Autonomously Navigating a Surgical Tool Inside the Eye by Learning from Demonstration

Ji Woong Kim[1], Changyan He[1], Muller Urias[3], Peter Gehlbach[3],
Gregory D. Hager[2], Iulian Iordachita[1], Marin Kobilarov[1]

***Abstract*— A fundamental challenge in retinal surgery is safely navigating a surgical tool to a desired goal position on the retinal surface while avoiding damage to surrounding tissues, a procedure that typically requires tens-of-microns accuracy. In practice, the surgeon relies on depth-estimation skills to localize the tool-tip with respect to the retina and perform the tool-navigation task, which can be prone to human error. To alleviate such uncertainty, prior work has introduced ways to assist the surgeon by estimating the tool-tip distance to the retina and providing haptic or auditory feedback. However, automating the tool-navigation task itself remains unsolved and largely unexplored. Such a capability, if reliably automated, could serve as a building block to streamline complex procedures and reduce the chance for tissue damage. Towards this end, we propose to automate the tool-navigation task by mimicking the perception-action feedback loop of an expert surgeon. Specifically, a deep network is trained to imitate expert trajectories toward various locations on the retina based on recorded visual servoing to a given goal specified by the user. The proposed autonomous navigation system is evaluated in simulation and in real-life experiments using a silicone eye phantom. We show that the network can reliably navigate a surgical tool to various desired locations within 137 μm accuracy in phantom experiments and 94 μm in simulation, and generalizes well to unseen situations such as in the presence of auxiliary surgical tools, variable eye backgrounds, and brightness conditions.**

I. Introduction

Retinal surgery is among the most challenging microsurgical endeavors due to its micron scale precision requirements, constrained work-space, and the delicate non-regenerative tissue of the retina. During the surgery, one of the most challenging tasks is the spatial estimation of the surgical tool location with respect to the retina in order to precisely move its tool-tip to a desired location on the retina. For example, when performing retinal-peeling or vein cannulation, the surgeon must rely on intuitive depth-estimation skills to navigate toward a targeted location on the retina, while ensuring that the tool-tip contacts the retina precisely at the desired location. Such maneuvers introduce high risk because the surgical tools are sharp, and the slightest misjudgement can damage the surrounding tissues, which could lead to serious complications.

[1]J. Kim, C. He, I. Iordachita, and M. Kobilarov are with Laboratory for Computing + Sensing (LCSR) dept. and [2]GD Hager is with the Computer Science dept. at the Johns Hopkins University, Baltimore, MD 21218 USA {jkim447, changyanhe, iordachita, marin}@jhu.edu, hager@cs.jhu.edu

[3]M. Urias, P. Gehlbach are with Wilmer Eye Institute at the Johns Hopkins Hospital, Baltimore, MD 21287 USA {murias1, pgelbach@jhmi.edu}.

