# Exploration of Haptic Force Feedback Algorithms for Confined Surgical Workspaces

Applications in Transoral Robotic Surgery and Cleft Palate Repair

ESC499 Thesis

Jiaheng liao

April 7th, 2025

Supervised by Prof. Looi, Thomas and Dr. Podolsky, Dale

The Wilfred and Joyce Posluns Centre for Guided Innovation and Therapeutic Intervention,

The Hospital for Sick Children

**B.A.Sc. Thesis**

# Abstract

The daVinci Surgical System is the most commonly used method for transoral robotics surgeries (TORS) like cleft repair despite having positional issues due to narrow surgical access ports. The space constraint on surgical space can cause unwanted collisions between surgical tools and between surgical tools and surrounding organs. This study explores the use of different haptic force feedback algorithms to enhance surgeon's spacial awareness with the goal to increase surgeon's accuracy while reducing collision between surgical tools and surgical tools and surroundings. A simple mouth cavity model and daVinci surgical tools are created in Blender with the Asynchronous Multi-Body Framework (AMBF) model. Closest instances of the surgical tools and mouth cavity walls are calculated based on the gradient of the signed distance field (SDF) of the surrounding walls. Linear, exponential and attractive force feedback algorithms are designed for the task of line tracing where the user will trace over a circular shape using the surgical tool. Results demonstrated that linear and attractive haptic force feedback had an improvement of 11.78% and 13.97% over no haptic feedback on average distances to the ground truth circle while exponential force feedback had a worsening performance of 7.71%. Based on the results collected from this study, a small amount of haptic showed improved user's accuracy while a larger haptic can worsen performance and be disruptive to the overall procedure. Overall, these results suggest that implementing haptic force feedback would improve the accuracy of surgeons in line tracing and cutting tasks.

## Acknowledgements

I would like to express my heartfelt gratitude to Prof. Thomas Looi and Dr. Podolesky Dale for granting me the invaluable opportunity to undertake this project at the Centre for Image Guided Innovation and Therapeutic Intervention at the Hospital for Sick Children. Their unwavering support, guidance, and access to vital resources have been essential to the success of this work.

I would also like to extend my sincere thanks to Ethan Tang. This thesis serves as an extension of the remarkable research he conducted during his master's thesis. Without the experimental setup he meticulously developed, this work would not have been possible. I am truly appreciative of his insightful guidance and the helpful tips he provided throughout the process.

# I. Table of Contents

# II. List of Figures

# III. List of Tables

# IV. List of Equations

## 1.0 Introduction

The daVinci Surgical System is a robotic platform used for minimally invasive surgery across various surgical fields. Previous studies from Centre for Guided Innovation and Therapeutic Intervention (CITIGI), explored the applications of the daVinci system in transoral robotic surgery (TORS) in the context of repairing infant cleft palates. Previous research showed success with the development of both the realistic silicone infant cleft model and haptic force feedback algorithm at providing training for surgeons and improving surgical accuracy. The development of haptic feedback also aims to address the lack of haptic feedback on many iterations of the commercially available daVinci Surgical Systems, in which many surgeons believed would be helpful to have [1] – [4].

The following study is an extension of previous studies with the goal to explore the use of Haptic Force Feedback Algorithms for Confined Surgical Workspaces Applications in Transoral Robotic Surgery and Cleft Palate Repair. The work will hopefully further contribute to 1) development of haptic feedback systems for robotic assisted surgery and 2) development of haptic feedback techniques and algorithms to aid surgical procedures.

## 2.0 Background

### 2.1 The DaVinci System

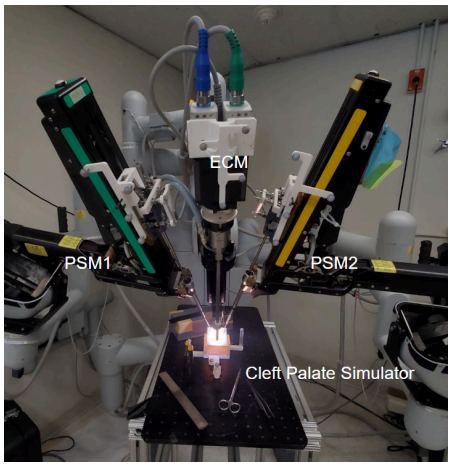



Figure 1. Image Shows PSMs on both sides and ECM in the middle with the suture box placed on the operating table

The daVinci System comprises of three instrument arms including two patient side manipulators (PSMs) and an endoscopic camera manipulators (ECM). The PSMs are teleoperation controlled by a surgeon using Master Tool Manipulators (MTMs) that interface with the surgeon's hands at the console. The PSMS, tool shafts and ECM must be put through the infant's oral cavity to access the surgical space for cleft palate surgeries. This set-up reduces the surgical workspace available for the surgeon [1].

During surgery, the surgeon has limited view of the surrounding environments from the ECMs and it is believed that additional information provided to the surgeon such as haptic feedback could include surgeon's spacial awareness during the procedure, reducing the number of collisions and improve procedure outcome. Bergholz et al. as well as previous studies at CITIGI support that haptic feedback generally reduced the average and peak forces experienced through surgery and that multimodal distance information produced from a signed distance field (SDF) improves surgical safety without introducing additional workload.

## 2.2 daVinci Research Kit (dVRK)

The daVinci Research Kit (dVRK) is an open source toolkit for the daVinci Classic Surgical System. dVRK is used to operate the daVinci Classic Surgical System available at CIGITI for exploring innovative concepts in minimally-invasive surgeries [5].

Figure 2. below shows the dVRK Flatform and the overall flow describing how it will be used in this experiment.

