Tele-operation Control of a High Dexterity Robot for Vitreoretinal Surgery

Team 22: Muhammad Hadi, Mojtaba Esfandiari

Mentors: Ali Ebrahimi, Dr. Iulian Iordachita, Adnan Munawar, Alireza Alamdar
Non-CIS II affiliates: Trent Tang, Abdullah Al Armouti

Abstract

In an effort to improve intuitive control of a steady hand robot for vitreoretinal surgery, we implemented teleoperative control of a hybrid robot using a Phantom Omni. The system offered 7 D.O.F, with a 5 D.O.F Eye Robot (SHER2) and a 2 D.O.F continuum snake robot (I²RIS), the combination of which offers greater manipulability and dexterity in the vitreous sphere during surgery. We accomplished segmented teleoperation by teleoperating the eye robot around an RCM point/trocar to situate the snake robot, and subsequently moving the snake robot once the idea position is achieved.

Background

Vitreoretinal surgery is a highly delicate and risky type of surgery, one that revolves around operating deep within the eye, on the surface of retina. Types of surgeries that fall within broader umbrella are epiretinal membrane peeling and retinal vein cannulation.

However, there are several factors that could be a potential risk during such surgeries. The primary issue arises from physiological hand tremor, which, as faint as it may be, can still result complications during surgery. For example, when peeling back scar tissue during an epiretinal membrane peeling, any slight jerks form hand tremor can result in retinal tears. Similarly, exerting forces > 7.5 mN to the retina could result in retinal tears, forces which are too fine to be felt by a human. All these factors result in an exceptional level of training that surgeons must undergo to be able to perform these surgeries [1][2].

To resolve the above stated issues, researcher at Johns Hopkins University, AMIRO LCSR, came up with a surgical robotic system, a Steady Hand Eye Robot (SHER), which performs robot-controlled surgery, allowing surgeons to mitigate (filter) physiological tremor affects and provide them with haptic force feedback. Systems like SHER have helped address a lot of the issues that surgeons faced in the OR. This impedance style robot drastically reduced the effects of tremor in surgery.

However, even with SHER, there are still some operative procedures that may not be easily executable, given the limited flexibility of the end effector. A straight needle attached to the end-effector of SHER does not provide enough dexterity for surgeons operating inside the eye, with surgeons often having to move the eye to navigate. Therefore, a 2 D.O.F Integrated Robotic Intraocular Snake (IRIS) was designed and attached to the SHER end-effector. This eliminated the need to move the eye by employing a more flexible end-effector [3].

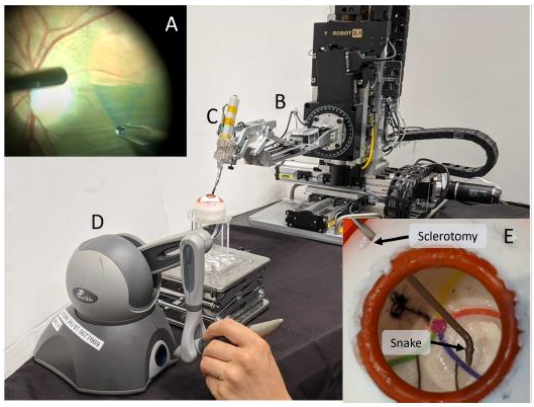

Figure 1. Envisioned high dexterity intraocular manipulator: (A) Epiretinal membrane peeling; (B) Steady Hand Eye Robot; (C) Integrated robotic intraocular snake robot; (D) Phantom Omni; (E) Distal snake-like tool-end inside eye phantom [4].

Existing approaches include teleoperation of both the snake and the eye robot, but separately, as well as a cooperative approach, which involves moving the eye robot by hand, and teleoperating the snake after moving it to the desired position. A recently accepted paper from the AMIRO lab successfully simulated the teleoperation of the 7 D.O.F system in question with a 5 D.O.F Phantom Omni, and we seek to improve upon that paradigm.

Technical Summary

I. Methods/Approach

1. Equipment

Our project made use of three distinct systems: the Steady Hand Eye Robot 2.0 (SHER), the Integrated Robotic Intraocular Snake 2.0 (I²RIS), and a Phantom Omni (3D Systems Inc., Rockhill, SC),. SHER and I²RIS form the 7 D.O.F hybrid system, while the Phantom Omni works as the primary, or leader, robot to control the position of the secondary, or follower, hybrid system.

Both SHER and I²RIS were developed in house, primarily through the AMIRO lab. The Phantom Omni is a commercially available device, but the communication protocol was developed privately, and readily available for the duration of the project.

The SHER is a 5 D.O.F robot with three translational stages and two rotational joints. The SHER also houses a mechanical Remote Center of Motion (RCM), which allows for a movement around a fixed point, almost as if pivoting around a point (as long as the robot's base translational stages are not moved). This is as a virtue of the parallelogram setup as can be seen in Figure 2. However, this point is fixed as long as the first three translational joints of the eye robot are locked.

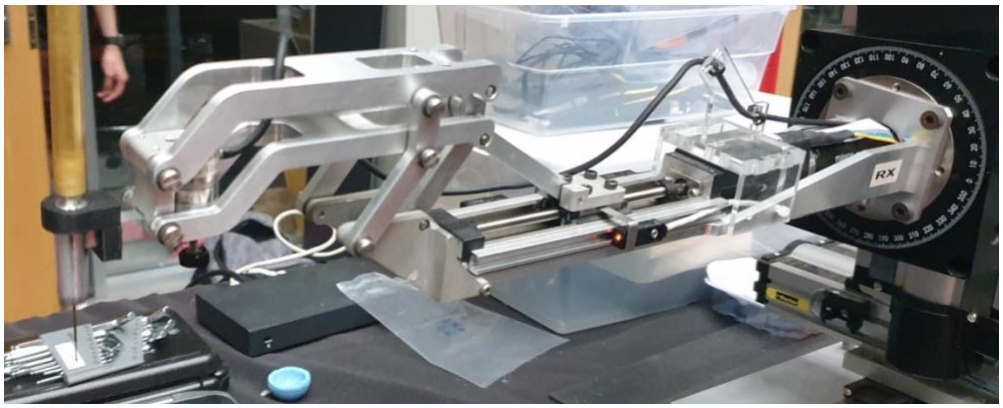

Figure 2. parallelogram design for mechanical RCM

The I²RIS is a 2 D.O.F continuum robot, with a diameter of 0.9 mm, measuring at ~3 mm fully extended. It utilizes 2 Maxon motors, and uses two thin cables looped around internal pulleys to bend the snake in X or Y direction. It is connected to an Arduino board, and two H bridges, in order to safely run the Maxon motors/ which serves as its connector. More details about the I²RIS can be found in the appendix.

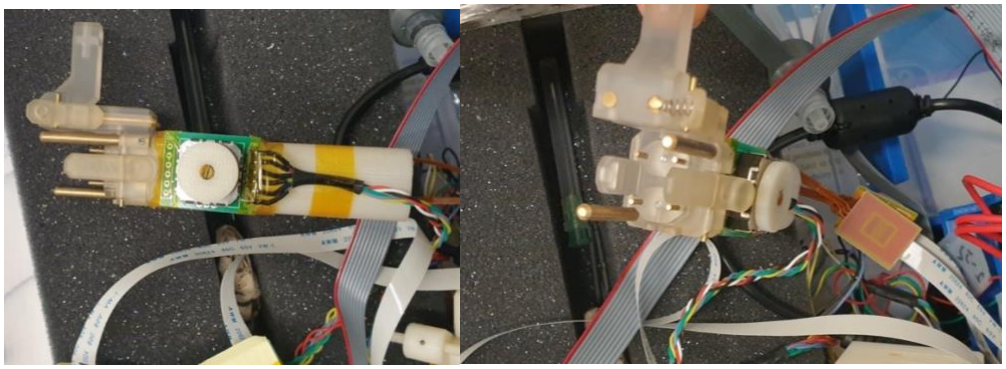

Figure 3. Side and top view of IRIS housing body

2. SHER and I²RIS integration

Both SHER and I²RIS were developed using the Computer Integrated Surgical Systems and Technology – Surgical Assistant Workstation (CISST-SAW) libraries, which is an open-source set of libraries developed specifically at Johns Hopkins to develop surgical robotics. SHER's code base also utilized ROS, and employed a CISST-SAW-ROS bridge, utilizing CRTK-messages to subscribe to and publish ROS commands, and converting them into CISST data types, allowing for seamless operation. CRTK-messages are essentially that are used to have CISST-SAW based system send and receive ROS messaged. The I²RIS robot's code base was not integrated with ROS, and as such, a sample implementation of CRTK-messages (developed by Henry Phalen) was utilized. Appropriate ROS publishers and subscribers were implemented to use ROS messages (output from the Phantom Omni) to control the I²RIS successfully.

