_x, p_y, p_z)$ and orientation $(n_x, n_y, n_z)$ of a circular object in Cartesian space. Due to lens distortions effects, these 6 parameters often inaccurately represent the true pose of circular objects, and a 3rd order polynomial fit (obtained from calibration and registration with the robotic system) corrects for this, accurately mapping undistorted 3D camera coordinates to corrected 3D robot coordinates.

In order to characterize the coupled and decoupled error effects of the polynomial model and pose estimation system, the smallest change in depth and angle were quantified over a range of circles in various positions and orientations. Synthetic image data was used to explicitly measure depth/angle changes in the pose estimation system, while real-world images were used to measure depth/angle changes in the combined pose estimation/3rd order model system.
Depth Resolution

Figure 5.19 describes the circular pattern grid setup used to measure the smallest change in depth synthetically and in the real-world. Circles were 5mm in size, spaced 20mm apart.

![4-Point Circular Grid To Quantify Depth Resolution](image)

**Figure 5.19: 4-Point Circular Grid To Quantify Depth Resolution.**

*Synthetic Images*

In the synthetic case, it was assumed that the grid was placed perfectly perpendicular to the camera, so the image projection corresponded to 4 (true) circles. In this scenario, changes in depth were linearly proportional to circle radius. In our application, 5mm circles corresponded to a 10 cm distance from the camera origin. These circles were varied in size to correspond to ± 1mm depth changes, simulating placing the grid closer/farther to the camera (Figure 5.20).

![Synthetic Circular Grid Varied In Magnification To Reflect Placing the Grid Closer/Farther From The Camera](image)

**Figure 5.20: Synthetic Circular Grid Varied In Magnification To Reflect Placing the Grid Closer/Farther From The Camera**

Figure 5.21 illustrates near-perfect accuracy for measuring depth changes at 1mm intervals, obtaining positioning errors of < 0.1mm.
Figure 5.21: Depth Resolution Results - Synthetic Images. For 1mm sampling intervals, < 0.1mm positioning errors were observed.

Real-World Images

For the real-world case, the circular grid was placed at 10.5cm. This was measured by placing the grid on a flat aluminum block, and using the robot tool-tip to touch the centre of each circle (10.5cm corresponds to the height from the circle centre to the base of the robot origin world frame coordinates). The height of all 4 circles were then averaged, and used as a ground-truth for comparison, against the coupled pose estimation/3rd order model system.

For each height, ellipses are fit to a binary image of the circles in the grid (using a sub-masking technique, similar to surface feature ellipse fitting), and run through the calibrated pose estimation system. The 4 heights are averaged and compared to the ground truth height. Heights were varied at ±5mm increments, covering a depth range of 90-120mm. Figure 5.22 illustrates good change in depth measurement capabilities (<1.5mm positioning error at 5mm depth intervals) for the calibrated (coupled) pose estimation system. While this is high (40 %), it is not unreasonable, considering < 1.5mm posi-
tioning accuracy for the 3rd order model, and < 0.1mm for only the pose estimation system. Obtaining samples at <5mm increments was difficult, as thin metal plates (5mm in height) were used to change height of the circular grid in 3D space. Any inaccuracies we see in Figure 5.22 are due to calibration inaccuracies (discussed earlier), resulting in the presence of minor non-linearities in the calibrated pose estimation system. The two outliers in Figure 5.22 may have occurred due to accidental motion of the circular grid during one or more runs.

Figure 5.22: Depth Resolution Results - Real World Images. For 5mm sampling intervals, < 1.5 mm positioning errors were observed.

**Angle Resolution**

Figure 5.23 (for synthetic images) and Figure 5.25 (for real-world images) describe setups used to measure the smallest detectable change in angle for decoupled (pose estimation system) and coupled (pose estimation and 3rd order model) systems respectively.

*Synthetic Images*
In the synthetic case, a circle at 0° is rotated about the major (horizontal image) axis, changing at 1° increments till ±45° (Figure 5.23). These are generated by rotating a perfect 200x200 pixel circle about the major axis $a$. For each degree change, the minor axis $b$ is reduced by $\sin(\theta) \times b$, therefore simulating projection of a rotated circle (about the major axis) on the camera plane. The initial (0°) tilt indicates a perfect circle directly perpendicular to the camera.

Figure 5.24 illustrates (near-perfect) 1 to 1 correspondence between expected and measured angles from the pose estimation system for synthetic image data (<0.1°). Any discrepancies may be due to minor variations in (RANSAC constrained) ellipse fitting.

**Real-World Images**

In the real-world case, the needle driver holds the centre of a semi-circular coloured needle (Figure 5.25). This needle is spray painted with green engine enamel paint (which sticks well to smooth metallic
Figure 5.24: Angle Resolution Results - Synthetic Images. A near-perfect 1 to 1 correspondence between expected and measured angles can be observed (error < 0.1°).

surfaces). It is positioned at approximately 0°, and rotated to ± 45° at 5° intervals. Figure 5.26 shows relatively inconsistent change in angle (with some significant outliers) over this range. Note: Ellipse fitting deteriorated for angles greater than ||45°|| (due to a lack of ellipse pixel power from needle rotation), and so values greater than these were not used. Note: the 3\textsuperscript{rd} order model could not be used with the pose estimation system for calibrated real-world angle measurement as it was only designed to correct for distortions in position, not orientation.

While there is some noise in the segmented ellipse image and minor variations in ellipse estimation (for the same angle), these are minimized (and potentially removed) due to skeletonization/thinning and RANSAC ellipse constraining. The non-linearities in lens distortion, while partially corrected for by OpenCv’s camera calibration routine, are not completely removed in the image, introducing mathematical inaccuracies in pose estimation. Pixel segmentation and ellipse estimation are skewed/distorted, and RANSAC constraining cannot discern ellipses in distorted images from ellipses in non-distorted images (so “valid” ellipses are still permitted). As a result, we see large, inconsistent (non-linear) angle mea-
surement inaccuracies in Figure 5.26, allowing us to conclude (with some certainty) that the **we can only discern angle differences with an accuracy of ±15°**.

In theory, if we could remove all non-linearities from the image, we could expect sub-degree, sub-mm orientation and position capabilities, as demonstrated by synthetic image pose estimation data.

Figure 5.25: Real World Image Setup: Coloured Needle Actuated By Robotic Tool, To Quantify Angle Resolution.
Chapter 5. Vision Subsystem

5.8 Touch Point Calibration

The circular grid jig used for camera calibration/robot space transformation can also be used to reinitialize the system, in an event that the robot or camera system moves.

The vision subsystem captures the 4 corner circles of the jig in 3D camera space, and the trajectory generation process converts this to robot-equivalent coordinates (using the $3^{rd}$ order polynomial fit). The user then manually guides the tool-tip (with the robot in LIMP mode) to the 4 corners of the jig, and calculates ground-truth robot world coordinates. A rigid transformation matrix maps trajectory-based robot-equivalent coordinates to these ground-truth coordinates [24], which is then used for accurate needle positioning and insertion during an automated suturing.
Chapter 6

Trajectory Generation Subsystem ||
Curved Needle Path Planning

Figure 6.1 provides a block diagram for the trajectory generation subsystem. For a given entry/exit suture point pair, an Optimal Centre Point (OPC) is calculated, from which insertion, pick-up and realignment trajectories are generated. Insertion and re-alignment motions depend on surface pose geometry, while pick-up motion depends on needle pose.

6.1 Planning A Minimal Deformation Path For Smooth Needle Insertion

To ensure robotically-guided automated suturing is robust to a variety of tissue types, an adaptable needle insertion method needs to be implemented, using vision to track and dynamically minimize deformation. Nagotte et al. [25] suggests that in thin tissues, needle reorientation is possible, and tool jaw/needle contact is rigid and secure. However in the case of thick tissues, these two conditions may be violated and result in tissue tearing. To circumvent these issues, they suggest the ideal motion of the needle be along its circular arc.