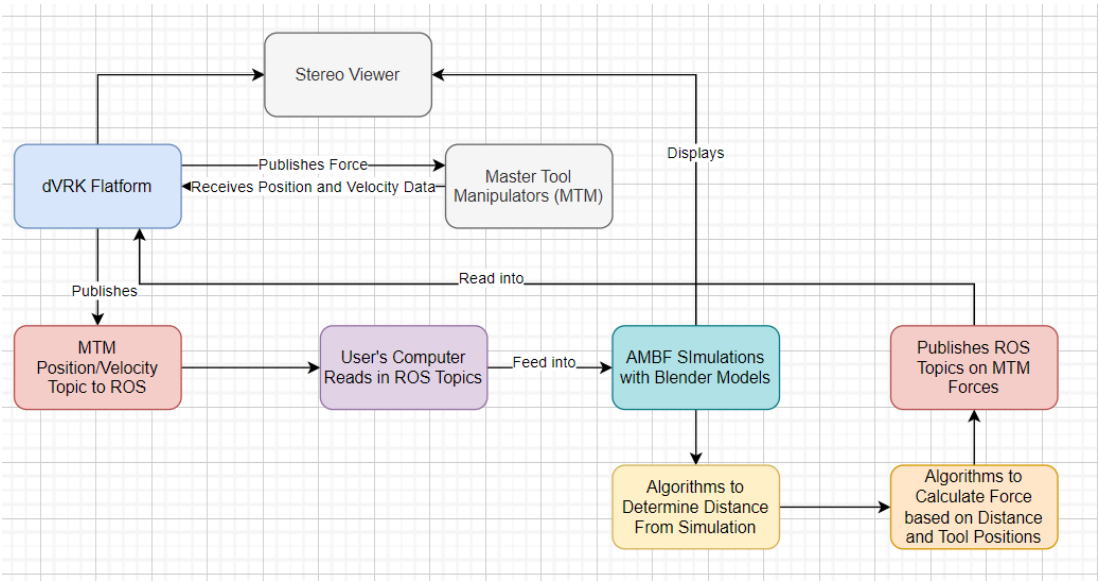



Figure 2. System Map showing dVRK Flatform, ROS Topics and Force Publication through AMBF Simulation

## 2.3 Cleft Palate Repair Surgery

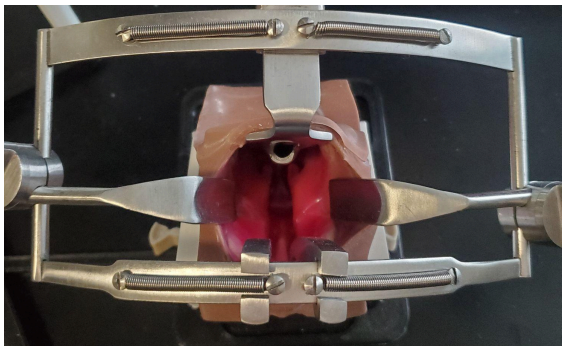


Figure 3. Realistic Silicone Model of Infant Mouth Cleft

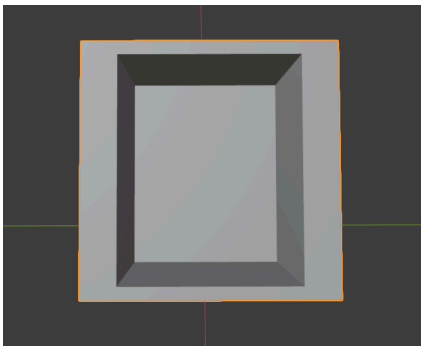


Figure 4. Simple Representation Model of Mouth Cleft Create in Blender, to be used in Simulation

Cleft palate is used to describe a child who is born with an opening in the roof of the month in either the hard palate in the front of the mouth, the soft palate in the back or both. A cleft palate makes it difficult for children to eat and speak with the food and liquid going up and through the opening into the nasal passage instead of going down the throat and into the stomach. For the child to be able to speak and swallow, the palate needs to seal off the inside passage of the nose from the throat. The only way to repair a cleft palate is through surgery [6].

The typical steps of the surgery are the following:

1. A device or brace is placed into the child's mouth to keep it open during the surgery
2. Incisions (cuts) on both sides of the palate along the cleft
3. Loosen the layer of tissue attached to the bone of the hard palate, allowing the tissues to be stretched
4. A cut is made along the gums. This allows the tissues of the palate to be stretched and moved toward the middle of the roof of the month
5. The inner nasal layer of tissues is closed using sutures (stitches)
6. Close the outer oral layer of tissue with sutures

Cleft Palate repair is a technically challenging procedure where visibility is limited and delicate dissection and interactions with delicate tissues within the confines of the infant oral cavity workspace is required. Surgical Robotics are commonly used due to the reachability of conventional surgical instruments while providing enhanced visualization, access and precision.

Previous studies at CITIGI demonstrated the effectiveness of a high-fidelity cleft palate simulator in providing a realistic, anatomical training tool. Figure 3. shows a realistic infant cleft palate mouth model and Figure 4. shows the simple box suture model representing the mouth cleft which will be used in simulation.

### 2.4 Virtual Fixture

Virtual fixtures is the use of overlaying simulated models of reality to provide additional sensory information to users of a teleoperated system. In the context of Robot Assisted Surgery (RAS),

this implied building a simulation model based on patients anatomy that allows for virtual interactions.

Virtual fixtures are used over direct force sensing or simulation as previous studies showed that collisions with the oral cavity and retractor were the most common and significant during simulated robotic-assisted cleft palate repair. It is important to discourage and avoid excessive contact with outer regions of the oral cavity for both the patient's safety and tools' longevity. Preoperation images techniques could also help adopt virtual fixture models to specific patient anatomy on a case to case basis. Sometimes a simulation model could also allow surgeons to respond to possible collisions that cannot be seen on ECM alone [6].

## 2.5 Asynchronous Multi-Body Framework (AMBF)

AMBF is an open-source 3D robotic simulator developed by Munawar et al in 2019. This multi-body framework provides a real-time dynamic simulation of multi-bodies such as robotics arms that can be paired with real-time haptic interactions from various input devices. AMBF serves as the basis for simulating the dVRK system in this study.