In order to the run the I²RIS, we utilized a local network for use the same **roscore** as the main computer, in order to successfully send and receive ROS messages across two machines (I²RIS was operated via secondary laptop). CRTK messages have not been implemented on the I²RIS code build in the primary computer, and this trivial step will be addressed in the immediate time frame following the conclusion of the project. Both SHER and the Phantom Omni utilize a Firewire based communication-based protocol, so operating either of those two systems on newer machines is rendered difficult without a custom motherboard, hence all relevant code must be launched/used from the primary computer.

As is mentioned in the introduction of this report, the primary aim of this project is to develop, or establish relevant framework for, a hybrid system that not only is intuitive to control, but also gives the user (ideally surgeons) increased dexterity intraoperatively, compared to using the same system with a rigid tool.

A proof-of-concept study conducted by Kaiyu et al. specifically validates the usage of a flexible end-effector (i.e., I²RIS), identifying the increased reach of the user within the eye sphere

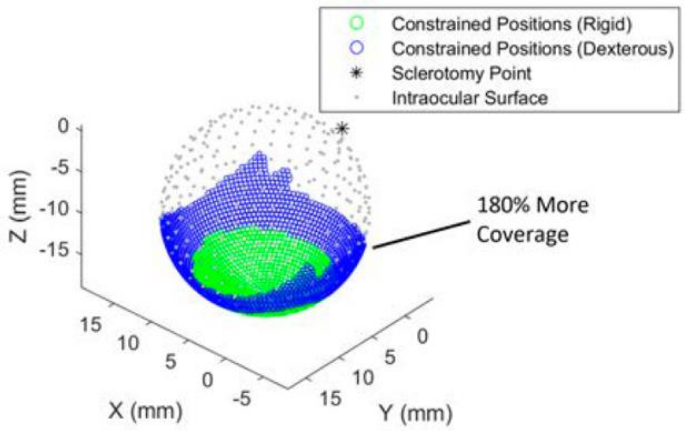

Figure 4. Comparison of rigid vs dexterous I2RIS reachability [5]

The dexterous snake tool, tested in simulation, granted surgeons/users greater dexterity intra-operatively, allowing them to reach positions that cannot be reached with a rigid tool, through paths that cannot be traversed with a rigid tool. The results, as can be seen in Figure 4, were very positive. The dexterous tool afforded the users 180% more workable space relative to the rigid tool. Moreover, areas that both tools could reach, could be reached through increased number of trajectories by the flexible tool, compared to the rigid tool. This is great validation for this project.

For the system to run together, and to solve the overall optimization problem (moving 7 D.O.F system with a 5 D.O.F follower robot), there is need for the overall system's forward kinematics. Proper physical integration is necessary in order to determine physically the right offsets, in order to both verify, and possibly correct, the analytical Jacobian used in operating the code. The total forwad kinematics chain is also necessary to obtain the appropriate manipulator Jacobian of the hybrid system, in order to set up the appropriate optimization problem, along with the appropriate constraints (discussed in following sections).

In addition, there were no previous attempts at teleoperating the real I²RIS (only in simulation). Our work directly produced a method to teleoperate the two Maxon motors in the I²RIS, by translating the movement of the Phantom Omni in X and Y direction into absolute encoder counts:

$$\Theta_{roll} = (\Delta Y + 40) * -0.0075$$
$$\Theta_{pitch} = \Delta Y * 0.002$$

The various numerical factors in the equations were determined experimentally. These values where then published via ROS, which the I²RIS subscribed to. Movement was capped by limiting the joint positions in the publishing node

3. Snake Robot Calibration

In order to control the snake robot, we need to know its kinematics model. Given that kinematics modeling of continuum robots is not as straightforward as conventional manipulators, we aim to develop an experimental forward kinematics and calibrate the movement of the new snake robot (I²RIS) to generate a mapping between the snake robot actuation space and its configuration space and finally to its task space. This will allow for a precise forward kinematic mapping, which is critical to establish any form of inverse kinematic model.

In the paper that forms the basis of this project (Kaiyu et.al), the authors, paper maps the bending of the snake as a function of the movement of each individual disk (which make up the snake) relative to the previous cylindrical surface. In short, they defined a kinematic chain between intermediate disks to represent the snake tip position and orientation. However, on the real robot, the intermediate disks angles can neither be measured nor controlled separately. Therefore, an experimental forward kinematic needs to be developed to map the actuation space (snake motor encoder angles) to the task space (snake tip spatial position and orientation.

Our experimental setup was devised as show in Figure 5. Using a Andonstar AD208 coin Microscope, we recorded the movement of the snake as we bent it at predetermined angles. We used the GUI file to move the snake motor with absolute encoder counts with steps of 100 counts to bend the snake robot for about 45 degrees (encoder counts ~8000) and saved the corresponding images. Then we used an application called "Color Thresholder" in MATLAB in which we defined to different masks to detect the red dot at the optical fiber tip, as well as the line of the optical fiber to report both tip position and also the bending angle. The results can be found under the results section.

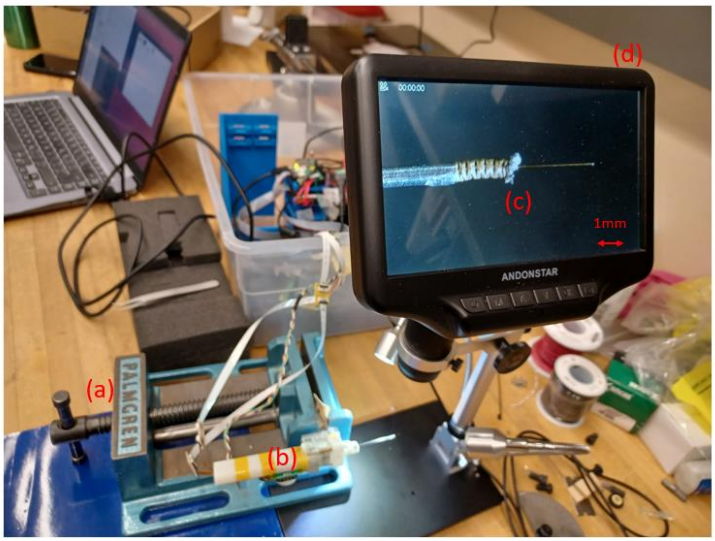

Figure 5. Calibration stage: a) fixture b) actuation unit of the snake robot including Maxon motors, derive pulley mechanism, cables etc. c) snake segment and an optical fiber passed through the snake as an indicator of measuring the bending angle as seen by the microscope d) microscope

4. Teleoperation Algorithm

1. Mathematical Approach

The control framework implemented in this project is a unilateral teleoperation control framework, since the algorithms only transfers the motion commands of the operator hand through the Phantom Omni robot to SHER and there is to force feedback reflected back from the eye robot-environment interaction to the operator's hand, that is, there is no haptic feedback and the operator controls the robot only with visual feedback. However, the controller framework is based on an optimization of the inverse kinematic which allows us to consider the constraints in the control framework. For example, joint position and joint velocity limits, insertion depth, RCM point, and virtual fixtures are all imposing some sort of constraints that could be dealt with in this approach.

Optimization-Based Inverse Kinematics:

Figure 6 shows the schematic view of the end-effector (handle and needle) of the eye robot relative to the eye. $\{S\}$ is the spatial frame attached to the base of SHER, $\{E\}$ is a frame rigidly attached to the eye phantom, B is the origin of the handle frame, and b is the origin of the frame attached to the tip, $\vec{P}_B$ and $\vec{P}_b$ are the absolute position vectors of the origin of those two points represented in the spatial frame, $\vec{P}_{trocar}$ is the position of the trocar point on the eyeball and $\vec{P}_{RCM}$ is the position of variable point along the needle which, the control action tries to hold it coincident with the $\vec{P}_{trocar}$. This means that the needle is allowed to insert to the eye trough the trocar point but is constrained from moving laterally to avoid causing injury to the eye.