It is assumed that for any given suturing sequence, the needle diameter, D, is greater than the distance between any given entry/exit point, i.e. \( ||I - O|| \leq D_n \). In this scenario, rotation about the needle centre is sufficient to pass the needle through the tissue, provided this centre point is positioned to ensure the needle tip passes through the entry point \( I \) and comes out through the exit point \( O \).
Figure 6.1: A Block Diagram Of The Trajectory Generation Subsystem. An entry/exit suture point pair is used for optimal centre point (OPC) calculation, subsequently generating joint-level needle insertion, pick-up and re-alignment trajectories for robot control.
6.2 Determining An Optimal Centre Point (OPC) of Rotation

For any given $I$ and $O$ point pair, an optimal centre point $C$ is selected such that a needle of radius $R$ intersects $I$ at an approach angle $\theta_i$ and $O$ at an exit angle $\theta_o$ (assuming no deformation). The following equations define relational constraints between $I$, $O$, and $C$:

\[ R_i = \sqrt{(Ix - Cx)^2 + (Iy - Cy)^2 + (Iz - Cz)^2} \] (6.1)

\[ R_o = \sqrt{(Ox - Cx)^2 + (Oy - Cy)^2 + (Oz - Cz)^2} \] (6.2)

Where $R_i$ is the distance between the needle centre and point $I$, $R_o$ is the distance between the needle centre and point $O$, and $R$ is the needle radius. Equating $R_i = R_o = R$ and solving for 6.1 and 6.2 results in an infinite set of solutions for an optimal centre point $C$, a subset of which are shown in Figure 6.3 below. The centre point solution which is vertically above $I$ and $O$, and yields a pitch angle closest to the horizontal plane (i.e. Pitch = 0) without the robot hitting a singularity is selected as the optimal $C$ (i.e. $((I - C) \times (O - C)) \cdot [1 \ 0 \ 0] = 0$). In Figure 6.3, this corresponds to the uppermost point on the semi-circle of multiple centre-point solutions.

In the case that $I$ and $O$ deform as the needle is being inserted, the vision subsystem will continuously scan entry/exit surface points, and a new optimal centre will be recalculated. This will iteratively modify the needle insertion trajectory, changing $C$ to minimize displacement error between the needle and point $O$. This should ensure the needle tip passes through the desired $O$ (within error) accurately.

This method can also be adapted for extremely thick tissue scenarios where $||I - O|| > D_n$. Given an $I$ and $O$ point pair, the optimized centre point algorithm attempts to find an $O^*$ closest to $O$ such that $||O - O^*|| \rightarrow 0$ while ensuring that the selected $O^*$ keeps the tool tip orientation as close to the horizontal plane as possible (Pitch = 0).

Approach and exit angles, given an optimal centre point $C$, are determined by solving the 3D Circle Equation for the angle parameter $\theta$:

\[ P = R\cos(\theta)u + R\sin(\theta)(n \times u) + C \] (6.3)

Where $P$ is a 3D-point on a circle (the semicircular needle) with centrepoint $C$ and radius $R$. $n$ is normal to the $C-I-O$ plane, and $u$ lies on this plane (pointing from $C$ to $P$):

\[ n = \frac{(I - C) \times (O - C)}{||n||} \] (6.4)
Figure 6.2: Infinite Solutions For An Optimal Centre Point Calculation. Note: Only a subset of these infinite solutions are visualized here.

\[ u = \begin{bmatrix} 0 & n_z & -n_y \end{bmatrix} \]  \hspace{1cm} (6.5)

\( u \) was arbitrarily selected to ensure the dot product \( n \cdot u = 0 \).

Equation 6.3 is simply a transcendental equation \([9]\) of the form:

\[ c = a \cos(\theta) + b \sin(\theta) \]  \hspace{1cm} (6.6)

with \( c = P \cdot C \), \( a = R \cdot u \), and \( b = R \cdot (n \times u) \). Making a substitution \( v = \tan \frac{\theta}{2} \), one can solve for the
quadratic in $v$:

\[(a + c)v^2 - 2bv + (c - a) = 0 \tag{6.7}\]

and therefore,

\[\theta = 2\tan^{-1}\left(\frac{b \pm \sqrt{b^2 + a^2 - c^2}}{a + c}\right) \tag{6.8}\]

A unique solution for $\theta$ is obtained by ensuring $P = I$ (for $\theta_i$) or $P = O$ (for $\theta_o$) in Equation 6.3. Note: the value of $\theta$ is with respect to $u$, NOT with respect to the surface normal.

To calculate the true “insertion” angle, let us denote the vector between $I$ and $C$ as $a$. Then, we can take the cross product of $a$ and $n$ ($a \times n$), and compare it to the the true normal of the surface point (obtained from surface feature pose, during camera image acquisition). The difference in angle between these two vectors (the scalar product) corresponds to the deviation from the ideal “90°” needed for optimal needle insertion.

Unfortunately, due to the bad angle resolution in our system, we cannot depend on surface normal acquisition, as a reliable means of computing deviation from the ideal insertion angle. To compensate for this, we can use needle pose acquisition and centre point extraction after needle insertion is complete. The assumption is that there will be some longitudinal deformation during needle insertion (assuming the insertion angle is at 90° relative to the surface), but pose acquisition can correct for this by tracking the translated needle centre point (which has deviated from the ideal insertion trajectory). As a result, subsequent pick-up and realignment phases are accurate.
Figure 6.3: Calculating the true “insertion” angle $\alpha$ as a deviation from the tissue surface normal (corresponding to the $z$ axis of $F_{tis}$), relative to $F_{tip}$ (the frame tangential to the needle tip). Note: $\theta$ corresponds to the angle between $u$ (which lies on the needle plane) and $F_{tip}$.

### 6.3 Generating Circular Needle Trajectories

1. **Insertion at $I_i$**

Equations 6.3 - 6.5 describe the pose of a semi-circular suture needle in 3D space and helps determine $\theta_i$ and $\theta_o$, approach/exit angles for a needle insertion procedure. The end-frame of the proposed inverse kinematics model (Section 3.4.1) is structured around the needle’s centrepoint of rotation. To accommodate circular rotation about the needle centre, all that is required are a set of angles $(\theta_i...\theta_n...\theta_{n+1}...\theta_o)$ describing needle passage through the tissue from $I_i$ to $O_i$. For each $\theta_n$, the inverse kinematics model offloads the rotational constraint angle to joint 7 and computes values for joints 1-6 by ensuring the CPR-frame is fixed (and only rotates) during needle insertion. $\delta\theta=\theta_{n+1}-\theta_n$ was chosen to be 0.1° to
ensure to ensure a smooth, circular arc trajectory through thick tissue.

Using this inverse kinematics model, the tool is aligned at the approach angle $\theta_i$. Figure 3.4 illustrates positioning of the needle centre $F_n$ at the optimal centre point $C_i$, at an entry angle along aligning the tip normal to the tissue and at the insertion point $I_i$ (located at $F_{tis}$ in Figure 3.4).

2. Piercing about $C_i$

The next set of needle-passing trajectories generated for $\theta_i\ldots\theta_n\ldots\theta_{n+1}\ldots\theta_o$ drive needle rotation about a CPR, whose axis is defined by the plane on which the needle lies during rotation.

3. Approaching $O_i$

The tool now approaches $O_i$ to facilitate pulling out the needle from the exit point in the tissue. Due to the thickness of the tool-tip and challenges with grabbing a thin-needle, an additional set of angles ($\theta_{o+1}\ldots\theta_{o+n}$) are required to ensure the needle tip is protruding well past the exit point ($\theta_{o+n} - \theta_o = 60^\circ$). After insertion is complete, the tool releases the needle (which is inside the tissue) and radially pulls back 30mm. The vision system then recaptures the needle pose to account for any translational/rotational changes during needle release, and computes a modified optimal-centre point. Then, $\theta_n\ldots\theta_{o+1}$ are re-computed, and the needle tip approaches $\theta_{o+n}$ with opens jaws, and partially closes (enough to prevent the needle from moving, but enough to allow the tool to slide along the needle). It then moves along the needle curvature (driven by the 7-DOF inverse kinematics model) until it reaches $O_i$ at $\theta_o$ and the jaws close. The needle is ready to be removed and realigned to $I_{i+1}$.