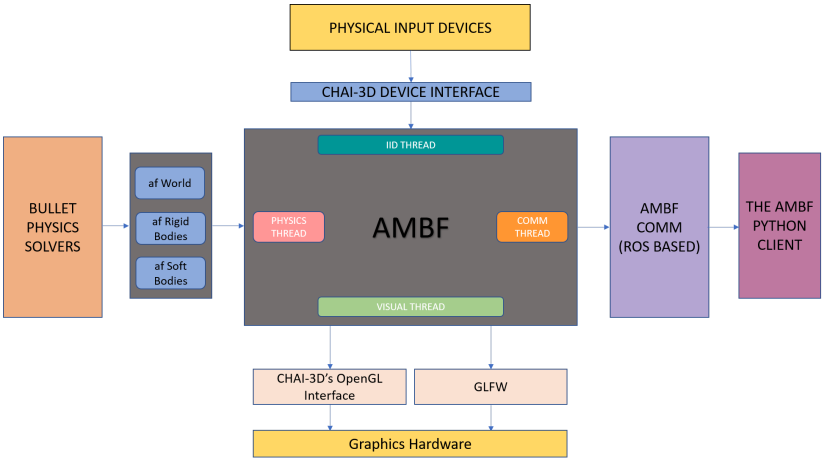



Figure 5. AMBF Work Flow

## 2.6 Signed Distance Fields (SDF) and Force Calculation

Signed distance Fields are 3D textures where each texel (3D pixel) stores the distance from the surface of an object. By convention, this distance is negative inside the object and positive outside.

A precomputed SDF provides an efficient method to compute distance queries and allows for fast calculation of the SDF gradient which can in turn help determine the point on the mesh closest to the tool position and thus determine the nearest wall to surgical tool distance.

To compute $F_{PSM}$, the roll link of the PSM is broken down into small segments. The closest PSM point to the collision mesh is identified through a query through the SDF in which the gradient is calculated. The direction of the applied force is then identified as the component of the SDF gradient that is perpendicular to the direction of the ECM view. The force applied is horizontal in nature and pushes the tool away from the collision body when it is in proximity [1].

Equation 1. Shows the Calculations to find Force Vector $F_{PSM}$

$$\vec{F}_{PSM} = -\nabla SDF_{mesh}(P_{query})$$

$$\vec{F}_{\parallel} = \frac{\vec{F}_{PSM} \cdot \vec{v}_{ECM}}{|\vec{v}_{ECM}|^2} \vec{v}_{ECM}$$

$$\vec{F}_{\perp} = \vec{F}_{PSM} - \vec{F}_{\parallel}$$

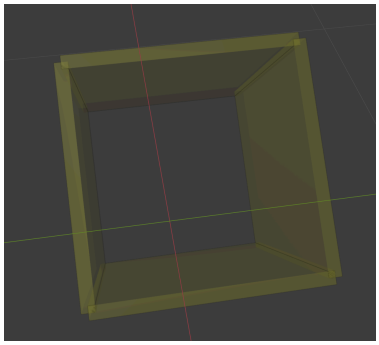


Figure 6. Image showing the SDF Collision Bodies Created in Blender

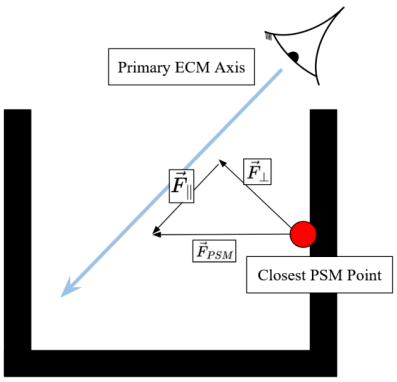



Figure 7. Diagram showing Force Vector $F_{PSM}$ generated through SDF [1]

Figure 6. shows the SDF collision bodies created in Blender to go along with the suture box model. Figure 7. shows the Force Vector $F_{PSM}$ generated through force computation algorithms.

All methods for virtual force calculation are carried out in Python with the *rospy* library being used to interface the computer and dVRK MTMs.

### 2.7 Cutting Tasks

Studies like Y. He *et al* and *H. Ishida et al* showed that virtual fixtures and SDF can be beneficial in the case of improving situational awareness in skull base surgery and bone cutting. Y.He *et al.* used a virtual fixture to restrain and guide the motion of the robot, ensuring motion safety for nasal endoscopy surgery. *H. Ishida et al* showed that haptic feedback mechanism enhances the safety of drilling around critical structures compared to systems lacking haptic assistance. Further, surgeons were able to safely skeletonize the critical structures without breaching any critical structure even under obstructed view of surgical sites. While the goal for this study was to explore haptic force feedback to avoid collision in suturing tasks with applications to TORS, indirectly, the algorithms could also be beneficial for cutting tasks around sensitive regions. Though, a more thorough study should be conducted for a conclusive result.

# 3.0 Methodology

## 3.1 Experiment Set-up

The experiment uses the dVRK platform, ROS and AMBF simulations with models built using Blender. Figure 8. shows the Master Tool Manipulator as well as the stereo viewer which will be used to conduct the experiment. Figure 9. shows the AMBF simulation featuring the circle which will be traced and the surgical tools. Only the right surgical tool will be used for tracing.