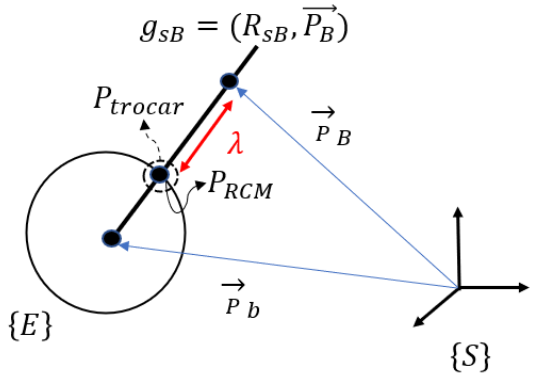

Figure 6. Schematic view of the last link (handle and needle) of the eye robot relative to the eye.

The teleoperation control framework with RCM and joint velocity constraints are formulated as follows,

$$\underset{\Delta q}{argmin} \|\Delta X - J_{sb}\Delta q\| + \Lambda\|\Delta q\| + \Gamma \left\|J_{RCM}(q,\lambda)\begin{bmatrix}\Delta q \\ \Delta\lambda\end{bmatrix}\right\|$$

$$s.t.$$

$$\varepsilon_{3*1min} < \left(\overrightarrow{P_{trocar}} - \overrightarrow{P_B}\right) \times R_B\overrightarrow{a_{tip}} \leq \varepsilon_{3*1max}$$

$$\underline{K}\Delta Q \leq \Delta Q_{limit}$$

$$\Delta\lambda_{min} \leq \Delta\lambda \leq \Delta\lambda_{max}$$

$$0 < \lambda_{min} \leq \lambda \leq \lambda_{max} < 1$$

$$J_{RCM} = \begin{bmatrix} J_{sB} + \lambda(J_{sb} - J_{sB}) \\ P_b - P_B \end{bmatrix}^T$$

In which $\Delta X \in \mathbb{R}^6$ is the difference between the current and previous end-effector position/orientation of the Omni robot, transferred to the body frame of the eye robot to generate intuitive motion direction. The first three element of $\Delta X(1:3)$ is the difference between current and previous position of the Omni end-effector and is calculated using direct subtraction ($P_{curr} - P_{prev}$) and the second three elements of $\Delta X(4:6) = \log(R_{prev}^T R_{curr})^\vee$ show the geodesic distance between the previous and current rotation matrices, J_{sb} is the Jacobian matrix of the point b, J_{RCM} is a Jacobian matrix to represent the RCM velocity. The first term in the constraints has to do with the geometrical constraint on the RCM point to constrain the needle from moving laterally relative to the trocar point, second constraint represents the SHER's angular joint velocity limits, the third and fourth one are related to the insertion velocity and insertion depth. The unit vector $\overrightarrow{a_{tip}}$ is along the needle from B towards b and represented in the Body frame B. Further details about the derivation of the objective and constraint relations are explained in the Appendix. Some key assumptions we make when proposing these constraints are that the target eye is perfectly registered, the exact position of the insertion point is known, and that we know the exact position of the center of the eye.

Due to unforeseen circumstances and unexpected issues with the code base, our team was unable to implement the full extent of our teleoperative algorithm. However, given that we were able to successfully derive the mathematical modeling of the optimization function, we do not foresee major challenges in implementing this.

2. *Implementation*

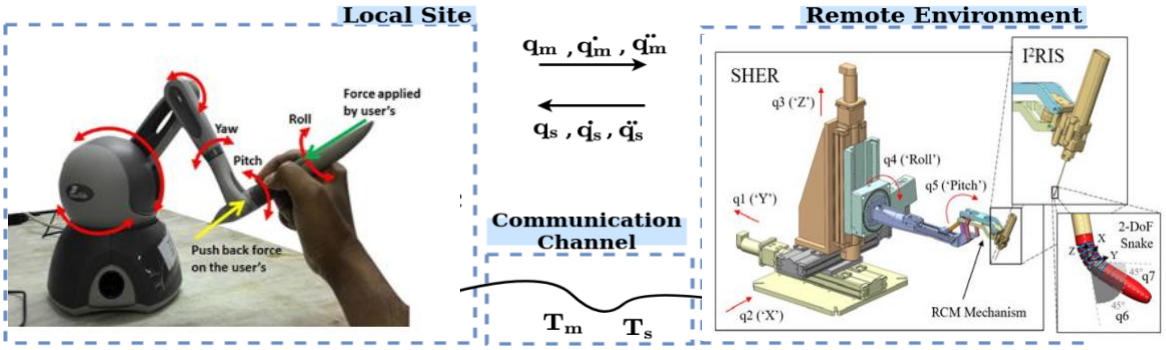

Figure 7. Proposed Teleoperation schematic

As mentioned, while we were unable to implement and test the teleoperative algorithm, we were able to identify the right approach to implementing this. We propose the use of NLopt, an open source non-liner C++ optimization library. With our objective function and key constraints, we will have to ensure that the variables passed into the optimizer match the standard data types that a C++ library would utilize (i.e., convert them from CISST data types into standard namespace variables). This would include, but not limited to, the Jacobain of the full hybrid system, the end position of the I²RIS's tip,

As an alternative to the full teleoperation algorithm implementation, we were able to implement the segmented teleoperative control. One key aspect of this method was the critical overlap with our suggested teleoperation algorithm, which was the implementation of the RCM point (i.e., the trocar). This RCM was different from the proposed RCM, instead using a compensatory motion of the SHER to translate the top of its end effector in the exact opposite direction as the tip, thus producing a resultant motion that would match the movement of the robot under a real RCM constraint (such as the one described in the previous subsection). This was combined with a simple teleoperation code, which read in the Phantom Omni's tip position, converted it into a CISST data type, and then transforming the recorded rotation by the following homogenous matrix to convert the Phantom Omni's tip position into SHER's workspace:

$$\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

The Phantom Omni's positions are then translated into the velocity vectors in the following manner:

$$V_x = -Pedal * 0.01 * 0.4 * (1 * Omni_x + SHER_{Joint_1})$$
$$V_y = Pedal * 0.01 * 0.4 * (1 * Omni_y - SHER_{Joint_2})$$
$$V_z = Pedal * 0.01 * 0.4 * (1 * Omni_z - SHER_{Joint_3})$$
$$\omega_{pitch} = Pedal * 0.01 * (1 * q_3 + SHER_{Joint_4})$$
$$\omega_{yaw} = -Pedal * 0.01 * (1 * q_4 + SHER_{Joint_5})$$
$$q_4 = -\text{asin}\,(\beta/\gamma)$$
$$q_3 = -\text{asin}\,(\alpha/\gamma)/\, q_4$$

Where Pedal is a value corresponding to how much the trigger pedal is being pressed, $Omni_*$ is the change in the Phantom Omni's movement a specified direction, $SHER_{Joint_*}$ is the value of that joint position of SHER, and α, β, γ are the Phantom Omni's rotation angles transformed into SHER's workspace.

These velocities are then converted in pseudo forces, which are then used to produce the pseudo RCM motion. The forces are used to recalculate velocities, which are then passed onto the Galil controllers of the SHER.

With this teleoperated RCM mode, the user can manually set the RCM point to be the point of insertion of the I²RIS into the eye and manipulate the SHER end effector to get the I²RIS tip in the ideal position. The user can then switch to operating the I²RIS to perform whatever task that needs to be performed, and then switch back to the SHER.

In order to validate the RCM teleoperation, we recorded the position of two points above and below the end effector, along the axis of the SHER tool. We utilized a predesigned phantom eye, with a narrow point of entry to simulate an entry point in an eye. This was done by outputting the cartesian positions of these two points in the SHER's base frame and plotting them in MATLAB.