4. Pulling out from $O_i$

The needle continues to rotate about $C_i$ form $\theta_o$ to $\theta_r$, where $\theta_r - \theta_o = 180^\circ$. The needle is then pulled up vertically to exhaust the suture thread through the tissue.

5. Approaching $I_{i+1}$

The optimal CPR for $I_{i+1}$ and $O_{i+1}$ is calculated using Equations 6.1 to 6.8, and the approach angle and new CPR is fed to the inverse kinematics module. The needle tip is aligned and inserted up to $\theta_r = \theta_i + 60^\circ$. 

6. Realigning at $I_{i+1}$

At this point, the tool is partially open, and slides back to the suture thread/needle boundary (as it was initially configured at the start of the procedure), $120^\circ$ along the needle curvature, and closes. The needle can then continue to be inserted until the tip reaches $O_{i+1}$.

Steps 1-6 can then be repeated for $n$ surgeon defined entry-exit suture pairs.
Chapter 7

System Performance

The following video (http://tinyurl.com/c3ub9kk) demonstrates a 2 minute automated vision guided suturing sequence for insertion through one set of surgeon defined entry/exit suture points and repositioning at the next set. Mechanical, vision and trajectory generation subsystems effectively communicate with each other to accurately insert and pull out a suturing needle through the first set of surgeon defined entry/exit suturing points, and re-position (and partially insert) it at the second insertion point.

Accurate surface feature detection (and interpolation, based on surgeon-defined entry/exit points) allows the needle to be positioned at the first desired insertion point $I^{1*}$. 7-DOF redundant manipulator inverse kinematics allows for rotation about the virtual needle centre, facilitating smooth needle insertion through the tissue. Accurate needle pose estimation (and centre point extraction) allows the tool to grab the needle at the surgeon defined exit point, $O^{1*}$, regardless of tissue-induced needle translation. Finally, the needle is pulled out (to simulate suture exhaustion through the tissue) and repositioned at the next surgeon defined entry point, $I^{2*}$, to show autonomous continuity of the procedure.

Figures 7.1 and Figures 7.2 visually highlight accuracy of the insertion, needle pose estimation, and pickup tasks.
Chapter 7. System Performance

(a) Accurate Insertion at $I_1$ and Exiting Through $O_1$ (Minimizing Tissue Deformation Through OPC Calculation). In this image, $I_1$ corresponds to a surgeon defined entry point lying between two adjacent surface markers, and similarly, $O_1$ corresponds to a surgeon defined exit point lying between two adjacent surface markers.

(b) Accurate Needle Centrepoint Extraction (And Tool tip Position Verification), From The Calibrated Pose Estimation System

Figure 7.1: Accurate Insertion and Pose Estimation Of A Curved Needle
(a) Moving About The Needle Curvature (With Partially Open Needle Graspers) to Approach $O_1$

(b) Accurately Picking Up The Needle At $O_1$

Figure 7.2: Accurate Pick Up Of a Curved Needle
7.1 Lack of Suture Thread

The curved needle used in the video sequence above had its suture thread removed, to ensure accurate needle segmentation and pose estimation. Background subtraction innately removes the ability to distinguish between the suture and suture needle (both are segmented and skeletonized), which skews ellipse fitting and pose estimation.

Colour thresholding can be used to segment the needle from the suture thread, if a coloured needle (with a different colour suture thread) is employed. During initial experimentation, a green coloured needle (similar to Figure 5.25) was attached to a suture thread to investigate these discriminating capabilities. Unfortunately, while colour thresholding provided superior segmentation of the needle from the suture thread, insertion through the tissue was difficult, as there was friction between the paint layer and the silicone in the tissue pad (causing tearing in some instances, and wearing away of the paint).

An alternate, commercially available solution to circumvent insertion issues while providing suture thread/needle discrimination is Ethicon’s VISI Black Needle [12], shown in Figure 7.3. This needle provides a slick surface for smooth needle insertion, and has a black, matte (non-reflective) colour for improved visibility and maneuverability in the surgical site of operation. Assuming the thread is non-black in colour (the needle comes attached with purple suture threads), one can use HSV colour thresholding in place of background subtraction (and skeletonization) to arrive at a well-segmented (lighting and shift invariant) needle, similar to those in Figure 5.14. Computationally, this is less expensive and avoids having to depend on a static background (which is unrealistic, in a surgical scenario) for needle segmentation.

![Figure 7.3: A Tapered VISI Black Needle](image)

7.1.1 Orientation Change In The Needle When Releasing (After Insertion)

It is assumed that the needle is being inserted into relatively thick tissue (which is a reasonable assumption for the case of the suture pad), and so it will remain relatively secure when the tool releases the
Pose estimation of the needle only captures the needle centre, not its orientation change, as system angle characterization (described earlier, in the “Smallest Detectable Angle” section) yields large, inaccurate results ($\pm 15^\circ$ error).

In an invivo scenario where tissue thickness can dramatically vary (and the needle will not necessarily remain rigidly secure when released), orientation estimation has to be refined (perhaps by identifying and applying a comprehensive distortion correction model on a captured, segmented image of the needle) before the same procedure can be repeated.

7.2 Challenges With Re-Aligning At The Second Suture Insertion Point

The video sequence above successfully demonstrated accurate placement of the grabbed suture needle through the first entry/exit point pair ($I_1/O_1$) and realignment to the second insertion point, $I_2$. To facilitate complete insertion through $I_2$, the needle needs to be partially released (by partially opening the needle graspers), and the tool needs to move along the curvature of the needle upwards (compared to downwards, as shown in Figure 7.2a). Unlike the approach $O_1$ scenario, the needle is not within the tissue and will uncontrollably fall out of place (drastically change in both position, and orientation).

As discussed in the previous section, any changes in orientation cannot reliably be captured, so an alternative, more controlled solution would need to be pursued.

The idea is to partially open the needle graspers, move by $1^\circ$ upwards, scan for pose changes, and update the needle centrepoint to the inverse kinematics algorithm. The assumption here is the needle orientation will remain relatively the same (or change minimally).

Of course, tracking with the needle grabbed by a tool is practically infeasible with the current iteration of this project as there are inherent segmentation problems with background subtraction (as discussed in the “Lack of Suture Thread Section”) when an occluded object is introduced. As a result of this limitation, the needle and tool will be fed to needle pose estimation (which will yield inaccurate ellipses). Either Hough lines can be used to mask the tool, or colour thresholding can be used to track the needle (and ignore the tool altogether) in each $1^\circ$ movement iteration.

If needle orientation dramatically changes when moving by $1^\circ$ upwards, a distortion correction strategy (as proposed by the previous section) would need to first be implemented (and refine orientation estimation) before Re-Alignment at the second suture point is feasible. Once this is accomplished, however, insertion can proceed like the first suture insertion step (with entry $I_1$ and exit $O_1$ points).
7.3 Quantifying Needle Positioning Accuracy

Figure 7.1a shows accurate needle insertion at a surgeon selected insertion point (an interpolated 3D weighted average of adjacent, surface features). However, it is hard to say to what accuracy the needle tip actually touched this point (and the 2D image may not be a completely reliable means of quantifying this).

100 experimental positioning tests were performed by having the needle touching all 3 surface insertion points (33 runs / surface feature), with the help of the surface feature pose estimation system, and entry angle calculation from the trajectory generation subsystem.

As long as the needle tip was within the perimeter of the circular surface marker, positioning was deemed appropriate. All tests satisfied this criteria. Therefore we conclude that the needle tip can be positioned with with at most 3mm accuracy. Based on the 1.5mm positioning accuracy established with the 3rd order model, we expect less than 3mm positioning accuracy, but have no accurate way of measuring/quantifying it.

7.4 Procedure Repeatability

33 autonomous suturing sequence runs were performed, similar in nature to the video presented earlier. 11 of these consisted of suturing through the same set of entry/exit points with neutral lighting, the next 11 corresponded to variations in lighting (4 low, 3 neutral, 4 high), and the last 11 corresponded to variations in suture point locations (by translating the tissue pad within the camera image). A successful run is one where the robot inserts the needle close to the first surgeon defined insertion point $I_1$ (within 3mm), picks up the needle at $O_1$ and repositions and partially inserts it at the second surgeon defined insertion point $I_2$ (within 3mm).