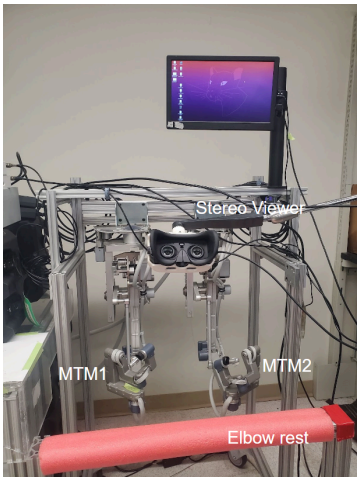



Figure 8. Master Tool Manipulator and Stereo Viewer. The Red Foam Provides Elbow Resting Support

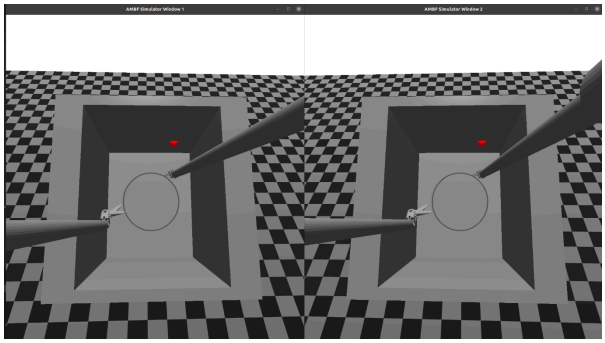


Figure 9. AMBF Set-up in Simulation environment. Red sphere shows the closest point on the wall to the right tool tip. The circle is outlined in the image for better visibility

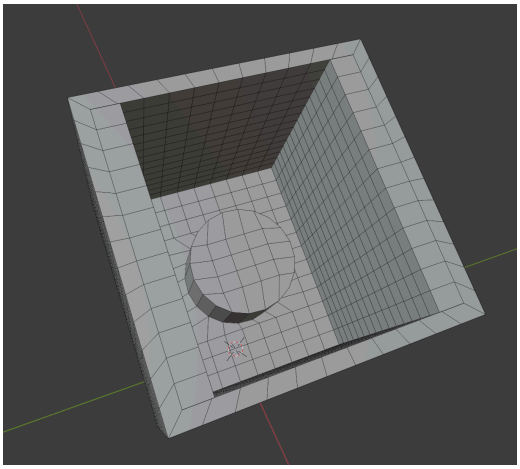


Figure 10. Image of Cylinder Created from the Suture Box Model for Circle Tracing Task in Blender

For the line tracing task, the entire procedure will be untimed. The position of the tool and forces applied to the controller will be tracked and analyzed. The exact position of the tool is taken at the pitch joint (Figure 11).

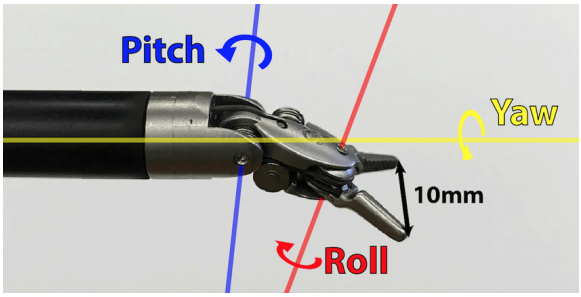



Figure 11. Image showing the Yaw, Pitch and Roll Joints of the Surgical Tool [12]

### 3.2 Experiment Procedure

1. Roscore, dVRK startup script and simulation startup script is ran on the computer
2. The right tool tip is placed at the top of the circle trace in simulation
3. The data logging script is started
4. The user traces the circle in simulation through stereo viewer
5. Data logging script is stopped, data is saved
6. The procedure is repeated five times for the baseline and each force feedback algorithm

### 3.3 Force Algorithms Designed

A total of three force algorithms are chosen and used for this experiment. The algorithms are Linear Force Feedback, Exponential Force Feedback, and Linear Attractive Force Feedback. The algorithms are chosen for their simplicity, ease of use, and flexibility.

#### 3.3.1 Linear Force Feedback Algorithm

In the linear force feedback algorithm, the force scales linearly with the distances to the wall.

Equation 2. Showing Linear Force Calculation

$$F = -k * displacement * v_{norm}$$

Where:

- $F$ is the force applied on the controller
- $k$ is the scaling factor, it is set to 300 for the line tracing task

- Displacement is the closest distance of the tooltips from the wall
- $v_{norm}$ is the vector pointing from the tooltips position towards the closest point to the wall normalized

The resultant $F$ is a 3 by 1 vector and the equation is multiplied by a factor of negative one so that the force is pointed in the opposite direction as $v_{norm}$, pushing the controller and surgical tools away from the wall.

### 3.3.2 Exponential Force Feedback Algorithm

Similarly, the exponential force feedback scales the force exponentially with the distance to the wall.

Equation 3. Showing Exponential Force Calculation

$$F = F_o * e^{(-\lambda * displacement)} * v_{norm}$$

Where:

- $F_o$ is the initial force, this is set to 1 for the line tracing task
- $\lambda$ is the scaling factor, this is set to 1 for the line tracing task
- Displacement is the closest distance of the tooltips from the wall
- $v_{norm}$ is the vector pointing from the tooltips position towards the closest point to the wall normalized

### 3.3.3 Linear Attractive Force Feedback Algorithm

Linear attractive force uses a similar formula as above.

Equation 4. Showing Attractive Force Calculation

$$F = -k * displacement * v_{norm}$$

Where:

- $F$ is the force applied on the controller
- $k$ is the scaling factor, this is set to 100 for the line tracing task
- Displacement is set to $distance - d_{rest}$ where distance is the magnitude of the vector $v$ and $d_{rest}$ is set to be the origin of the circle

- $v_{norm}$ is the vector pointing from the tooltips position towards the closest point to the wall normalized

For the circle tracing task, the linear attractive force feedback will apply a force on the controller toward the centre of the circle when the tool tip is too far away from the centre. The linear attractive force feedback will also apply a force on the controller pushing it away from when the tool tip is too close to the center. The offset from the ground truth for the force algorithm to activate is set to 2 millimeters. There was also a force limiter set on all forces published. The force is capped at 5 N maximum.

### 3.4 Rationale for Choosing the Algorithms

There were additional algorithms explored but not included in the study, including region based distance force feedback, attractive exponential force feedback and damping force feedback algorithms. The idea for a region distance force feedback was to have different force scaling at different distances from the side wall. Attractive exponential force was created to test whether linear or exponential attractive force had a better effect on the spatial awareness of the operation space. The rationale behind damping force was that some friction would make the controllers more responsive to movement. However, when damping force was applied, it was too infrequent and disruptive due to the refresh rate of the system. Region based distance feedback algorithm and attractive exponential force feedback was not included in the study because it was more difficult to tune these controls and they require longer time to get used to while.