Figure 8. Positions of the plotted points

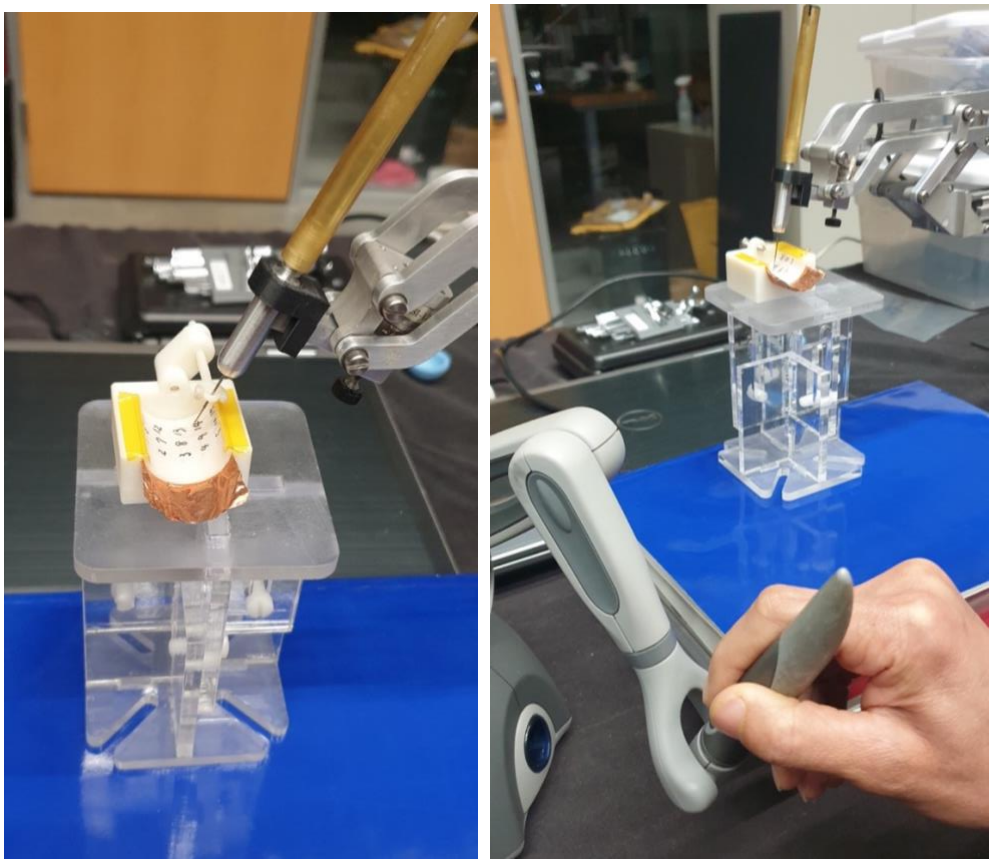

Figure 9. Testing setup to validate RCM teleoperative control

II. Result

Results of the snake calibration show that a GMM model with at least 3 different Gaussian distributions is capable of representing the input-output behavior of the snake with good precision (Figure 10). Training the GMM algorithm over 5 cycles of data points has shown good performance in capturing the nonlinear input-output feature of the robot by having a Coefficient of Determination (R2) of 0.99 and Root-Mean-Square Deviation (RMSD) of 2.74 deg and Normalized Root-Mean-Square Deviation (NRMSD) of 2.5%. Further details about the calibration results are provided in the Appendix.

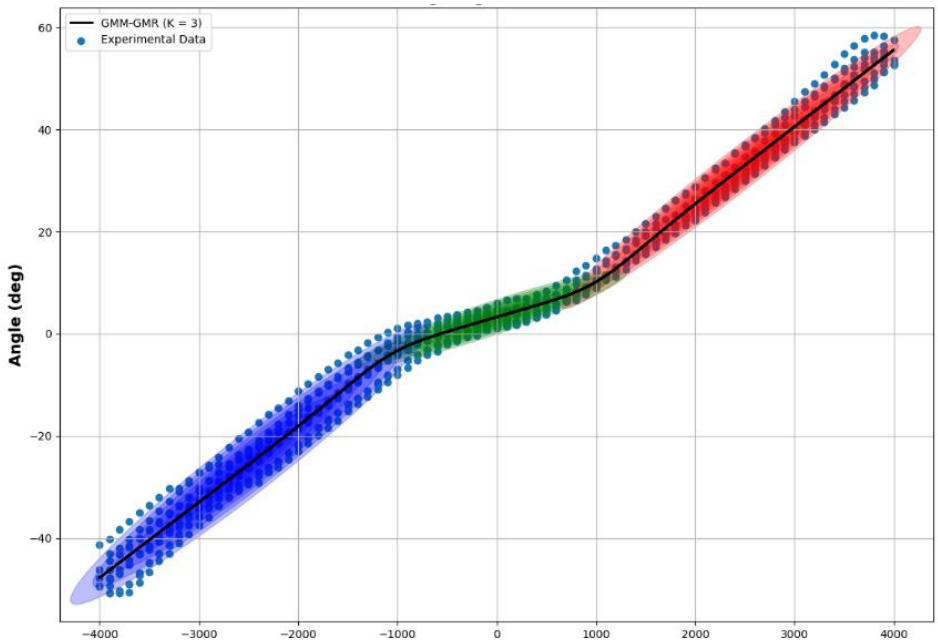

Figure 10. The probabilistic GMM-GMR algorithm for modeling the snake bending angle vs. motor encoder counts.

The results shown in Figure 11 are a graphical representation of the position of two different points: the a point that is situated a known distance above the SHER end effector (on the tool's axis), and the end effector tip frame (shown in figure). The plot is a clear depiction that the teleoperation control of the robot, with the RCM control implemented, **maintains** the RCM point, which can be inferred from the point where all the trajectory lines meet. This is a critical success, as this validates the segmented approach; the surgeon can proceed to insert the I²RIS tip into the eye, set the RCM point to the insertion point on the eye, and then switch to this teleoperation to position the I²RIS however it needed **safely**.

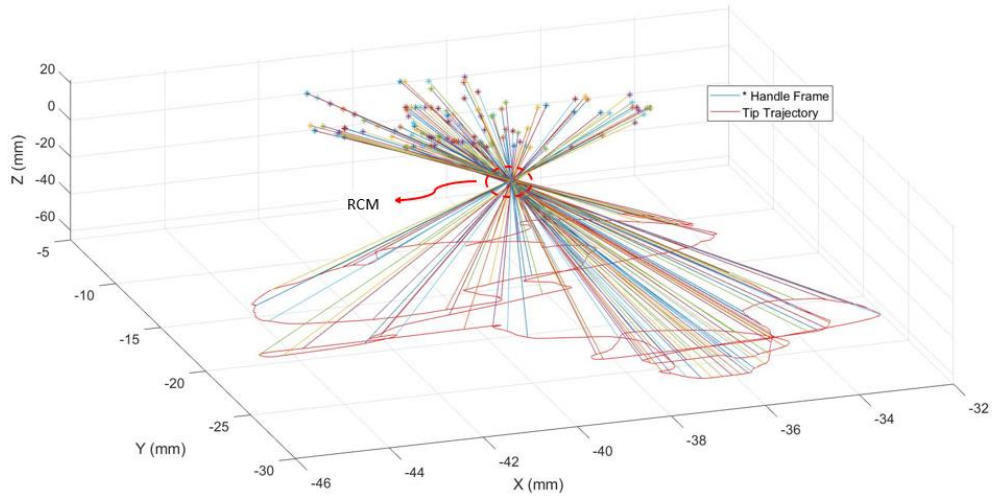

Figure 11. Tool tip trajectory in the teleoperation control mode with holding the RCM point

While there is no quantitative data to validate the teleoperation of I²RIS, we can qualitatively report that the I²RIS can be moved successfully, and relatively intuitively, with the messages published by the Phantom Omni. There is a noticeable latency, however, as well as delayed commands that sometimes get grouped with the next command, which results in overshoot. The team will address this by performing additional calibration, as well as making the communication protocol between the Phantom Omni and the I²RIS more efficient.

Management

Our team met weekly with either Professor Iordachita, or with Ali Ebrahimi (our primary advisor) for continuous feedback and troubleshooting. Our documentation (including but not limited to our reports, snake calibration results, and all relevant additions to the code) was stored in a One-Drive folder shared with Ali. In addition, updated code was stored on the lab computer (not pushed to Git-Hub, to preserve the default code base, and due to proprietary reasons), which was readily available to all mentors and team members.

Our algorithmic descriptions are included in this report, as is a detailed, step-by-step operational workflow (appendix A). Our system architecture can be found in Appendix B. These materials can also be found on the wiki page.