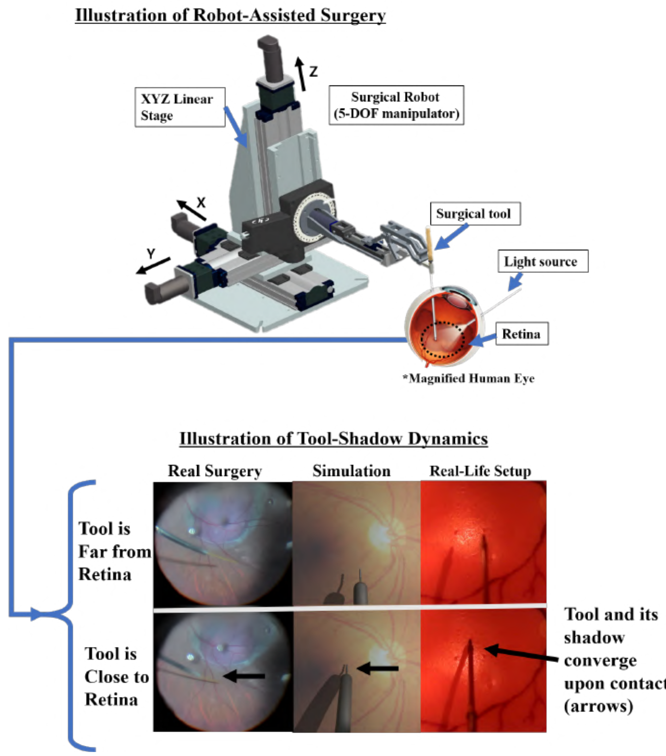

Fig. 1: (Top) Robot-assisted eye surgery set-up; a light source projects a shadow on the retina which can be used as cues to estimate proximity between the tool-tip and the retina. (Bottom) Demonstration of tool-shadow dynamics; as the surgical tool approaches to the retina, the tool and its shadow converge (compare top row to bottom row), which can be used as cues to train a network how to navigate inside the eye.

In order to assist retinal surgeons, several works introduced ways to close the uncertainty gap of estimating the depth between the tool-tip and the retina. For example, proximity between tool-tip and retina can be computed using stereo vision [1], by estimating the converging characteristics of the tool-tip and shadow-tip [2], and using optical coherence tomography systems (OCT) [3]. Often, such system require precise calibration and hand-tuned adjustment to specific environments. Furthermore, prior works strive to assist the surgeon by providing rough state estimates, rather than automating the tool-navigation task.

In this paper, we propose to automate surgical-tool navigation task by learning the closed-loop visual servoing process employed by surgeons, i.e. mapping from visual input (video) to euclidean position control commands to actuate the robot. Specifically, we train a deep network to

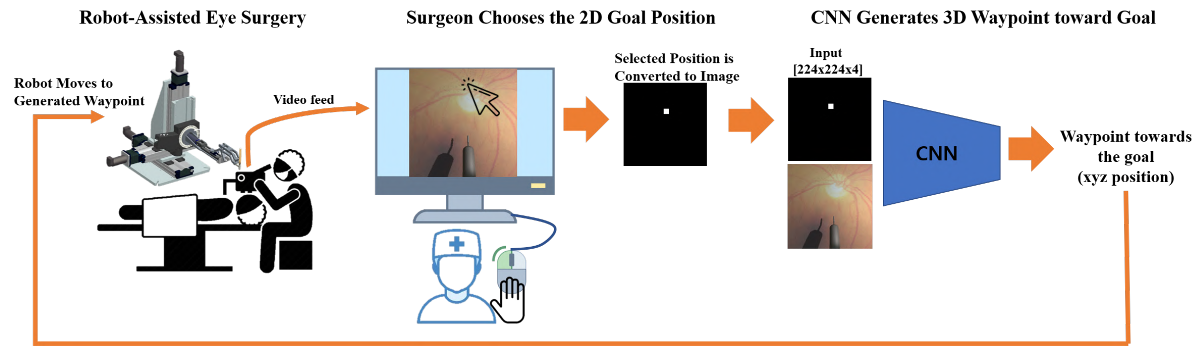

Fig. 2: System setup: a surgeon chooses the goal location in 2D and the network generates a 3D waypoint that navigates the surgical tool toward the selected location. The control cycle is repeated until contact is detected between the tool-tip and the retina, which is detected using a force-sensor at the end-effector.

imitate expert trajectories toward various locations on the retina based on many demonstrations of the tool-navigation task. The input to the network are the monocular top-down view of the surgery through a microscope and user-input defining the 2D goal location to be reached. The advantage of this method is that the user only specifies the goal in 2D, e.g. as simple as clicking the desired location using a mouse (Fig. 2), and the network outputs a 3D waypoint toward the target location on the retina. Since estimating depth is the challenging task for humans, the network takes the burden of extrapolating how to navigate along the depth dimension based on its training experience.

We note that our approach is grounded in the hypothesis that the tool-navigation task may be achieved primarily using vision. In fact, surgeons rely on their vision to localize objects and estimate their spatial relationship to navigate the surgical tool. Furthermore, the surgical scene captures a distinct tool-shadow dynamics which can be useful for estimating proximity between the tool-tip and the retina. Specifically, the tool and its shadow converge upon approaching the retina (Fig. 1), which can be as cues to train the network. In addition, while a complete setup can include stereo vision, in this work we rely on a single camera alone for simplicity. We also utilize a force-sensing modality to detect contact with retina, such that the surgical tool can be stopped upon contact.