### 3.5 Hypothesis

The implementation of force feedback algorithms should decrease the average distance of the points from ground truth, implying an increase in the user's accuracy.

## 4.0 Results and Discussion

Table 1. No Force Feedback Compared to Different Force Feedback Algorithms at Average Distance to Ground Truth

| | No Force Feedback | Linear Force Feedback | Exponential Force Feedback | Attractive Force Feedback |
|---|---|---|---|---|
| **Average distance to Ground truth in the X and Y Direction (mm)** | 3.226 | 2.886 | 3.496 | 2.831 |
| **Comparison to Baseline (%)** | N/A | 11.78% | -7.71% | 13.97% |

Table 2. Average Force Applied on Controller for Different Force Algorithms

| | Linear Force Feedback | Exponential Force Feedback | Attractive Force Feedback |
|---|---|---|---|
| **Average Force Applied on Controller (N)** | 0.6978 | 1.8474 | 0.3148 |

The results demonstrated promising results for the linear force feedback algorithm and attractive force feedback algorithms where they showed a 11.78% and 13.97% improvement over no force feedback at average distance to the circular ground truth over the average of five trials. The average forces applied on the controllers were 0.6987N, 1.8474N and 0.3149N for the linear force feedback, exponential force feedback and attractive force feedback respectively. The exponential force algorithm showed worsening results where the average distance from ground truth circle was 7.71% worse than no baseline.

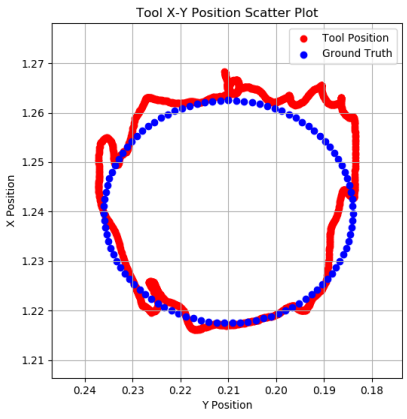



Figure 12. X-Y Position Plot with No Force Feedback

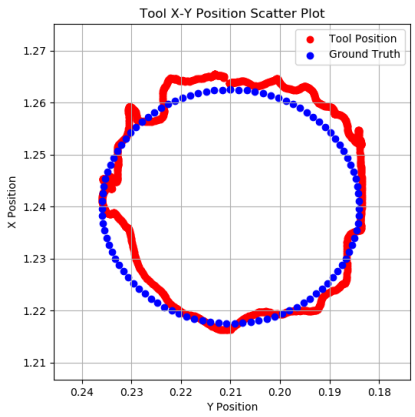



Figure 13. X-Y Position Plot with Attraction Force Feedback

Figure 12. and Figure 13 displays the x-y position plot from a trace with no force feedback and force feedback. Note that the direction of tracing is counterclockwise and the experiment is conducted on the dominant hand which is the right hand. On the no force plot, there are noticeable more sudden movements of the tool tip near positions (1.26, 0.225), (1.25, 0.235) and (1.26, 0.195). In comparison, the attraction force plot is a lot smoother even in regions with sudden deviations in positions (1.23, 0.23) and (1.26, 0.19).

The sudden and sharp deviations could be a result of user fatigue, loss of concentration and hand instability. Each circle trace took around 20~30 seconds to complete and it would require a great amount of concentration. Long, repetitive movement can introduce fatigue and further hand instability and the loss of attention. In addition, in contrast to writing on a piece of paper where the palm of the hand can rest on the table, the stabilizing point when using dVRK system is the elbow. The extended resting point can further introduce instability.

There seems to be a correlation between the amount of force applied and the distance to ground truth. Attractive force feedback applies the least amount of force but has the closest result to the ground truth. Exponential force applies the most amount of force and has the worst result in comparison to other force feedback algorithms and the baseline case of no haptic feedback. The reason for thai could be due to that a relatively small amount of force can serve as a reminder for the user for when the tool tip has deviated from the ground truth. Exponential force can be too subtle at times and be disruptive to the user's motion, where the surgical tip is pushed toward the opposite side by the haptics.

## 4.1 Weaknesses and Limitations

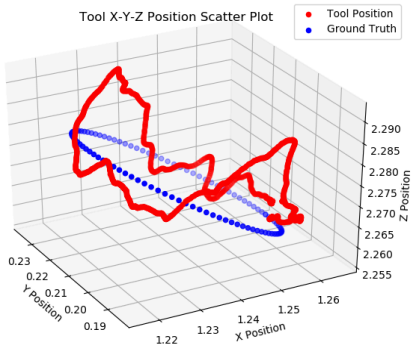



Figure 14. 3D Position Plot with Attraction Force Feedback

The data analysis in this study is only done in 2D partly due to the tracing surface being a 2D surface and that the algorithms kept the tools within the suture box space and not a 2D surface. Tracing in 3D was also difficult due to the depth perception of the simulation being not perfect and sometimes it would be hard to pinpoint the exact tool tip position in simulation.

The tracing was only done on the right hand going counterclockwise and the experiment was conducted by the experimenter. Each method was run five times before switching to the next force algorithm. There could be fatigue build up towards the later part of the experimentation, resulting in more hand instability.

The circle tracing shape was chosen arbitrarily. Nonetheless, it could be seen that the circle shape could skew the experiment results toward the attraction force. Attraction force is based on the center of the circle and provides a cubical area for operation. This means that the attraction force acts on all parts of the circle tracing process even if the tool were to deviate a little from the right path. However, linear force feedback and exponential force feedback provided a cubical area for operation. Because of this, force feedback can be felt at points of the circle nearest to the four walls and not much at parts of the circle near the corner of the walls unless there is a large deviation from the correct path. Despite this, linear force feedback provided strong results in circular trace, demonstrating that perhaps only occasional haptic to steer the user onto the correct right is sufficient.