While most work was shared, our tasks were split into two broad categories: Mojtaba Esfandiari was responsible for the mathematical modelling and design of testing protocols, while Muhammad Hadi was responsible for working on the code and software development. Both teammates were involved equally in the documentation and any CIS II related presentations.

Team 22 was also assisted briefly by Trent Tang and Abdullah Al Armouti. Mr. Tang assisted the team by establishing the ROS publishers and subscribers used to communicate between the Phantom Omni and the I²RIS. Mr. Al Armouti is currently involved in manipulating the existing variables in the code base to use in an implementation of the NLOPT optimizer.

The deliverables proposed at the checkpoint presentation are listed below

- Minimum
    - Successful integration of the Snake Robot software with the Eye Robot software
    - A model of accurate_mapping of snake robot movement/calibration mode of I2RIS

- Expected
    - Code package with successful control algorithms for the tele-operated system with constraint, with appropriate documentation for ease of future development
    - A comprehensive report highlighting the results of experimental testing of the implemented teleoperation code
- Maximum
    - Design and execution of experiment to evaluate complete teleoperation control vs cooperative control
    - Academic paper on teleoperation control of hybrid system (Eye + Snake Robot).

We were able to meet our minimum deliverables for this project, as can be inferred by the details provided in this report. We were also able to produce a report on the calibration of the I²RIS, which has been attached as an appendix. We were not, however, able to meet a full version of our expected deliverables. We were not able to implement the actual teleoperation control algorithm in C++, and so implemented a segmented teleoperation control of the hybrid system. We were able to validate the accurate performance of the SHER's teleoperation, and qualitatively validate the teleoperation of the I²RIS.

We did not expect to meet our maximum deliverables and will be focusing on them in the next phase of this project (described below)

Future Work

In the short-term following the conclusion of the semester, our team hopes to perform a finer calibration of the I²RIS and produce a final forward kinematic chain that utilizes the results of this new calibration. We also aim to implement relative teleoperation control of the overall system, which caters directly to increasing the intuitive control of the robot. We also hope to improve upon the switching between the two control modes for the segmented teleoperation, as the current paradigm might require a certain level of dexterity to be able to generate smooth trajectory.

In retrospect, to be able to develop a more general assistive control framework that is capable of assisting the surgeons to follow a desired trajectory (e.g. a line or a curve on the sphere surface) by generating virtual fixtures, we will aim control both eye robot and snake robot as one hybrid robot with a new Jacobian matrix that represents the kinematics of the two robots combined (i.e., implement the complete teleoperation of the 7 D.O.F robot, allowing for unified teleoperation control). This will be accomplished in our medium to long term, goals. By doing so, we will be able to come up with a new constrained control framework that compensates surgeons lack of dexterity or sudden hand movement while following a desired trajectory, also constraining the snake tip angle with the trajectory to be within a desired range. This ability of the control algorithm reduces the surgeons effort to maintain the robot close to the desired trajectory via trial and error and, as a result, improves performance of the operation and reduces its risk.

To do this, the forward kinematics of the hybrid robot is developed according to the following equations:

$$V_{st}^{s} = V_{sB}^{s} + Ad_{g_{sB}} V_{Bt}^{s}$$

$$V_{Bt}^{s} = V_{Bb}^{s} + Ad_{g_{Bb}} V_{bt}^{s}$$

$$V_{bt}^{s} = J_{I2RIS} \begin{bmatrix} \dot{q}_6 \\ \dot{q}_7 \end{bmatrix}$$

$$V_{sB}^{s} = J_{SHER} \dot{q}_{5\times 1}$$

$$V_{st}^{s} = J_{SHER}\dot{q}_{5\times1} + Ad_{g_{sB}}\left(V_{Bb}^{s} + Ad_{g_{Bb}}J_{I2RIS}\begin{bmatrix}\dot{q}_6\\ \dot{q}_7\end{bmatrix}\right) = \begin{bmatrix}J_{SHER} & Ad_{g_{sB}}Ad_{g_{Bb}}J_{I2RIS}\end{bmatrix}\begin{bmatrix}\dot{q}_{SHER(5\times1)}\\ \dot{q}_{I2RIS(2\times1)}\end{bmatrix}$$

In which V_{st}^{s} represents the spatial generalized velocity of the snake tip frame {t} relative to the robot base frame {S}, J_{I2RIS} is the Jacobian of the snake robot tip frame relative to the snake base frame {b} (attached to the intersecting point of the snake base and the needle tip), $\dot{q}_6$ and $\dot{q}_7$ are the angular velocities of the snake motors, and Ad_g is the adjoint transformation of the $g = (R, P) \in SE(3)$ such that,

$$Ad_g = \begin{bmatrix}R & \hat{P}R\\ 0 & R\end{bmatrix}$$

Finally, the forward kinematics of the hybrid system can be formulated as follows,

$$g_{hybrid} = g_{SHER} \times g_{base} \times g_{snake}$$

In which g_{SHER} is the froward kinematics of the SHER robot from its base frame {S} to its Body frame {B}, g_{base} is the coordinate transformation due to the offset between the Body frame {B} of the SHER and base frame of the snake {b}, and g_{snake} is the froward kinematics between the snake base frame {b} to the snake tip frame.

Finally, the Jacobian of the hybrid system can also be developed as,

$$V_{st}^{s} = J_{Hybrid}\dot{q}$$

$$J_{hybrid} = \begin{bmatrix}J_{SHER} & Ad_{g_{sB}}Ad_{g_{Bb}}J_{I2RIS}\end{bmatrix}$$

$$\dot{q} = \begin{bmatrix}\dot{q}_{SHER(5\times1)}\\ \dot{q}_{I2RIS(2\times1)}\end{bmatrix}$$

Once this is accomplished, we will then perform a comparative study between different modes of operation, such as cooperative control, cooperative/teleoperative, and segmented teleoperation, and full teleoperation. This will then be used to produce a publishable piece of literature.

Roadblocks and Key Takeaways

The team did not anticipate the sheer complexity of the existing code base for SHER, which is what contributed to the delay in accomplishing some of the deliverables. Moreover, our inexperience with working with ROS, and especially CISST-SAW, was a great roadblock for us. While we were able to get a handle on the code in time to accomplish a modified version of our proposed teleoperation algorithm, we struggled quite a bit to implement it. Moreover, our unplanned dependency on non-CIS II affiliates proved to be a liability, when we should have assumed full responsibility for every single deliverable and sub-deliverable. It would also have served us well to produce a well thought our system schematic, as we often found ourselves discussing the same elements of our teleoperation approach repeatedly.

We learned a lot about code structure, and especially about ROS and CISST-SAW, both of which are very important for our research spheres here at JHU. We also developed keen acumen in control algorithms, especially during our efforts to produce well rounded and comprehensive constraints for our objective functions, especially data based modelling. We also appreciate the importance of maintaining key documentation throughout a project, as well as how to collaboratively work on large code packages.

References

[1] "The peripheral aspects of vitreoretinal surgery," *Centre for Sight*, 07-Mar-2020. [Online]. Available: https://www.centreforsight.net/blog/the-peripheral-aspects-of-vitreoretinal-surgery/. [Accessed: 01-Mar-2022].

[2] "Eye anatomy," *Glaucoma Research Foundation*. [Online]. Available: https://www.glaucoma.org/glaucoma/anatomy-of-the-eye.php. [Accessed: 01-Mar-2022].

[3] Group 22 (2022). *Team_22_project_proposal* [PDF], CIS II.

[4] Ebrahimi, Ali. (2018). *CIS II Project – Spring 2022* [PowerPoint presentation], CIS II.

[5] Shi, Kaiyu; Zhou, Yishun; Ebrahimi, Ali; Li, Gang; Iordachita, Iulian (2022), Optimization-based Concurrent Control of a High Dexterity Robot for Vitreoretinal Surgery. Manuscript submitted for publication

[6] Jinno, Makoto, and Iulian Iordachita. "Improved Integrated Robotic Intraocular Snake." *2020 International Symposium on Medical Robotics (ISMR)*. IEEE, 2020.

[7] Song, Jingzhou, et al. "Intraocular snake integrated with the steady-hand eye robot for assisted retinal microsurgery." *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[8] Sylvain, C. "Robot programming by demonstration: A probabilistic approach." (2009).