The system performance is validated experimentally using both an artificial eye-phantom as well as in simulation employing the Unity3D (Unity Technologies) environment [4]. The main objective is to assess the quality of surgical tool navigation to desired locations on the retina. To achieve this, we employ a batch of benchmark tasks where various positions on the retina are targeted in a grid-like fashion (Fig. 6, 9). For simplicity, we keep the eye position and tool-orientation fixed during the experiments. While this is not a realistic assumption in practice, since the eye could involuntarily move during procedures, our approach can easily extend to the more general setting of different eye rotations through additional training. To test the robustness of our network, we also perform the benchmark task in the presence of unseen distractions in the visual input, such as a light-pipe (used for illuminating the surgery scene) and forceps (used in retinal-peeling) which are commonly used surgical tools. On average, we report that the network achieves less than 137 μm accuracy in various unseen scenarios in real-experiment using a silicone phantom, and 94 μm accuracy in simulation. Lastly, we propose a change to the baseline network resulting in marked improvement in its performance, specifically by training the network using future images along with waypoints as labeled outputs, which turns out to be a richer representation useful for control. We show that learning such auxiliary task improves the performance on the tool-navigation task.

II. Related Work

A. Retinal Surgery

Past works in computer-assisted retinal surgery have focused on state estimation or detection systems to assist surgeons with more information about the surgery. For example, several works have attempted to close the uncertainty gap of estimating the depth between the tool-tip and the retinal surface. Image segmentation can be applied to estimate tool-tip and shadow-tip to model proximity when the tips approach close by a predefined threshold pixel-distance [2], though in a flat environment rather than in realistic concave surface to correctly simulate eye geometry. In addition, stereo vision can be employed to estimate the depth of the tool and the retina respectively to create a proximity detection system [1]. More recently, optical coherence tomography (OCT) was utilized to manually sense depth between the tool-tip and the retina [3], [5].

B. Learning

Various works have shown the effectiveness of deep learning in sensorimotor control such as playing computer games [6], [7] or navigating in complex environments [8], [9], [10], [11], [12], [13]. In particular, the approach employed in our work borrows from the architecture proposed in [12], where a network is trained to drive a vehicle based on user's high level commands such as "go left" or "go right" at an intersection. Similarly, [13], [11] employ topographical maps to communicate the desired route to a destination selected by a user to drive a vehicle. In our work, we also communicate the goal position to navigate the surgical tool as a topographical representation (Fig. 2). Furthermore, several prior works employ the idea of learning auxiliary tasks to

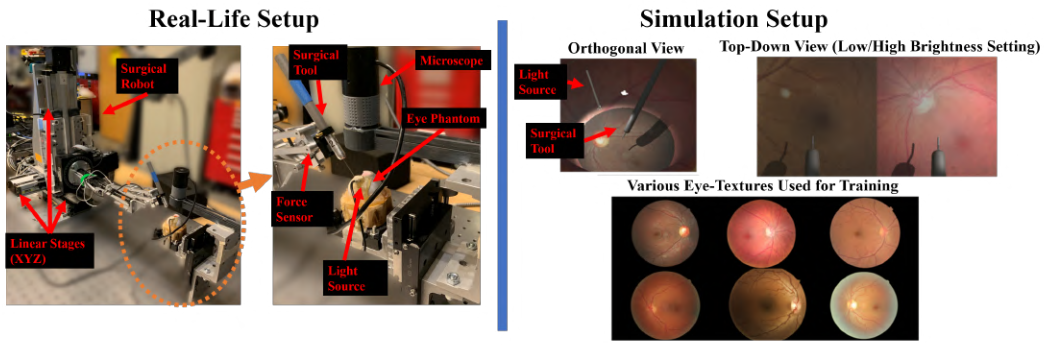

Fig. 3: (Left) Real-life experimental setup using an eye phantom. (Right) Experimental setup in simulation

improve accuracy, such as predicting high-dimensional future image conditioned on input (e.g. goal or action) [14], [15], [16], [17], [18], and learning auxiliary tasks for improved sensorimotor control [19], [13].