### 4.2 Strengths

The experiments demonstrated strong results for improving the user's accuracy at line tracing tasks compared to no force feedback. Attractive force feedback was the best of the three algorithms, soon followed by linear force feedback and the worst of the three being exponential feedback. The presence of light haptic force provided a good reminder to the user if the tool were to veer too far off course. An excess or poorly tuned force feedback, like the case with exponential force, could be more disruptive than helpful. Proper tuning, training and familiarity will be required if the algorithms were to be implemented for surgical uses.

### 4.3 Extensions

Due to the lack of time and one of the surgical tools' wire slack during the experiment, the second part of the experiment was not conducted and was cut from the report. The experiment would test no force along with the three algorithms at avoiding wall collisions during three consecutive sutures with applications to TOR. The results from this study suggests that implementing haptic force feedback would decrease collisions in suturing methods and improve the safety of robotic assisted surgery in confined workspaces.

## 5.0 Conclusion

This study presents comparison of no haptic force feedback and linear, exponential and attractive force feedback in line tracing tasks within confined surgical spaces. The results demonstrated linear and attractive force feedback reduces the amount of deviations from the ground truth, improving surgeon's accuracy. The presence of a light force haptic could increase surgeon's awareness and alertness but an excessive haptic force could worsen one's performance. Future works will involve accessing the haptic force feedback performance on suturing tasks within confined surgical spaces to further validate the algorithm design and function for TORS applications.

While this study focuses on haptic force feedback algorithms for transoral surgery, the algorithms may apply to incisions tasks as well. Though, more thorough study should be conducted for a more conclusive result.

## 6.0 References


[1] E. Tang, T. Looi, and D. Podolsky, "Virtual Fixtures for Confined Surgical Workspaces: Applications in Transoral Robotic Surgery and Cleft Palate Repair," *Posluns Center for Image Guided Innovation and Therapeutic Intervention, Hospital for Sick Children, and University of Toronto*, 2025.

[2] E. Tang, G. Maguire, T. Looi, and D. Podolsky, "Robotic cleft palate repair using novel 3 mm wristed tools," in *Proc. Institute for Craniofacial and Cleft Innovation*, Hospital for Sick Children, Toronto, ON, Canada, 2024.

[3] D. J. Podolsky, D. M. Fisher, K. W. Y. Wong Riff, T. Looi, J. M. Drake, and C. R. Forrest, "Infant Robotic Cleft Palate Surgery: A Feasibility Assessment Using a Realistic Cleft Palate Simulator," *Plastic and Reconstructive Surgery*, vol. 139, no. 2, pp. 455e–465e, Feb. 2017.

[4] M. Bergholz, M. Ferle, and B. M. Weber, “The benefits of haptic feedback in robot assisted surgery and their moderators: A meta-analysis,” Scientific Reports, vol. 13, no. 1, Nov. 2023. doi:10.1038/s41598-023- 46641-8

[5] P. Kazanzides et al., “An open-source research kit for the Da Vinci® Surgical System,” 2014 IEEE International Conference on Robotics and Automation (ICRA), May 2014. doi:10.1109/icra.2014.6907809

[6] "Cleft Palate Repair: The Surgery," Nationwide Children's Hospital. [Online]. Available: https://www.nationwidechildrens.org/family-resources-education/health-wellness-and-safety-resources/helping-hands/cleft-palate-repair-the-surgery. [Accessed: Apr. 4, 2025].

[7] A. Munawar, Y. Wang, R. Gondokaryono, and G. S. Fischer, "A real-time dynamic simulator and an associated front-end representation format for simulating complex robots and environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019. [Online]. Available: https://www.researchgate.net/publication/338948596_A_Real-Time_Dynamic_Simulator_and_an_Associated_Front-End_Representation_Format_for_Simulating_Complex_Robots_and_Environments. [Accessed: Feb. 8, 2025].

[8] H. Ishida et al., “Improving surgical situational awareness with signed Distance Field: A pilot study in virtual reality,” 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8474–8479, Oct. 2023. doi:10.1109/iros55552.2023.10342004

[9] H. Ishida *et al.*, "Haptic-Assisted Collaborative Robot Framework for Improved Situational Awareness in Skull Base Surgery," *arXiv:2401.11709*, Jan. 2024.

[10] Y. He *et al.*, "Human–Robot Cooperative Control Based on Virtual Fixture in Robot-Assisted Endoscopic Sinus Surgery," *Applied Sciences*, vol. 9, no. 8, article 1659, Apr. 2019.

[11] A. Munawar, Y. Wang, R. Gondokaryono, and G. S. Fischer, "A real-time dynamic simulator and an associated front-end representation format for simulating complex robots and environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019. [Online]. Available: https://www.researchgate.net/publication/338948596_A_Real-Time_Dynamic_Simulator_and_an_Associated_Front-End_Representation_Format_for_Simulating_Complex_Robots_and_Environments. [Accessed: Feb. 2025].

[12] D. Seita, "The dVRK end-effector ('large needle driver') and its three axes of rotation,"

ResearchGate, Figure 2 in "Fast and Reliable Autonomous Surgical Debridement with Cable-Driven Robots Using a Two-Phase Calibration Procedure," Sep. 2017. [Online]. Available: https://www.researchgate.net/figure/The-dVRK-end-effector-large-needle-driver-and-its-three-axes-of-rotation-We-denote_fig5_319952851. [Accessed: Feb. 8, 2025].