Figure 7.4 tabulates successful and unsuccessful runs for consistency, lighting and positioning variation scenarios.

7.4.1 Consistency For Controlled Lighting/Suture Point Locations

In the first 11 runs, the robot sutured through the same set of suture points (i.e. positioned as shown in the video), in neutral lighting conditions (i.e. the Endoscope Xenon Lamp illumination level was set to 0). Of these 11, 10 (90\%) successfully completed accurately.
Table 7.4: Procedure Repeatability - Consistency, Sensitivity To Lighting, and Sensitivity To Suture Point Locations. “Lighting” Corresponds to Variations In Lighting, and “Translation” Corresponds to Translation of The Suture Pad In The Image. Note, red numbers indicate low-percentage success runs.

<table>
<thead>
<tr>
<th>Type</th>
<th>Sub-type</th>
<th>Runs</th>
<th>Successful</th>
<th>Unsuccessful</th>
<th>% Successful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consistency</td>
<td></td>
<td>11</td>
<td>10</td>
<td>1</td>
<td>90.91</td>
</tr>
<tr>
<td>Lighting</td>
<td>Low</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td><strong>Average Success:</strong></td>
<td>83.333333333</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Translation</td>
<td>Up</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>Down</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>33.33</td>
</tr>
<tr>
<td></td>
<td>Left</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>Right</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td><strong>Average Success:</strong></td>
<td>83.333333333</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td>33</td>
<td>28</td>
<td>5</td>
<td>84.85</td>
</tr>
</tbody>
</table>

One run (the 8th run) failed at the pick-up stage, missing the needle during approach, and subsequently (incorrectly) grabbing nothing. This was due to an underestimated ellipse fit from segmented needle pixels, that were low in pixel power because the needle was inserted slightly too deep (this may have been because the suture points may have been deformed from a previous run, and were still recovering to their original undeformed state). RANSAC constraining did not deem these ellipses invalid (as all satisfied the focus distance criteria). Lack of needle pixel information simply limits the (inherent) accuracy of least-squares ellipse fitting.

One potential solution to circumvent this issue is to track the needle (assuming a colour thresholding approach is used) with the tool during insertion, and use a Kalman filter [9] to estimate the position of the needle in the tissue, based on previous (high pixel power) observations of the segmented needle before being inserted into the tissue. Ellipses fit to these needle segmented images are assumed to be more precise due to a large plethora of pixel information (and therefore a more accurate least-squares fit and RANSAC constraining). In order to accurately utilize previous (accurate) observations of the needle when it is in the tissue, one needs to assume a constant velocity model for the needle (which is reasonable, as insertion is relatively slow) [9].
7.4.2 Sensitivity To Lighting

Autonomous suturing was performed in low (4 runs, illumination level -4), neutral (3 runs, illumination level 0), and high (4 runs, illumination level 4) lighting conditions. The tissue pad was maintained in the same (default) location as consistency tests. All neutral and high runs resulted in successful suturing sequences.

2 out of the 4 low levels illumination runs failed, for reasons similar to the previous section (extremely low pixel power needle segmented images, and underestimated ellipse fits). In this scenario, however, pixel power is low whether the needle is in or out of the tissue, so a Kalman filter has no “accurate” observation to estimate from. This is an inherent limitation of vision-based robotic guidance - high contrast needs to be provided to accurately discern various objects and perform a motion reliably and accurately.

Overall, autonomous suturing was successful 83% of the time for extreme variations in lighting conditions. If illumination is maintained at a neutral/high level, the robot will perform more reliably and accurately.

7.4.3 Sensitivity To Suture Point Locations

The default location for the suture pad corresponds to its placement in the consistency tests (and video sequence) described earlier. The suture pad was translated 6mm in all 4 directions (up, down, left and right in the camera image), relative to its default location (which was marked). Of the 11 runs, up, down and left were repeated 3 times (increasing in 2mm increments), and right was repeated 2 times (increasing in 3mm increments). Notice, a 6mm translation maximum was chosen to ensure all points were visible within the camera image, and fell within the calibrated 3rd order polynomial volume.

Of the 11 runs, up, left and right translations of suture pad points yielded successful runs. 2 out of the 3 down runs (at 4 and 6mm) however, failed, as the default location of the suture pad was placed extremely close to the bottom of image (to begin with), and 4-6mm translations from that point downward caused some 3D pose estimated feature points to fall on the boundary (and some even outside) the calibrated 3rd order polynomial volume. This is not surprising, as depth estimation is unreliable (non-linear, inaccurate) outside the calibrated volume, which subsequently skews insertion, pick-up and repositioning of the needle.

Overall, autonomous suturing was successful 83% of the time for variations in suture point translation (within the image). As long as suture points fell within the calibrated 3rd order volume, the procedure can reliably and accurately complete.
A general note with regard to procedure repeatability - over time, holes in the tissue pad form from repeated needle insertion, which could potentially skew results. For more reliable future experimentation, perhaps repeating the procedure at various (unsutured) areas on the tissue pad might remove these biasing effects.
Chapter 8

Conclusions And Future Work

Chapter 3 demonstrates sophisticated trajectory path planning, smart vision guidance algorithms, and a 7-DOF robotic system working seamlessly to realize the task of automated vision guided suturing on a silicone tissue pad. The following chapter discusses some of the challenges that still need to be addressed before robust automated suturing is feasible, and clinical integration of the system for potential animal testing and OR use.

8.1 Completing The Suture Sequence For N Entry/Exit Suture Points

As discussed in Section 3.7.3, real-time tool-occluded needle tracking needs to be implemented before re-alignment and complete reinsertion at $I_2$ is possible. Background subtraction inherently limits robust tracking due to the inability to discriminate between the tool and needle, ultimately skewing ellipse fitting and pose estimation. Alternatives were presented to address this issue (i.e. Colour Thresholding, with the VISI black needle), which may facilitate real-time tracking of the needle before, during and after insertion for any suture point pair.

8.1.1 Suture Management

Colour thresholding also filters out the suture thread (assuming it is of a different colour from the VISI black needle), which will physically enable the robot to suture an open tissue crevice shut. However, suture management is a challenging (and important) subtask in laparoscopic surgery that needs to be addressed to facilitate 1) proper suture motion through the tissue to close any open gaps and 2) avoiding entanglement due to complex tool motions.
Ensuring Proper Suture Motion Through Tissues

Theoretically, the suture should follow the motion of the needle, as it is rigidly secure to it. Therefore, if a cured needle is accurately inserted and picked up (from the other end), the suture should pass through entry point \( I^* \) and exit point \( O^* \). This holds true assuming the robot is manipulating the needle and controlling its motion through the tissue. If, however, due to potential errors in the vision system (from real-time needle tracking) the robot grabs the suture thread instead of the needle, the needle is now uncontrolled (dangling), which skews all positioning and orientation capabilities and ultimately prevents automated suturing from accurately completing.

If the suture thread is also tracked (via colour thresholding), then the boundary of the needle/suture thread can be used as a validation measure to reject incorrect grabs (i.e. if the tool has grabbed in the “thread” region, versus the “needle” region). If an invalid grab is detected, the tool will release, rescan the needle, and grab again (until it has made contact with the needle region).

Avoiding Suture Entanglement

Suture entanglement avoidance is addressed in the current iteration of automated robotic suturing through two motion tasks - 1) pulling radially back (10cm along the tool axis) when the needle is released after insertion and 2) Pulling back after the tool has re-grabbed the needle at exit point \( O^* \) (i.e. the “Suture Exhaustion Step”).

Radially Pulling Back After Needle Insertion

The idea behind pulling the tool 10cm away from the surgical scene is to avoid and remove any contact between the suture thread and tool body, and having thread fall onto the tissue pad. This allows the tool to approach the exit point \( O^* \) without having the suture thread follow the motion (if it is hanging on the tool body).