III. Problem Formulation

We consider the task of autonomously navigating a surgical tool to a desired location using a monocular surgical image and topographical 2D goal-position specified by the user as inputs (Fig. 2). We formulate the problem as a goal-conditioned imitation learning scenario, where the network is trained to map observations and associated goals to actions performed by the expert. The goal-conditioned formulation is necessary to enable user-control of the network at test time (e.g. navigate the surgical tool to a desired location). Given a dataset of expert demonstrations, $D = \{(o_i, g_i), a_i\}_{i=1}^{N}$, where o_i, g_i and a_i denote observation, goal, and action, respectively, the objective is to construct a function approximator $a = F(o; \theta)$ with parameters θ, that maps observation-goal pairs to actions performed by the expert. The objective function can then be expressed as the following:

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} L(a_i, F(o_i, g_i; \theta)), \tag{1}$$

where L is a given loss function.

In our case, we choose the observation to be an image $o \in \mathcal{I}$ of the surgical scene, the action $a \in \mathcal{W} \subset \mathbb{R}^3$ to be the 3D euclidean coordinates of a point in the surgical workspace $\mathcal{W}$ or a waypoint, and the goal input to be $g_i = (x_i, y_i) \in \mathbb{R}^2$ which specifies the final desired projected 2D position on the retinal surface. Further details on how the expert dataset is collected and network is trained are given next.

IV. Method

A. Eye Phantom Experimental Setup

Our experimental setup consists of the robot, a surgical needle, and a microscope that records top-down view of the surgery as shown in Fig. 3. For our robot platform, we used the Steady Hand Eye Robot (SHER), which is a surgical robot built specifically for eye surgery applications [20]. The surgical needle is attached at the end-effector with thickness $500\mu m$ in diameter (Fig. 6). The artificial eye phantom (i.e. a rubber eye model) is 25.4mm in diameter, slightly larger than a human eye which ranges 20 - 22.4mm [21]. To collect data in our experiments, we control the robot using motors attached on the robot joints. We record the images from the microscope and the robot kinematics from the XYZ motor encoders.

B. Simulation Setup

In simulation, we used Unity3D software to replicate similar experiment scenarios as the actual experiment as shown in Fig. 3. For sense of scale, the thick part of the tool shaft measures $500\mu m$ and the tool-tip measures $300\mu m$ (Fig. 9). Because it is easier to change experimental setup in simulation, we perform domain randomization to change the eye backgrounds and the lighting condition. We also created 15 different eyes, each varying in dimension at 20.4mm, 21.2mm, and 22.4mm. These measurements reflect the minimum, medium, and maximum dimension of human eye sizes [21], [22], and 5 eyes were created for each dimension. The texture of the eyes were obtained from [23]. The goal of domain randomization was so that the network will be agnostic to changing brightness conditions, size of the eye, and the background texture of the eyes for robust generalization. For training, 3 eyes from each size were used, and the remaining 2 eyes from each size were used for testing.

C. Data Collection

In practice, the proposed network must be trained using either expert surgeon trajectories or trajectories known to be safe in achieving the goal. In this work, we do not rely on actual expert surgeon motions, but instead employ an automated data collection setup that exploits ground truth knowledge about the eye geometry, camera calibration, relative localization of tool and eye, and also precise distance to collision and tool-tip force sensing (used to slow-down and stop the motion). Under such controlled conditions, we can generate multiple high-quality trajectories that can be regarded as "expert motions." During inference, the assumption is that none of this information is available.

For the physical phantom-based experiments, we collected 2000 trajectories in total, 1000 in low brightness setting and 1000 in high brightness setting. In simulation, we collected 2500 trajectories under a wide range of brightness conditions, while various eyes with different size and backgrounds were randomly replaced. The procedure for data collection were as

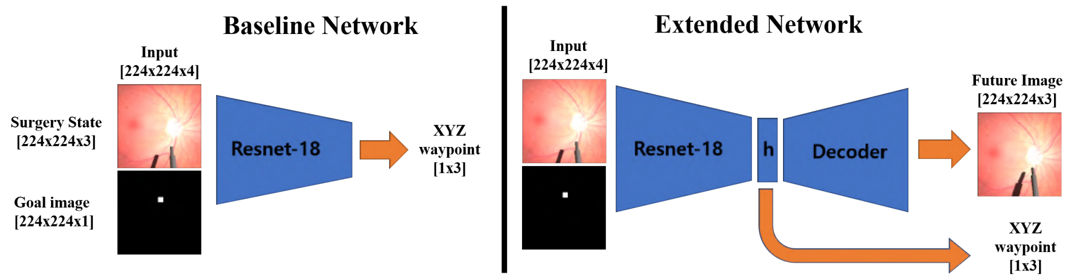

Fig. 4: (Left) Baseline network which predicts incremental waypoint toward the target specified by the user (Right) Extended network which predicts a waypoint and also learns the auxiliary task of predicting the future state.