## 7.0 Appendix

---

**Appendix A - Linear Distance Force Algorithm**

```
def hookean_force_sdf(self, v_p, dist, d_rest=0.1275, k=300.0):
    # Normalize the vector direction
    v_norm = v_p / np.linalg.norm(v_p)

    # print(v_norm)
    # Compute displacement from the rest distance
    # print("dist to wall",dist - d_rest )
    if dist < d_rest:
        displacement = dist - d_rest

        # Hookean force formula: F = -k * (dist - d_rest) * v_norm
        f = -k * displacement * v_norm
    else:
        f = v_norm * 0

    # print(f)
    return f
```

**Appendix B - Exponential Distance Force Algorithm**

```
def exp_decay_force_sdf(self, v_p, dist, F0=1.0, lambda_val=1.0):
    # Normalize the direction vector
    v_norm = v_p / np.linalg.norm(v_p)

    if dist < 0.115:
        f = F0 * np.exp(-lambda_val * dist) * v_norm
    else:
        f = v_norm * 0
    # Exponential decay force: F = F0 * exp(-lambda * dist) * v_norm

    return f
```

**Appendix C - Attraction Force Algorithm**

```
def attraction_force(self, d_rest=0.026, k=100):
    # print("last position: ", self.last_position)
    # print("roll_position:", self.roll_position)
    v_p = np.transpose([1.24, 0.21, 2.271] - self.roll_position)
    # Normalize the vector direction
    v_norm = v_p / np.linalg.norm(v_p)
    dist = np.linalg.norm(v_p)
    # print("dist:", dist)

    # Compute displacement from the rest distance
    if dist > d_rest:
        displacement = dist - d_rest
        # print("distance from rest", displacement)
        # Hookean force formula: F = -k * (dist - d_rest) * v_norm
        f = -k * displacement * v_norm
    else:
        f = [0, 0, 0]

    return f
```

**Appendix D - Variable Zone Force, Attractive Exponential Force and Damping Force Algorithms**

```
def variable_zone_force(self, v_p, dist, d_min=0.15, d_max=0.25, k1=15.0, k2=2.0):
    # Normalize the direction vector
    v_norm = v_p / np.linalg.norm(v_p)
    # print(dist)

    # Force varies by distance zone
    if dist < d_min:
        f = k1 * self.roll_vel
    elif d_min < dist:
        f = k2 * self.roll_vel
    else:
        f = np.zeros_like(v_p)  # No force outside range

    return f

def attraction_force_exp(self, d_rest=0.027, F0=4.0, lambda_val=7.0):
    # print("last position: ", self.last_position)
    # print("roll_position:", self.roll_position)
    v_p = np.transpose(self.last_position - self.roll_position)
    # Normalize the vector direction
    v_norm = v_p / np.linalg.norm(v_p)
    dist = np.linalg.norm(v_p)
    # print("dist:", dist)

    # Compute displacement from the rest distance
    if dist > d_rest:
        displacement = dist - d_rest
        # print("distance from rest", displacement)
        f = -F0 * np.exp(lambda_val * displacement) * v_norm
```

```
        else:
            f = [0, 0, 0]

        return f

def damping_force(self, c=5):
        # Normalize velocity direction
        # print("roll_vel", self.roll_vel)
        # if np.linalg.norm(self.roll_vel) > 0:
        #     v_dir = self.roll_vel / np.linalg.norm(self.roll_vel)
        # else:
        #     v_dir = np.zeros_like(self.roll_vel)

        # Damping force formula: F = -c * velocity
        f = c * self.roll_vel

        return f
```

**Appendix E - Data Recording and Plotting Script**

```
if self.plot_xy_pos:
        # shift distance values to zero seconds starting at program start
        oldest_time = self.pos_list[0][3]
        self.pos_list[:, -1] -= oldest_time

        print('program finished')
        # print(self.roll_dist)

        time_data = self.pos_list[:, -1]
        x_pos = self.pos_list[:, 0]
        y_pos = self.pos_list[:, 1]
        z_pos = self.pos_list[:, 2]

        # Generate circular points centered at (1.29, -0.21)
        center_x, center_y, center_z = 1.24, 0.21, 2.271
        radius = 0.026
        theta = np.linspace(0, 2 * np.pi, 100)  # 100 points around the circle
        circle_x = center_x + radius * np.cos(theta)
        circle_y = center_y + radius * np.sin(theta)
        # circle_z = center_z + radius * np.sin(theta)
        circle_z = np.full_like(circle_x, center_z)

        # Rotation matrix for 30 degrees (pi/6 radians)
        angle = np.radians(30)
        R_y = np.array([[np.cos(angle), 0, np.sin(angle)],
           [0, 1, 0],
           [-np.sin(angle), 0, np.cos(angle)]])  # FIXED ROTATION

        # Rotate points
```

```
rotated_points = R_y @ np.vstack((circle_x - center_x, circle_y - center_y, circle_z - center_z))

# Shift to the desired center position
rotated_x = rotated_points[0, :] + center_x
rotated_y = rotated_points[1, :] + center_y
rotated_z = rotated_points[2, :] + center_z

# Compute distance from tool positions to closest ground truth point
distances = []
for i in range(len(x_pos)):
    tool_point = np.array([x_pos[i], y_pos[i], z_pos[i]])
    ground_truth_points = np.vstack((rotated_x, rotated_y, rotated_z)).T
    min_dist = np.min(np.linalg.norm(ground_truth_points - tool_point, axis=1))
    distances.append(min_dist)

# Create figure and axis
fig, ax = plt.subplots(figsize=(6, 6))
# fig = plt.figure(figsize=(8, 6))
# ax = fig.add_subplot(111, projection='3d')

# Plot X vs. Y as scatter points
ax.scatter(y_pos, x_pos, color='r', marker='o', label="Tool Position")
# ax.scatter(y_pos, x_pos, z_pos, color='r', marker='o', label="Tool Position")
# Plot Ground Truth Circular Points
# ax.scatter(circle_y, circle_x, circle_z, color='g', marker='o', label="Ground Truth")
ax.scatter(rotated_y, rotated_x, color='b', marker='o', label="Ground Truth")
# ax.scatter(rotated_y, rotated_x, rotated_z, color='b', marker='o', label="Ground Truth")
# Set axis limits
# ax.set_ylim(1.2, 1.5)  # X-axis range
# ax.set_xlim(0.1, 0.41)  # Y-axis range
```

```
# Reverse X-axis to plot in decreasing order
ax.invert_xaxis()

# Add labels, title, and grid
ax.set_xlabel("Y Position")
ax.set_ylabel("X Position")
# ax.set_zlabel("Z Position")
ax.set_title("Tool X-Y Position Scatter Plot")
# ax.set_title("Tool X-Y-Z Position Scatter Plot")
ax.legend()
ax.grid(True)

fig2 = plt.figure(figsize=(6, 6))
ax2 = fig2.add_subplot(111, projection='3d')
ax2.scatter(y_pos, x_pos, z_pos, color='r', marker='o', label="Tool Position")
ax2.scatter(rotated_y, rotated_x, rotated_z, color='b', marker='o', label="Ground Truth")
ax2.invert_xaxis()
ax2.set_xlabel("Y Position")
ax2.set_ylabel("X Position")
ax2.set_zlabel("Z Position")
ax2.set_title("Tool X-Y-Z Position Scatter Plot")
ax2.legend()
ax2.grid(True)

# Show the plot
plt.show()

sum = 0
index = 0
for i in range(len(distances)):
    sum += distances[i]
```