Suture Exhaustion After Needle Re-Grabbing At \( O^* \)

In order to ensure ample amounts of suture thread are available for \( N \) suture entry/exit points, the robot vertically pulls the needle up (5cm) and radially back (10cm), and then realigns to the next suture point. This ensures that the suture thread (rigidly attached to the needle) passes through the tissue and facilitates smooth (unconstrained) insertion at subsequent suture point pairs. Ideally, if the suture length \( L \) is known, the robot could pull out \( L-10\text{mm} \) (to ensure some thread remains at the first insertion point, \( I_1 \)).
For knot-tying applications, these values can be increased accordingly.

Suture Thread Tracking

The ideal solution for avoiding suture entanglement during any phase of the suturing procedure would be to track the suture thread (using colour segmentation techniques), and using that information to infer if it is interfering with the tool/needle manipulation during the stitching task.

8.2 Ex-Vivo Tissue Testing

Once the robot can autonomously suture through any $N$ entry/exit suture pair points on the tissue pad, the procedure can be adapted for thinner, more deformable vessels (such as ex-vivo pig lumen tissue samples).

There are two main challenges here, however in seamlessly performing automated suturing for deformable vessels - 1) obtaining accurate 3D tissue surface information, and 2) suturing with one tool through deformable tissue.

8.2.1 Obtaining Accurate 3D Surface Information

Obtaining 3D surface information for suture entry/exit points on ex-vivo pig lumen is a challenging task, as placing circular stickers (for pose estimation) on the surface is inviable and impractical for laparoscopic surgery (they wont stick, or durably remain in position, and cannot be placed on during a procedure).

Practically, some combination of 3D reconstructed preoperative/intra-operative MR data, biomechanical tissue model data, and registration with live endoscope images will need to be used to infer 3D position of the surface being operated on.

For proof of concept purposes, however, these steps are quite intensive and complex. A good intermediate step for translating the existing surface pose estimation setup for ex-vivo testing may be to use a laser-based heating system to “imprint” 0.25-0.5 mm circular tattoos on the lumen surface. The main assumption here, is that the size of these circles will remain relatively constant as the tissue deforms during automated suturing (as the TRIP pose estimation system assumes circles of constant radius).

At 0.25-0.5mm sized circular surface features, this assumption may be reasonable and hold for small deformations on the lumen surface. However, the camera has to be able to detect such circles and fit
ellipses to them accurately (currently it can detect 3mm circles accurately). This can be accomplished in two ways: 1) moving closer to the surgical scene and/or 2) digitally zooming the image. Both have intrinsic issues - moving closer to the scene might cause interference with the robotic tool during automated suturing, and digitally zooming the image means pixel information will be coarse (and inaccurate), potentially skewing pose estimation.

### 8.2.2 Suturing With One Tool Through Deformable Vessels

Experimentation was done with inserting a needle through a deformable silicone vessel, similar in fashion to suture point insertion on a tissue pad.

A centre point for needle rotation was arbitrarily chosen, through which the robot attempted to pierce the tissue surface. Initially, it was noticed that for points near the edges of the vessel, the needle slid on the surface, and failed to go through the tissue (as the tissue surfaced deflected inwards from the force/weight of the needle/tool tip). However, aiming to insert at a point well within the interior of the silicone vessel surface resulted in needle insertion through an edge point (although unreliably).

A secondary tool needs to be used to hold the vessel in place (and prevent it from collapsing on itself) during needle insertion. Only in this fashion can automated suturing for ex-vivo tissues occur reliably.

Furthermore, the tensile/shear forces exerted on the tissue also need to be measured, so deformation (and potential tearing) can be minimized/circumvented during real-time suture needle insertion. A 6-DOF ATI Mini-40 Force/Torque sensor was evaluated to measure tissue/needle forces, but was insensitive to any changes. Strain gauges at the tool tip need to be introduced to measure these small forces, and accurately guide robotic needle steering during automated suturing.

### 8.3 RCM Constraining

Section 3.4.3 discusses the challenges of imposing RCM constraints for the automated suturing task. Pose constraints from trocar placement may make needle insertion angles nonideal for the tissue surface being sutured on, introducing unnecessary deformation (and inaccurate exit of the needle through the wrong exit point, $O^1$).

While optimization techniques for appropriate trocar placement were discussed (to circumvent these challenges), they may not necessarily converge on a solution for a given set of suture points. A flexible
manipulator tip (8-DOF + system) may need to be developed (similar to DaVinci’s Endowrist instruments) to circumvent pose-constraint issues, and allow positioning of the needle (in any orientation) at a suture point. The kinematics/inverse kinematics of the system would have to be reconfigured for these added degrees of freedom, but the vision and trajectory generation subsystems can remain intact (working with Cartesian coordinates). A new inverse kinematics model would simply convert these coordinates and drive the robot in n-DOF joint space.

8.4 OR Compatibility

The automated robotic suturing system presented in this thesis can be readily incorporated into an OR-suite for ex-vivo (and potentially in-vivo) testing. Aseptic models of the 6-DOF Denso VP-6 series exist [28], which are identical in operation their non-aseptic analogs (which were used in this project) and can be readily draped for procedures. The EndoPath needle driver (attached to the custom motor mount) can be detached and autoclaved for sterilization and re-use in new automated suturing sequences (for various animal models). As long as the robot and motor mount are draped (similar in fashion to the DaVinci) and the tool is the only exposed part of the system making contact with the animal/patient, risk of infection is minimized.

Furthermore, a standard clinical grade Olympus endoscope was used for vision-guidance and feature recognition. This can also be sterilized using standard techniques and re-used for visualization during multiple automated suturing sequences.

8.5 Training Applications

The path planning methodology presented in this thesis can easily be adapted for a virtual simulator (for example, Mimic Solution’s DaVinci simulator), to aid in suture motion training for complex tissue geometries. The pose of the surface features and needle are provided (error-free) in real-time, which can be fed to the Trajectory Generation subsystem to calculate optimal centrepoints for needle insertion. These trajectories can be highlighted on the screen, and haptics can constrain the user to move about a fixed plane (to minimize shear deformation).

The idea is that a novice trainee practices with this suturing training system and is eventually able to tackle complex anastomosis scenarios laparoscopically (with or without robot guidance) in the future.
8.6 Final Thoughts

Given the challenges in realizing a robust, complete automated suturing system for in-vivo operation, this thesis presents an initial proof-of-concept prototype that manages to suture through 1 set of surgeon-defined suture points, and realign at the second set, on a silicone tissue pad.

The use of a curved suture needle, and robust vision techniques for robot guidance presents many suturing applications for this robotic system in variety of thick tissue scenarios. The eventual hope is that the challenges (described above) are addressed effectively, and this system helps surgeons suture complex lumen structures in a safe and efficient manner.
Appendices
Appendix A

DH Parameter Table

<table>
<thead>
<tr>
<th>Frame</th>
<th>a</th>
<th>α</th>
<th>d</th>
<th>θ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>-π/2</td>
<td>280</td>
<td>θ₁</td>
</tr>
<tr>
<td>2</td>
<td>210</td>
<td>0</td>
<td>0</td>
<td>θ₂</td>
</tr>
<tr>
<td>3</td>
<td>75</td>
<td>-π/2</td>
<td>0</td>
<td>θ₃</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>π/2</td>
<td>210</td>
<td>θ₄</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>-π/2</td>
<td>0</td>
<td>θ₅</td>
</tr>
<tr>
<td>6</td>
<td>-255</td>
<td>0</td>
<td>117.3</td>
<td>θ₆</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>π/2</td>
<td>0</td>
<td>π/2</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>π/2</td>
<td>0</td>
<td>θ₇</td>
</tr>
<tr>
<td>9</td>
<td>n_h</td>
<td>0</td>
<td>n_v</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-π/2</td>
</tr>
</tbody>
</table>

Figure A.1: DH Parameter Table For the 7-DOF Denso System. \( n_h \) and \( n_v \) correspond to the horizontal and vertical needle centrepoint offsets (z and y axes of \( F_n \), in Figure 3.3) relative to \( F_{nt} \). In our case, \( n_h = 0 \) and \( n_v = 9 \text{mm} \) (the needle radius)
Appendix B

Hough Transforms And
Least-Squares Fitting Techniques for
Shape Estimation

B.1 Line Estimation

Line detection is based on a simple point/line duality [3]. A line in an image is defined by the points whose coordinates satisfy the equation:

\[ y = kx + c \]  \hspace{1cm} (B.1)

Each point which satisfies equation B.1 can also be defined by a series of lines that pass through it, i.e. a point, \( P \), in image \((x,y)\) space defines a line \( p \) in parameter space, where as a point, \( l \), in parameter \((r,\theta)\) space defines a line, \( L \) [3] in image space.