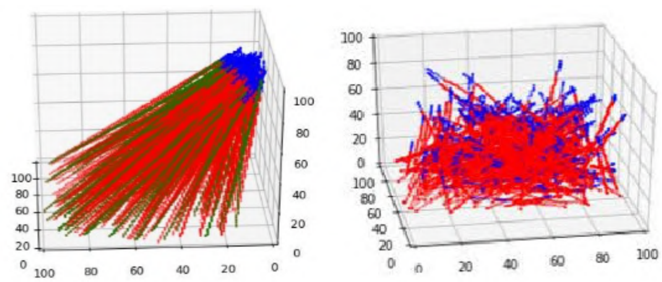

Fig. 5: (Left) training trajectories with small initialization region; blue tips indicate starting position of the trajectory (Right) Training trajectories with large initialization space as data augmentation.

follows: we initialized the tool at a random position in view of the camera, then navigated towards a randomly selected position below the eye in a straight-line trajectory. We use straight-line trajectories as a way to generate predictable and simple ground-truth expert data. When collision was detected between the tool and the eye phantom using a force sensor at the end-effector (Fig. 3), we logged the images and the robot kinematics of the trajectory. For simplicity, we kept the orientation of the tool fixed and only moved the xyz-pose of the tool. The position of the eye remained fixed as well. These conditions apply for both real-life experiments and in simulation.

To synthesize the goal for each trajectory, we used the last xy-coordinate of the executed trajectory and plotted it as a white square with dark background as illustrated in Fig. 2. Effectively, we changed the 2-dimensional coordinate representation of the goal to an image representation. This is an important design choice because ultimately our goal was to train the network based on visual goals that spatially corresponded with the surgical scene. This can be useful in an application where a surgeon may simply use a mouse to click the goal location in the visual-feed of the surgery, then the corresponding image of the white square can be generated, and the network will navigate the surgical tool to the exact location of the white square corresponding to the surgical scene (Fig.3). For such an application, it requires that the dataset contain goal images where the center of the white square corresponds exactly with the final tool-tip position, or with some consistent offset. However, this requires manual annotation to create the goal images. In this work, however, our objective is to test the accuracy with which the network can navigate to the desired location given a particular goal image. Thus, we do not annotate the images manually. Furthermore, this approach is necessary because it allows us to calculate tool-navigation errors during inference; since we know that the plotted position of the white square is the desired final xy-tool position in the surgical workspace, we can calculate the final accuracy by comparing the plotted values to the final landing position of the tool.

D. Network Details and Training

The input to the network are the current image of the surgery (224x224x3) and the goal image (224 x 224 x 1) stacked along the channel dimension, yielding a combined dimension of (224 x 224 x 4). The output of the network is a xyz-waypoint in the surgical workspace (XYZ values or 3-dimensional vector) which the network must travel in order to reach closer to the target location on the retina. Specifically, for a single trajectory consisting of n frames $I_1, ...I_n \in \mathcal{I}$, n robot-kinematic positions $p_1, \ldots, p_n \in \mathbb{R}^3$, and the goal image coordinates $g \in \mathbb{R}^2$ specific to this trajectory, a single sample is then expressed as $(\text{input}, \text{output}) = ((I_t, g), p_{t+d})$, for $t = 1, \ldots, n–d$, where d is a parameter denoting the *look-ahead* of the commanded action, which is used as a feed-forward reference signal to the robot. We chose $d = 8$, which is equivalent to approximately $70\mu m$ apart between learned waypoints to ensure that the network moved the surgical tool by a noticeable distance every control cycle. All inputs and outputs, including the waypoints and the images, are normalized from 0 to 1 via min-max scaling. The complete data set D is constructed using multiple such trajectories and their corresponding samples. Internally, the network maps the goal g into an image $I_g \in I$ which is concatenated with the actual camera image I_i to form the complete network input.