[9] Moon, Todd K. "The expectation-maximization algorithm." *IEEE Signal processing magazine* 13.6 (1996): 47-60.

[10] Aghakhani, Nastaran, et al. "Task control with remote center of motion constraint for minimally invasive robotic surgery." *2013 IEEE international conference on robotics and automation*. IEEE, 2013.

Appendix

Additional documentation, MATLAB files, and details can be found on the project's wiki page

Wiki Page

Teleoperation Setup Workflow (instructions)

In order to operate the current segmented teleoperative control of the 7 D.O.F system, the following steps should be followed

1. Ensure I²RIS is attached properly to the SHER's end effector, with the appropriate adapter
    a. Ensure I²RIS is plugged into the laptop that will be running the *snake_omni.cpp* and the *StartMaxonWithUI.launch* will be running
    b. Ensure that the power supply to the I²RIS is plugged in and the power is on
        i. The LED on the Arduino will be on
2. Ensure the Phantom Omni is connected to the primary PC via its Firewire cable
3. Follow the following steps to start the Phantom Omni's control
    a. Open a terminal window
    b. On the primary computer**, cd** into "omni_ws" from the home directory
        i. Run: **source devel/setup.bash**
    c. Repeat step 2 on another terminal window
    d. Open another terminal window, and run **roscore**
    e. In the other two terminals, execute the following commands
        i. **rosrun geomagic_launch geomagic.launch**
        ii. **rostopic echo /Geomagic/pose**
            1. This is to ensure that the Phantom Omni is indeed publishing its pose
    f. On a third terminal, run the following commands
        i. **cd catkin_ws**
        ii. **source devel/setup.bash**
        iii. **rosrun beginners_tutorials GeomagicSubscriber**
            1. this is to ensure that the Phantom Omni publishes messages that can be read by both the SHER and I²RIS
            2. in order to verify this, in another terminal, run **rostopic echo /eye_robot/Set…** (any of the three topics that start with that)
4. Follow the following steps to start operating the I²RIS (on the secondary computer)
    a. Disconnect from the WIFI, and use an ethernet cable to connect to the local network router located on the desk that houses the SHER
    b. Edit your ".bashrc" to include the following commands at the bottom of the script (DO NOT FORGET TO CHANGE THIS BACK WHEN DONE USING THE SYSTEM)

```
export ROS_MASTER_URI=http://(primary computer's IP):11311
export ROS_HOSTNAME= (secondary computer's IP)
```

        i. IP addresses can be found by running **ifconfig** in the terminal for any given machine
    c. Open a terminal window
    d. **cd catkin_ws**
    e. **source devel/setup.bash**
    f. **rosrun snake_omni snake_omni_node**

g. **roslaunch snake StartMaxonPositionMode.launch**
h. Press the grey button, and the snake should move with the movement of the Phantom Omni in the X-direction (disengage the snake by ending the execution in step 4.)

5. Follow the following steps to use the SHER on the primary computer
    a. Open a terminal window
    b. **rosrun eye_robot_example eye_robot_example**
    c. Release the e-Stop button
    d. Click on the red button on the top right of the GUI labelled "Robot Off." Upon clicking, this button should turn green.
    e. On the GUI that opens, click on "3DM control"
    f. Pick up the Phantom Omni's pen, press the grey pedal, and move the SHER (the system should be in the RCM mode by default)
        i. In order to change this to a normal, free space teleoperation control, follow the following steps:
            1. Navigate to the code package (~/home/catkin_ws/src/eye_obot/EyeRobot2/code"
            2. Open robottask.cpp
            3. On line 1848, change teleoperation mode from "5" to "3"
            4. Once saved, run **catkin_build eye_robot_example** in a terminal.

Non-sequential workflow instructions:

- Pressing the grey button on the Phantom Omni pen will engage the I²RIS. In order to pause operation of the SHER, release the grey pedal
    - Relative movement has **not** been implemented, so the Phantom Omni pen should be returned to the same position as the position of the SHER's end effector
- If the SHER behaves erratically, immediately release the pedal and push the e-Stop
- If you wish to collect data, click on the logger button on the top right, enter the name of your file, and click reocrd.
    - To end data collection, press the white button on the Phantom Omni
        - Take care to do this before pressing "quit" on the GUI, to avoid file saving errors

Snake robot calibration details

In order to develop a forward kinematic model for the snake robot (I2RIS), four general operating spaces could be defined (Figure 12). There is also a mapping between the encoder values and the motor shaft angle due to the existence of gearbox which is not shown in Figure 12 for simplicity. In our application, $q \in \mathbb{R}^2$, since there are two motors (Maxon DCX08M EB KL 4.2V) with gearhead (GPX08 A 64:1) that are responsible for bending the snake robot in two perpendicular planes, $X \in \mathbb{R}^3$ is the snake tip position described in the snake base coordinate and $\gamma \in \mathbb{R}^2$ represents the two bending angles pitch and yaw (there is no roll).

An analytical forward kinematic mapping between the encoder values to the motor shaft angles and to the cable displacements is developed in previous work by Jinno-Iordachita [6].

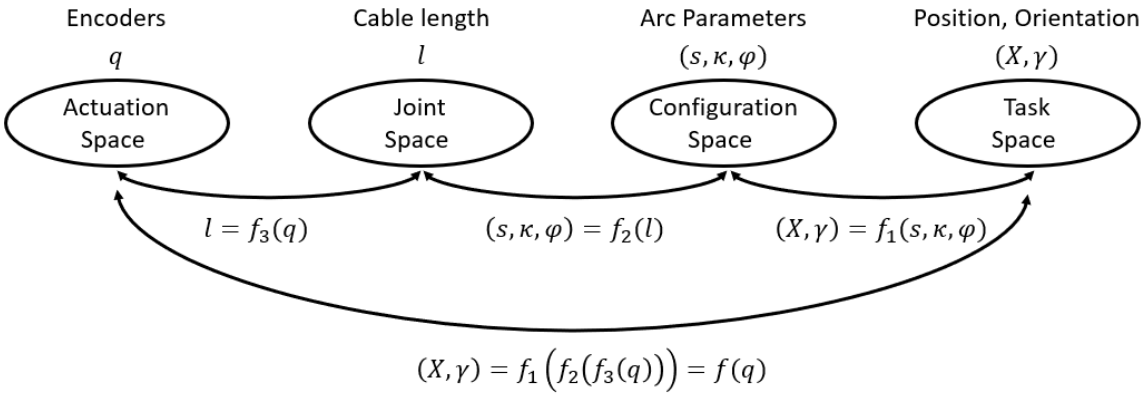

Figure 12 Schematic view of the operating spaces of the snake robot (I2RIS).

Given that this approach models the robot kinematic based on the cable displacement, which is not measurable in practice and we can only measure the encoder values and the position/orientation using camera, we will utilize an experimental kinematic modeling approach employed by Song et. al. [7]. Figure 13 shows the calibration stage that is used for measuring the snake tip position and orientation (bending angle) as output of the vision-based algorithm for different encoders values as the input for five fully cycles in the snake robot's range of motion (~ -40 deg to + 40 deg).

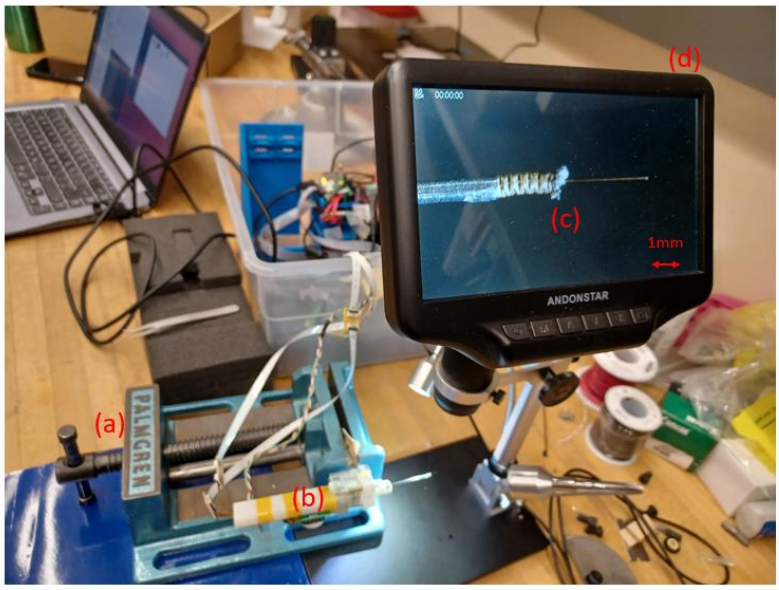

Figure 13. Calibration stage: a) fixture b) actuation unit of the snake robot including Maxon motors, derive pulley mechanism, cables etc. c) snake segment and an optical fiber passed through the snake as an indicator of measuring the bending angle as seen by the microscope d) microscope

In the first test, calibration has been done on the brass snake which shows that it has several mechanical problems such as backlash, hysteresis, and wire tension problem around the neutral axis (when the snake is oriented strait) (Figure 14 and Figure 15).