As a practical example, suppose a line \( L \) in image \((x,y)\) space is traceable by white pixels in a a binary image, spread in small disconnected segments (and possibly interweaved with non-linear elements from other features). By the line-point duality, it must have a corresponding point \( l \) in parameter \((k,c)\) space [3].

Notice that the line-point duality fails when vertical lines are present (i.e. the slope \( k \) is undefined).
In order to circumvent this, Dude and Hart proposed an alternative polar form for the Cartesian line equation for Hough Transform processing[3]:

\[ \rho = x\cos(\theta) + y\sin(\theta) \]  \hspace{1cm} (B.2)

Equation B.2 represents a point in image space as a sine curve in parameter space. This means that the task of finding a suitable line \( L \) for noisy linear pixel data corresponds to finding the largest accumulator of sine curves intersecting at a common point \( l \) in parameter space.

### B.2 Circle Estimation

The method proposed above can be extended for any parametric curve (such as circles), with points in image space mapping onto multi-dimensional surfaces in parameter space [3]. The parametric equations for a circle in image space are:

\[ x = a + r\cos(\theta) \]  \hspace{1cm} (B.3)

\[ y = b + r\sin(\theta) \]  \hspace{1cm} (B.4)

\( a \) and \( b \) represent the centrepoint of the circle, while \( r \) represents the radius of the circle. This implies a 3 degree of freedom parameter space. This can be reduced by 1) knowing the gradient angle, \( \theta^* \), or 2) knowing the radius of the circle \textit{apriori}.

#### Pixel Points With Known Gradient Angles

For the case of known gradient angle, the equations above can eliminate \( r \) and reduce to:

\[ b = \tan(\theta^*) - x\tan(\theta) + y \]  \hspace{1cm} (B.5)

The gradient angle for a given pixel point describes the vector pointing from the pixel point \((x,y)\) to some centrepoint \((a^*,b^*)\). Gradient angles can be obtained using standard edge detection techniques, such as the Sobel edge operator [3]. Equation B.5 can be then be used for \textit{all} pixel points in image
space to detect a common circle centre (i.e. the intersection point at which the majority of edge normals intersect).

Circles Of Known Radius

In the case of known radius, \( r \), for each point in the image space, there is a centre \((a^*,b^*)\) and angles \( \theta_i \) (from 0-360°) which define a circle in parameter space. The intersection of all these parameter space circles is the true centrepoint of the circle.

Practical Considerations

In practice, inaccurate edge detection and inaccurate circle pixel segmentation (due to noise and discontinuous pixel regions) can dramatically introduce errors in the location of the circle’s centre [3]. Larger convolution masks can be used to improve the accuracy of edge detection, at the cost of increased computational load. However, Xavier et al. proposed an alternative solution which can circumvent edge detection altogether, exploiting secant lines between 3 points to find the circle centre [3] (Figure B.1).

For a given point triplet \( P_1, P_2, P_3 \), define two secant lines \( \overrightarrow{P_1P_2} \) and \( \overrightarrow{P_2P_3} \). One can then find the circle centre \( C \) by finding the intersection of lines perpendicular to, and on the midpoint of the secant lines [3] (Figure B.1).

![Figure B.1: Secant Line Intersections For Accurate Circle Centre Detection](image)

B.3 Fast Hough Transforms For Lines And Circles

Since the Hough transform works with accumulator arrays, it is slow, takes up large amounts of memory, and is computationally inefficient. A large effort goes into scanning the parameter space, in order to determine how many point transforms (curves/surfaces) intersect each other.
Hough transforms also work better for larger shapes than smaller ones, simply because there are less well defined accumulation points in parameter space (and therefore solutions are ambiguous).

Li and LeMaster proposed the Fast Hough Transform (FHT) as a means of reducing the search time in parameter space, by dividing into a series of "hypercubes" arranged in tree structure - in the 2D case (i.e. for a line, or circle with known radius/gradient angle), quadrants of the plane are repeatedly divided into sub-quadrants, and arranged into a search quadtree.

The size of the quadtree is prevented from being exponentially divided by pruning it down to "promising" quadrants (i.e. quadrants with many intersections), and as a result, the parameter search converges rapidly. Majority of the computation lies in determining whether curves in parameter space (transforms of image points) intersect any of the quadrants in the quadsearch tree. This "coarse" search allows easy quadrant negation, and subsequent focusing on a specific quadrant of interest for exponentially converging towards a parameter solution.

FHT could theoretically be exploited to operate on a parallel processing architecture, as it treats each pixel as an independent unit [3]. This could yield superior results in shape estimation algorithms that take advantage of multi-core processors and utilize FHT techniques for efficient object detection.

B.4 Ellipse Estimation

B.4.1 The Hough Transform Approach

Similar to circle detection, Hough transforms attempt to determine elliptical solutions in a 5 degree of freedom parameter space:

\[ x^2 + b'y^2 + 2d'xy + 2e'x + 2g'y + c' = 0 \] (B.6)

\[ b', d', e', g', c' \] are constant coefficients normalized with respect to the coefficient of \( x^2 \). This reduces the search space from 6 (present in the conventional ellipse equation) to 5 in Equation B.6.

Conventionally, a centrepoint is obtained by finding a line between two image points with parallel tangents. The centrepoint is then the midpoint of this line. However, like circle shape estimation, since Sobel edge detection and gradient angles are error prone, this method is not completely accurate. Furthermore, this method requires segmented pixels symmetric about the ellipse centre. If a highly occluded set of pixles are present on an arc of the ellipse, this method fails.
Yuen et al. proposed an alternative generic centrepoint extraction method that is robust to ellipse occlusions of varying degrees [26]. Suppose there are two image points \( P \) and \( Q \) that have non-parallel tangent lines, intersecting at point \( T \). \( M \) is the midpoint of \( PQ \). Then the ellipse must lie on a straight line \( TM \).

This is done for a variety of image point pairs, and all \( TM \) lines are processed in an accumulator array (similar to lines and circles). The common intersection point of all these lines is the centrepoint of the ellipse [13]. These values correspond to \( e' \) and \( g' \) in Equation B.6.

To determine the 3 remaining parameters - \( b' \), \( d' \) and \( e' \) - a space-efficient (from a computational storage point of view) peak detection algorithm based on the adaptive Hough Transform (AHT) method [15] is employed. The algorithm locates peak areas of the Hough transform space without computing background level in full detail. It is based on an iterative coarse to fine accumulate and search strategy.

It operates on the following simplification of equation B.6, based on substitution of \( e' \) and \( g' \), and translating the data relative to the centrepoint:

\[
x^2 + b''y^2 + 2d''xy + c'' = 0
\]  

(\text{B.7})

\( b'' \), \( d'' \) and \( c'' \) are simultaneously solved through the AHT method by first coarsely resolving each parameter (where the range is large). An accumulator peak determines a rough common value range for each parameter based on common intersection regions between image point pairs in parameter space (compare this to finding a unique intersection value in parameter space from image space). It then "focuses" on a convergence for each parameter by iteratively narrowing each parameter’s range by \( \frac{1}{3} \).

Note, that each parameter’s "resolution" can be varied independently during the iterative reduction routine - which, in the case of ellipse detection is useful - because the range of values for \( c \) is larger than those of \( b \) and \( d \), so computationally, it is better to first resolve \( c \) in fine detail and then using that to more rapidly fine tune \( b \) and \( d \).

### B.4.2 The Least Squares Fitting Approach

While Hough Transforms are robust against outliers (segmented data can be partially noisy), blurring and spurious peaks in the accumulator array make them somewhat ineffective. Furthermore, they are slow, require large memory, accuracy is moderate, and solutions are non-unique- even when curves have been pre-segmented [15]. While Fast Hough Transforms attempt to address some of these issues and
work effectively for lines and circles, the intrinsically large (5 degree of freedom) search space makes FHT inviable for ellipses.