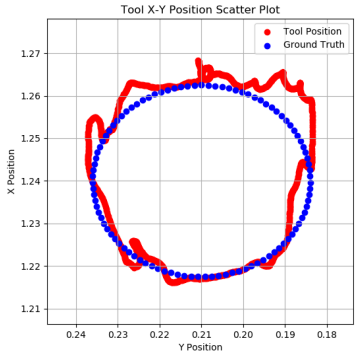
```
        average = sum / len(distances)
        print(average)

        sum = 0
        for i in range(len(mag_list)):
            sum += mag_list[i]
            if mag_list[i] != 0:
                index += 1
        print('avg force applied', sum / index)

        # Save data to CSV
        trial_time = datetime.datetime.now().strftime("%Y-%m-%d_%H-%M-%S")  # Get
current timestamp
        filename = f"trial_{trial_time}.csv"  # Create filename

        # Create DataFrame and save
        df = pd.DataFrame({'Time': time_data, 'Y Position': y_pos, 'X Position': x_pos, 'Z
Position': z_pos, 'Deviation': distances, "Forces":mag_list})
        df.to_csv(filename, index=False)
        print(f"Data saved as {filename}")
```

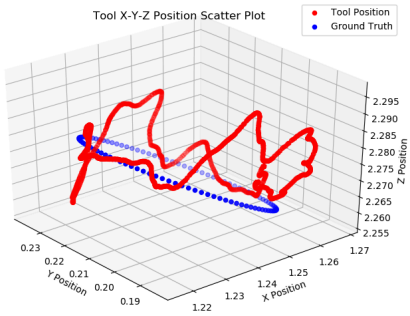
**Appendix F - Raw Data**

| | No Force Feedback | Linear Force Feedback | Exponential Force Feedback | Attractive Force Feedback |
|---|---|---|---|---|
| **Trial 1** | 0.003835424412 | 0.002180179457 | 0.003754790573 | 0.003464491437 |
| **Trial 2** | 0.002723998187 | 0.002529149997 | 0.003934297708 | 0.002731038126 |
| **Trial 3** | 0.003557909714 | 0.003588150007 | 0.003912972237 | 0.002500415468 |
| **Trial 4** | 0.002966439984 | 0.002790328161 | 0.002652765075 | 0.002580838663 |
| **Trial 5** | 0.003046298018 | 0.003342533329 | 0.003223454902 | 0.002876157711 |
| **Averages** | 0.003226014063 | 0.002886068203 | 0.003495656099 | 0.002830588274 |
| **Comparison to Baseline** | N/A | 11.78% | -7.71% | 13.97% |

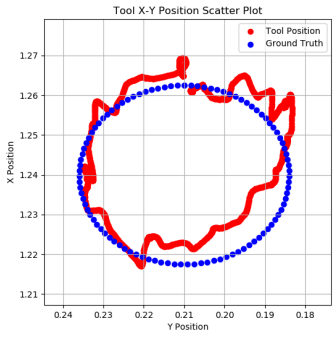
| | Linear Force Feedback | Exponential Force Feedback | Attractive Force Feedback |
|---|---|---|---|
| **Trial 1** | 0.6875253916 | 1.83210122 | 0.3384509699 |
| **Trial 2** | 0.5642982312 | 1.842983628 | 0.2850507577 |
| **Trial 3** | 0.712041138 | 1.835218322 | 0.3773661121 |
| **Trial 4** | 0.6148314594 | 1.83379244 | 0.3369278553 |
| **Trial 5** | 0.9103226142 | 1.8926912 | 0.2366040237 |
| **Averages** | 0.6978037669 | 1.847357362 | 0.3148799437 |

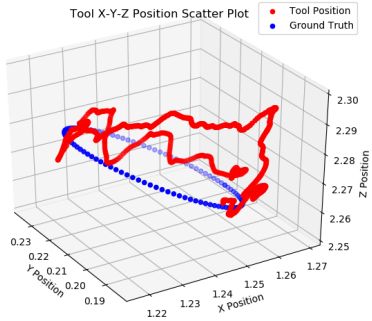
**Baseline, No Force**



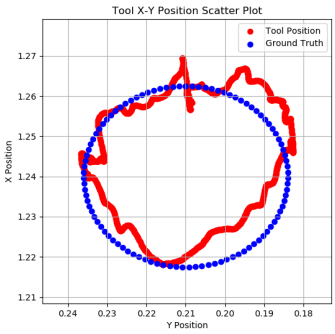



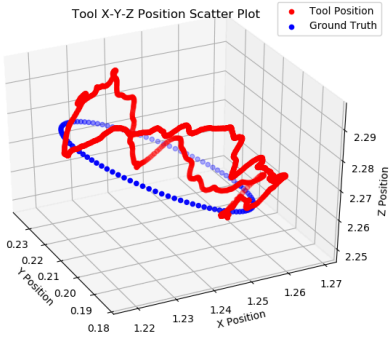

## Linear Distance Force




## Exponential Distance Force




## Attractive Force