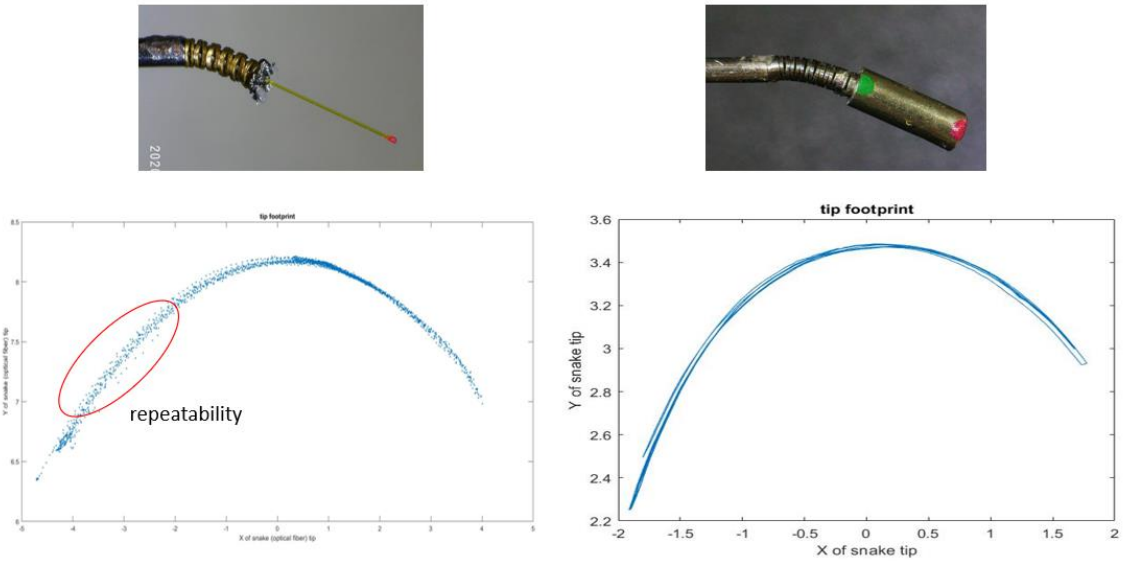

Figure 14. Left: Calibration of the brass snake $z(mm) - x(mm)$; Right: calibration of the stainless-steel snake $z(mm) - x(mm)$. We can see that the brass snake has low repeatability specially for the negative values of x.

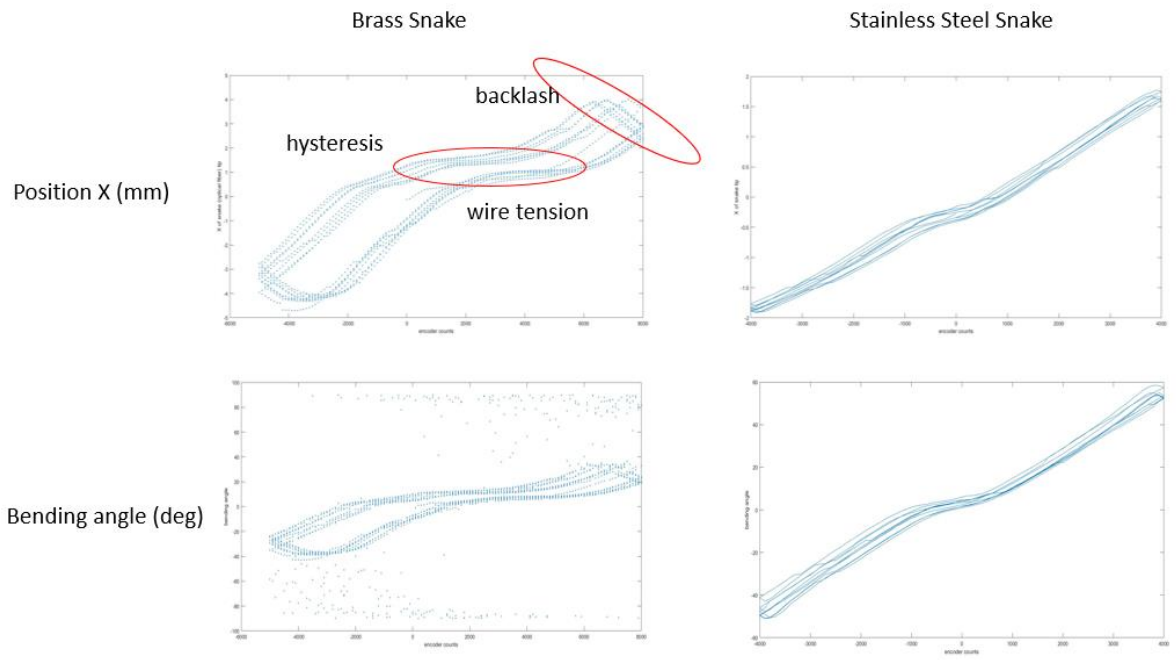

Figure 15. Left column: Brass snake calibration result, top: Snake robot tip position $x\ (mm)$ vs encoder counts, bottom: snake robot bending angle $\gamma\ (deg)$ vs encoder counts; Right column: stainless steel snake calibration result, top: Snake robot tip position $x\ (mm)$ vs encoder counts, bottom: snake robot bending angle $\gamma\ (deg)$ vs encoder counts.

We employed a probabilistic model, Gaussian Mixture Model (GMM) – Gaussian Mixture Regression (GMR), for nonlinear mapping between the input (encoder counts) and the outputs (snake tip position and bending angle)[8].
The total dataset collected during eight reciprocating movement over the snake range of motion is defined as follows.

$$\mathcal{D} = \begin{bmatrix} q_{1,1} & q_{1,2} & \cdots & q_{c,n} & \cdots & q_{C,N} \\ \gamma_{1,1} & \gamma_{1,2} & \cdots & \gamma_{c,n} & \cdots & \gamma_{C,N} \end{bmatrix} \in \mathbb{R}^{D\times(C\times N)} = \mathbb{R}^{D\times\mathbb{N}}$$

In which $\gamma_{c,n}$ is the snake bending angle for n^{th} data point of the c^{th} test cycle, $q_{c,n}$ is the motor encoder counts for the n^{th} data point of the c^{th} cycle, C is the total number of calibration cycles, N is the number of data points in each cycle, $\mathbb{N}$ is the total number of data points for all (training) cycles, D is the dimension of data points and j^{th} datapoint ($j = 1,2,\dots,\mathbb{N}$) is defined as follows,

$$\xi_j = \begin{bmatrix} q_j \\ \gamma_j \end{bmatrix} = \begin{bmatrix} \xi_j^I \\ \xi_j^O \end{bmatrix} \in \mathbb{R}^{\boldsymbol{D\times 1}}$$

In which super scripts I and O denote input and output, respectively. Using the Gaussian Mixture Model (GMM), the probability for a datapoint ξ_j belonging to the GMM including K Gaussian distributions could be defined as follows:

$$\mathcal{P}(\xi_j) = \sum_{k=1}^{K} \mathcal{P}(k)\mathcal{P}(\xi_j|k)$$

$$\mathcal{P}(k) = \pi_k$$

$$\mathcal{P}(\xi_j|k) = \mathcal{N}(\xi_j|\mu_k, \Sigma_k) = \frac{1}{\sqrt{(2\pi)^D|\Sigma_k|}} e^{-\frac{1}{2}\left((\xi_j-\mu_k)^T\Sigma_k^{-1}(\xi_j-\mu_k)\right)}$$