Least-squares techniques (LSTs), on the other hand are computationally cheap and effective at ellipse fitting if and only if the image is free of noise and moderately occluded. They focus on finding a set of parameters that minimize some distance measure between the data points and the ellipse.

In a non-ideal scenario where a segmented ellipse is noisy and highly occluded, most least-squares methods fail and yield bogus elliptical solutions. A number of iterative refinement-rejection procedures (such as RANSAC, Kalman Filter, and the Bookstein algorithm) alleviate this problem, but do not eliminate it [15]. Furthermore, invalid ellipse rejection comes at an added computational cost.

The following discussion will be framed as such: theoretical foundations for LSTs will be presented, followed by practical algorithm implementation with edge detection and other image filters, with considerations for computational time and real-time performance.

Most least-squares fitting methods address ellipse parameter estimation by tackling the more general problem of conic fitting, since an ellipse is simply a cross sectional slice of a conic along a plane. The equation of a conic can be represented by a second order polynomial:

\[ F(A, X) = A \cdot X = ax^2 + bxy + cy^2 + dx + ey + f = 0 \]  \hspace{1cm} (B.8)

where \( A = \begin{bmatrix} a & b & c & d & e & f \end{bmatrix}^T \) and \( X = \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}^T \) [15]. \( F(A, X_i) \) is the algebraic distance of a point \((x, y)\) to the conic \( F(A, X) = 0 \). The fitting of a general conic is pursued by minimizing the sum of squared algebraic distances:

\[ \sum_i 1^N F(X_i)^2 \]  \hspace{1cm} (B.9)

of the curve to \( N \) data points \( X_i \). In order to avoid the trivial solution \( A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \), and multiple solutions of \( A \) which represent the same conic, the parameter vector \( A \) is constrained in some way. Many LSTs differ only in the form of constraint applied to \( A \) [15]:

- Paton suggests \( \|A\|^2 = 1 \)
- Bolles and Fischler suggest \( a=1 \)
- Rosin and Gander impose \( a + c = 1 \)
Appendix B. Hough Transforms And Least-Squares Fitting Techniques for Shape Estimation

- Rosin also investigates $f = 1$
- Bookstein proposes $a^2 + \frac{d}{2}b^2 + c^2 = 1$
- Agin and Taubin use data dependant quadratic constraints $||NA||^2 = 1$ when $N$ is the Jacobian

$$\begin{bmatrix} \nabla F(A, X_1) & \ldots & \nabla F(A, X_N) \end{bmatrix}^T$$

Notice that these constraints are either a) linear: $C \cdot A = 1$ or b) quadratic: $A^TCA = 1$, where $C$ is a $6 \times 6$ constraint matrix. The methods presented above do not restrict fitting to only be an ellipse; a hyperbola or parabola can also be returned for given elliptical data.

Some groups, on the other hand have focused on specifically recovering ellipses, rather general conics.

Porril and Ellis et al. use Bookstein’s method [15] to initialize a Kalman filter, which iteratively minimizes the gradient distance and reject non-ellipse fits by ensuring the discriminant (from parameters in $A$) $b^2 - 4ac < 0$ at each iteration. Although this constraint removes the non-specific, under-constrained nature of general conic fitting, it requires many iterations to be successful (especially in the presence of noisy data), and may fail to converge for extreme cases [15].

Haralick [15] takes a different approach - he guarantees that a conic is an ellipse by replacing the coefficients $a, b, c$ in $A$ with new expressions $p^2, 2pq, q^2 + r^2$ so that the discriminant $b^2 - 4ac$ is $-4p^2r^2$, which is guaranteed negative [15].

However, both these ellipse constraining techniques are ultimately limited to the Kuhn-Tucker conditions [26], i.e. that in general, the discriminant $b^2 - 4ac < 0$ does not guarantee a solution.

Direct Ellipse Fitting

Fitzgibbon et al. presents a noise, occlusion and affine invariant least-squares fitting approach which approaches the least squares fitting problem from a non-generalized conic to ellipse-constrained perspective [13].

Instead of imposing the inequality constraint $b^2 - 4ac < 0$, they instead impose an equality constraint $4ac - b^2 = 1$, which can be expressed quadratically as $A^TCA = 1$:

$$A^T \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} A = 1$$ (B.10)
where \( C \) is the constraint matrix. The goal is to minimize \( E = \|DA\|^2 \), subject to \( A^TCA = 1 \) [13]. \( D \) is the design matrix defined as an \( n \times 6 \) matrix \([x_1 \ x_2 \ \ldots \ x_n]^T\). Introducing the lagrange multiplier \( \lambda \) and differentiating \( E = \|DA\|^2 \), we arrive at the following system of simultaneous equations:

\[
2D^TDA - 2\lambda CA = 0 \quad (B.11)
\]

\[
A^TCA = 1 \quad (B.12)
\]

which can be rewritten as:

\[
SA = \lambda CA \quad (B.13)
\]

\[
A^TCA = 1 \quad (B.14)
\]

where \( S \) is the scatter matrix \( D^T D \). This system can be solved using a generalized eigenvectors matrix [13], which yields 6 eigenvalue-eigenvector pairs (as, in this case \( C \) is a \( 6 \times 6 \) matrix). An elliptical solution only physically exists if \( \lambda_i > 0 \), since \( S \) is positive definite.

A Cholesky decomposition of \( S \) [13] splits it into a lower triangular matrix with strictly positive diagonal entries, and its conjugate transpose. This decomposition reveals that there is only one positive diagonal entry for \( S \). This implies there is only one positive eigenvalue, and a unique solution must to Fitzgibbon’s approach to the ellipse fitting solution. Once 5 ellipse parameters, they can be scaled to 6, simply by scaling \( b, c, d, e, f \) by \( a \), since this still satisfies the equation of the conic (Equation B.8).

Fitzgibbon presents a theoretical LST method which can fit a set of pixel points to an ellipse efficiently. Wu et al. implemented his method with the aid of image filters and some rejection schemes to ensure ellipses were accurately detected in a real-time setting[13].

First some image pre-processing techniques were employed to prepare the image for quadratic constraint ellipse fitting: using a low pass Gaussian filter to remove noise, and a Canny Edge detection algorithm to extract edges. A simple non-ellipse pixel rejection scheme was also implemented to filter out an additional remaining noise or artifacts, by simply examining changes in gradient angle between adjacent pixels. If the difference in gradient angles were larger than some threshold, then the pixel was classified as a non-ellipse pixel and removed.
Fitzgibbon’s method was then implemented, followed by an ellipse validation step. For any point on the ellipse, the sum of its distances to the two focal points is a constant (the length of the major axis) \([13]\). Each pixel point fitted on an ellipse should adhere to this constant (within \(\pm 5\%\)). If any pixel’s sum of focal distances are greater than this tolerance level, the ellipse is invalidated.

Wu et al. also specified that the length of the pixel cluster must be at least 45 \% of the ellipse’s perimeter (to ensure non-ambiguity in ellipse parameter approximation). With non-ellipse pixel rejection, smoothing, edge detection, and a good ellipse-invalidation scheme, Wu et al. was able to practically demonstrate Fitzgibbon’s method with robustness to occlusion and noise.

Experimental results indicated that 97\% of ellipses were correctly detected, at a rate of 4 frames/sec. Most errors in the ellipse invalidation step were on the sub-pixel length \([13]\).
Appendix C

Pose Estimation Techniques With A Monocular Camera System

C.1 Perspective Geometry

Under perspective projection, a 3D points in space is projected to an image point, via a 3x4 projection $P$ matrix:

$$s\vec{m} = P\vec{x} = K\begin{bmatrix} R & t \end{bmatrix} \bar{x} = K\begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \bar{x}$$  \hfill (C.1)

Where $\bar{x} = [x_1 \ x_2 \ x_3 \ 1]$ and $\vec{m} = [m_1 \ m_2 \ 1]$ are the homogeneous forms of the world and image space points, respectively. $R$ and $t$ are the rotation matrix and translation vector from the world system to the camera system, $s$ is a non-zero scalar, and $K$ is the camera calibration matrix.

C.2 Camera Pose Estimation With Two Orthogonal Points

Theoretically, it is possible to recover camera pose with two orthogonal points (in world space). Wang et al. presents a noise-invariant method to accomplish this [35].

Assume two orthogonal points lie in the $X$ ($x_w = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$) and $Y$ ($y_w = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$) axes of a
Appendix C. Pose Estimation Techniques With A Monocular Camera System

normalized world coordinate system, whose origin lies at $\vec{o}_w = [0 0 0 1]^T$. Then, one can calculate the equivalent image space points via [35]:

$$s_x \vec{m}_x = P_n \vec{x}_w = [r_1 \ r_2 \ r_3 \ t] \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T = r_1 \quad (C.2)$$

$$s_y \vec{m}_y = P_n \vec{y}_w = [r_1 \ r_2 \ r_3 \ t] \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}^T = r_2 \quad (C.3)$$

$$s_o \vec{m}_o = P_n \vec{o}_w = [r_1 \ r_2 \ r_3 \ t] \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T = t \quad (C.4)$$

The rotation matrix can then be computed from:

$$r_1 = \pm \frac{\vec{m}_x}{||\vec{m}_x||}, \ r_2 = \pm \frac{\vec{m}_y}{||\vec{m}_y||}, \ r_3 = r_1 \times r_2 \quad (C.5)$$

where the rotation matrix $R = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix}$. There are potentially 4 solutions for $R$, but two can be discounted (since objects lie only in front of the camera). The formulation above assumes operating in a normalized world coordinate system. Wang et al. shows that for a preassigned world-coordinate system, the rotation matrix can be uniquely determined.

If there is metric information about the scene, the magnitude and direction of the translation vector $s_o \vec{m}_o$ also be determined (providing depth perception) [35].

Practically, $r_1$ and $r_2$ may not be orthogonal due to image noise, so the orthonormal constraint should be imposed during computation [35]. An SVD decomposition of $R$, where $R = U \Sigma V$ may provide the best approximation for an orthogonal compliant rotational matrix (in the least squares sense) [35]:

$$R = U \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} V \quad (C.6)$$

C.3 Circle Pose Estimation

de Ipina et al.’s TRIP tag system describes a robust circle pose estimation method, based on mapping the projection of a circle in image space (an ellipse) to a circle of known size in world space [35].
A well-segmented (noise-free) ellipse is first obtained from shape estimation and refinement techniques (such as RANSAC least-squares fitting), with 6 ellipse parameters. This ellipse $C$ is then normalized using the intrinsic camera matrix, $K$:

$$C_n = K^T \cdot C \cdot K$$  \hspace{1cm} (C.7)

Where the intrinsics matrix $K$ modelling a distorted camera lens is given by:

$$K = \begin{pmatrix}
k_u \cdot f & k_u \cdot f & u_o \\
0 & k_v \cdot f & v_o \\
0 & 0 & 1
\end{pmatrix}$$  \hspace{1cm} (C.8)

The normalized ellipse $C_n$, corresponding to a projected circle in an image plane, defines a 3D cone, whose vertex $O$ is in the centre of the projection of a pinhole camera. If $ax^2 + bxy + cy^2 + dx + ey + f = 0$ is the equation of the normalized ellipse, then the 3D cone is defined as:

$$ax^2 + bxy + cy^2 + dxz + eyz + fz^2 = P^T C_n P = 0$$  \hspace{1cm} (C.9)

where $P = [XYZ]^T$ is a point in the cone, and $C_n$ (the normalized ellipse) is the real symmetric matrix of a cone, defined as:

$$C_n = \begin{bmatrix}a & b/2 & d/2 \\
b/2 & c & e/2 \\
d/2 & e/2 & f\end{bmatrix}$$  \hspace{1cm} (C.10)

### C.3.1 Orientation Extraction

The orientation of the circle’s plane $\Pi_t$ is found by rotating the camera so that the intersection of the cone with the image plane becomes a circle [35]. This only happens, when the image plane $\Pi_i$ is parallel to the target plane. See [35] for a description of the image and target planes.

The rotation matrix $R_c$ describing the rotation of the image plane to the target plane can be seen as a composition of two successive rotations ($R_1$ and $R_2$):

$$R_c = R_1 \cdot R_2$$  \hspace{1cm} (C.11)
The first rotation, $R_1$, is determined by diagonalizing $C_n$, i.e., removing the coefficients with $xy, xz$ and $yz$ terms. This 3D rotation transforms the coordinate system, $OXYZ$, to the eigenvector frame, $OX'Y'Z'$, and the ellipse $C_n$ into $C'$. If $\lambda_1, \lambda_2, \lambda_3$ are the eigenvalues of $C_n$ with $\lambda_1 < \lambda_2 < \lambda_3$ and eigenvectors $\vec{e}_1, \vec{e}_2, \vec{e}_3$, the rotation matrix $R_1$ and transformed elliptical projection, $C'$ are described by:

$$R_1 = \begin{bmatrix} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 \end{bmatrix}$$  \hspace{1cm} (C.12)

$$C' = R_1^T \cdot C_n \cdot R_1 = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$  \hspace{1cm} (C.13)

The second rotation, $R_2$ is obtained by imposing the equality of coefficients of $x^2$ and $y^2$ in $C'$:

$$\theta = \pm \tan^{-1} \sqrt{\frac{\lambda_2 - \lambda_1}{\lambda_3 - \lambda_2}}$$  \hspace{1cm} (C.14)

$$R_2 = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$  \hspace{1cm} (C.15)

$$C'' = R_2^T \cdot C' \cdot R_2$$  \hspace{1cm} (C.16)

where $C''$ now describes a circle on the rotated image plane, $\prod_r$.

A vector normal, $\vec{n}$ describing the orientation of the circular target plane, in 3D camera coordinates, can be extracted from $R_c$:

$$\vec{n} = R_c \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T = \begin{bmatrix} R_{13} & R_{23} & R_{33} \end{bmatrix}^T$$  \hspace{1cm} (C.17)
C.3.2 Position Extraction

$C''$ represents a circle in the rotated image plane $\Pi_r$, with a radius $r_o$, and centre $(x_o, 0, 1)$ in terms of the coordinate system $OX''Y''Z''$:

$$
C'' = \begin{bmatrix}
1 & 0 & -x_o \\
0 & 1 & 0 \\
-x_o & 0 & x_o^2 - r_o^2
\end{bmatrix}
$$

(C.18)

If this is equated to Equation C.16, then $x_o$ and $r_o$ can be expressed in terms of eigenvalues of $C_n$:

$$
x_o^2 = \frac{(\lambda_2 - \lambda_1)(\lambda_3 - \lambda_2)}{\lambda_2^2}, \quad r_o^2 = \frac{-\lambda_3 \cdot \lambda_1}{\lambda_2^2}
$$

(C.19)

Similar triangles can then be used to scale and translate the circle $C''$ from the rotated image plane $\Pi_r$ to $\Pi_t$, in the $OX''Y''Z''$ coordinate system, by exploiting the following realtionships:

$$
d = \frac{\rho}{r_o} = \frac{\delta}{x_o}
$$

(C.20)

where $d$ is the distance between the camera and target plane, $\rho$ is the physical (known) radius of the circle, $r_o$ is the radius of the circle in $\Pi_r$, and $x_o$ and $\delta$ are deviations of the circle centres from the $Z''$ axis. Equating equation C.19 and equation C.20, $d$ can be solved in terms of the known radius, $\rho$:

$$
d = \sqrt{-\frac{\lambda_2^2}{\lambda_1 \cdot \lambda_3}} \cdot \rho
$$

(C.21)

Then, a translation vector $\vec{T}$ can be expressed in original coordinate system $OXYZ$:

$$
\vec{T} = R \cdot \begin{bmatrix}
\delta & 0 & d
\end{bmatrix}^T
$$

(C.22)

Notice, that since there are two solutions for $\theta$ in equation C.14, there are two possible solutions for $\vec{T}$ and $\vec{n}$. Physically, these correspond to solutions in or out of the camera, and the one which is projecting outward (i.e. where $\vec{T}$ yields a positive value for $z$) is selected [35].
Bibliography