In which $\mathcal{P}(k) = \pi_k$ is a prior probability, $\mathcal{P}(\xi_j|k)$ is the conditional probability density, μ_k and Σ_k are the mean value and covariance matrices of the k^{th} Gaussian distribution $\mathcal{N}(\xi_j|\mu_k, \Sigma_k)$.
Then We used Expectation Maximization (EM) algorithm over the dataset $\boldsymbol{\mathcal{D}}$ to iteratively optimize the parameters of the GMM ($\Theta_k = \{\pi_k \in \mathbb{R},\ \mu_k \in \mathbb{R}^2,\ \Sigma_k \in \mathbb{R}^{2\times2}\}_{k=1}^K$) subject to the following constraint:

$$\sum_{k=1}^K \pi_k = 1 \quad , \quad \pi_k \in [0,1]$$

After training the GMM model with EM algorithm [9] and optimizing the parameters Θ_k, it is now possible to estimate the output (snake bending angle or tip position) for any given input (encoder counts) with Gaussian Mixture Regression (GMR). The conditional probability of output ξ^O for a given input ξ^I could then be estimated in the GMR phase as

$$\mathcal{P}(\xi^O|\xi^I) \sim \sum_{k=1}^K h_k \mathcal{N}(\hat{\xi}_k, \hat{\Sigma}_k)$$

In which $\hat{\xi}_k$ and $\hat{\Sigma}_k$ are the expected mean values and covariance matrices of the k^{th} Gaussian distribution and are calculated as follows,

$$\hat{\xi}_k = \mu_k^O + \Sigma_k^{OI}(\Sigma_k^{OI})^{-1}(\xi^I - \mu_k^I)$$

$$\hat{\Sigma}_k = \Sigma_k^O - \Sigma_k^{OI}(\Sigma_k^I)^{-1}\Sigma_k^{IO}$$

and $h_k = \mathcal{P}(k|\xi^I)$ specifies the probability of the k^{th} Gaussian distribution being responsible for ξ^I and is calculated as

$$h_k = \frac{\mathcal{P}(k)\mathcal{P}(\xi^I|k)}{\sum_{k=1}^K \mathcal{P}(i)\mathcal{P}(\xi^I|i)} = \frac{\pi_k \mathcal{N}(\xi^I; \mu_k^I, \Sigma_k^I)}{\sum_{k=1}^K \pi_i \mathcal{N}(\xi^I; \mu_i^I, \Sigma_i^I)}$$

Finally, it is possible to approximate the conditional expectation of the output ξ^O for a given input ξ^I using a single Gaussian distribution function $\mathcal{N}(\hat{\xi}, \hat{\Sigma})$ such that

$$\hat{\xi} = \sum_{k=1}^K h_k \hat{\xi}_k$$

$$\hat{\Sigma} = \sum_{k=1}^K h_k^2 \hat{\Sigma}_k$$

Control, objective function, and constraints:

Joints Velocity Constraint:

The constraints on the desired joint velocity to be sent to the Galil controller ($K\Delta q$) could be defined as follows:

$$\begin{bmatrix}\Delta q_{1min}\\ \Delta q_{2min}\\ \Delta q_{3min}\\ \Delta q_{4min}\\ \Delta q_{5min}\end{bmatrix} \leq \begin{bmatrix}k_1 & 0 & 0 & 0 & 0\\ 0 & k_2 & 0 & 0 & 0\\ 0 & 0 & k_3 & 0 & 0\\ 0 & 0 & 0 & k_4 & 0\\ 0 & 0 & 0 & 0 & k_5\end{bmatrix}\begin{bmatrix}\Delta q_1\\ \Delta q_2\\ \Delta q_3\\ \Delta q_4\\ \Delta q_5\end{bmatrix} \leq \begin{bmatrix}\Delta q_{1max}\\ \Delta q_{2max}\\ \Delta q_{3max}\\ \Delta q_{4max}\\ \Delta q_{5max}\end{bmatrix}$$

In which $k_1, \dots, k_5 > 0$ are user inputs for tuning all different five joint velocities of the eye robot, $\Delta q_{imin},\ \Delta q_{imax}\ (i = 1, \dots, 5)$ are the maximum and minimum angular velocity limits of the i^{th} joint. The above relation can be restated in the following form,

$$K = \begin{bmatrix}k_1 & 0 & 0 & 0 & 0\\ 0 & k_2 & 0 & 0 & 0\\ 0 & 0 & k_3 & 0 & 0\\ 0 & 0 & 0 & k_4 & 0\\ 0 & 0 & 0 & 0 & k_5\end{bmatrix}, \Delta q = \begin{bmatrix}\Delta q_1\\ \Delta q_2\\ \Delta q_3\\ \Delta q_4\\ \Delta q_5\end{bmatrix},\ \Delta q_{max} = \begin{bmatrix}\Delta q_{1max}\\ \Delta q_{2max}\\ \Delta q_{3max}\\ \Delta q_{4max}\\ \Delta q_{5max}\end{bmatrix}, \Delta q_{min} = \begin{bmatrix}\Delta q_{1min}\\ \Delta q_{2min}\\ \Delta q_{3min}\\ \Delta q_{4min}\\ \Delta q_{5min}\end{bmatrix}$$

$$\Rightarrow \quad \Delta q_{min} \leq K\Delta q \leq \Delta q_{max} \quad \Rightarrow \quad \begin{cases} K\Delta q \leq \Delta q_{max} \\ -K\Delta q \leq -\Delta q_{min}\end{cases}$$

$$\Rightarrow \quad \begin{bmatrix}K & 0_{5\times5}\\ 0_{5\times5} & -K\end{bmatrix}\begin{bmatrix}\Delta q\\ \Delta q\end{bmatrix} \leq \begin{bmatrix}\Delta q_{max}\\ \Delta q_{min}\end{bmatrix} \quad \Rightarrow \quad \underline{K}\Delta Q \leq \Delta Q_{limit}$$

RCM Velocity:

According to the Figure 6, $\vec{\mathrm{P}}_{\mathrm{trocar}}$ in the robot base frame $\{\mathrm{S}\}$ is known from registration and $\vec{\mathrm{P}}_{\mathrm{B}}$ and $\vec{\mathrm{P}}_{\mathrm{b}}$ are known from forward kinematics of the eye robot. Knowing the absolute position information of these points, we can develop the following equations to represent the RCM velocity and then try to penalize it in the objective function.

Note: This J_{RCM} is different than the one defined in the eye-robot code [10].

$$P_{RCM}\big(q(t)\big) = P_B\big(q(t)\big) + \lambda(t)\ \Big(P_b\big(q(t)\big) - P_B\big(q(t)\big)\Big)$$

$$\dot{P}_{RCM} = \dot{P}_B + \dot{\lambda}(P_b - P_B) + \lambda\big(\dot{P}_b - \dot{P}_B\big)$$

$$\dot{P}_{RCM} = J_{sB}\dot{q} + \dot{\lambda}(P_b - P_B) + \lambda(J_{sb}\dot{q} - J_{sB}\dot{q})$$

$$\dot{P}_{RCM} = \begin{bmatrix}J_{sB} + \lambda(J_{sb} - J_{sB})\\ P_b - P_B\end{bmatrix}^T \begin{bmatrix}\dot{q}\\ \dot{\lambda}\end{bmatrix}$$

$$\text{Assumption: } P_{RCM} = P_{trocar} = cte \quad \Rightarrow$$

$$\dot{P}_{RCM} = J_{RCM} \begin{bmatrix} \dot{q} \\ \dot{\lambda} \end{bmatrix} = 0 \qquad , \; J_{RCM} = \begin{bmatrix} J_{sB} + \lambda(J_{sb} - J_{sB}) \\ P_b - P_B \end{bmatrix}^T$$

Since the above assumption is problematic in practice, we avoided using this equation in the constraints, rather we tried to minimize it inside the objective function and used another geometrical method as a constraint for the RCM point which tries to minimize the cross product between the needle direction $R_B\overrightarrow{a_{tip}}$ and the vector connecting the two points $\left(\overrightarrow{P_{trocar}} - \overrightarrow{P_B}\right)$. This geometrical constraint allows the needle to freely move in and out to the eye trough the trocar and also to bend in pitch and yaw directions about the trocar point but restricts the RCM point on the needle from having lateral deviation from the trocar point (Figure 11).