We experimented with two architectures, a baseline network that predicts an xyz waypoint and another network that predicts an xyz waypoint plus the future image as shown in Fig. 4. We refer to the latter network as the extended network. Predicting the future image was considered as an auxiliary objective to learn a richer representation for control. Intuitively, the motivation was that if a network could generate the correct xyz-waypoint for navigation and also predict what that future looked-like, it would arguably have learned a richer understanding of the navigation task. The benefit of using auxiliary objective for learning sensorimotor

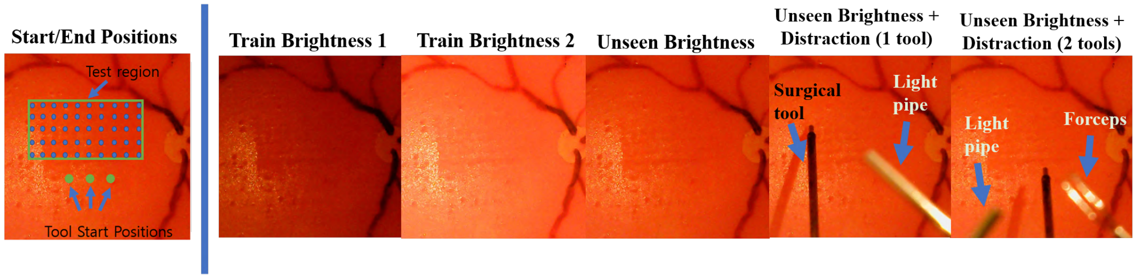

Fig. 6: (Left) Example start positions of the surgical tool in real-experiment (Right) Various conditions used to train/test the network.

control has been similarly demonstrated in [19][13]. For the extended network, a single sample for training can be expressed as $(\text{input}, \text{output}) = ((I_t, g), (p_{t+d}, I_{t+d}))$. In the following, we discuss each network in greater detail.

TABLE I: Eye Phantom Experiment Results

Test Condition	Baseline Network Error (mm)	Extended Network Error (mm)
Train Low Brt.1	0.134	0.139
Train High Brt. 2	0.092	0.108
Unseen Brt.	0.177	0.127
Unseen Brt. + Distr. (1 tool)	0.165	0.146
Unseen Brt. + Distr. (2 tools)	0.155	0.137
"Unseen" Avg. (above 3 rows)	**0.166**	**0.137**

TABLE II: Eye Phantom Training Results

Axes	Baseline Val. Acc. (%)	Extended Network Val. Acc. (%)
X	82.0	82.8
Y	76.0	76.7
Z (Depth)	60.8	61.9
XYZ Total Sum	**218.8**	**221.3**

1) Baseline Network: The baseline network architecture is shown in Fig. 4. The baseline architecture aims to learn the tool-navigation task in the simplest manner using a feed-forward network. We use Resnet-18 [24], which encodes the high dimensional input image (224 x 224 x 4) to 512-dim feature vectors. To learn the waypoints or the action output, we discretize the continuous x, y, and z coordinate representation into 100 steps. We discretized the action space because training the network with cross-entropy loss yielded better results than using root-mean-squared-error loss in the continuous case. Specifically, we add a fully-connected layer outputting 300 neurons on top of the 512-feature vectors, where each 100 neurons is a discretized representation of the continuous x, y, and z coordinates of the surgical workspace respectively. The network was trained using cross-entropy loss with Adam optimizer [25] with an initial learning rate of 0.0003 and batch size of 170. After discretizing the space, the loss function is redefined as

$$L(b, \hat{p}) = \sum_{j\in\{x,y,z\}} \sum_{c=1}^{M_j} -b_{j,c}\log(\hat{p}_{j,c}), \tag{2}$$

where $b_{j,c}$ are binary indicators for the true class label c, and $\hat{p}_{j,c}$ are the predicted probability that the coordinate j is of class c. The cost combines the errors for all three dimensions $j \in \{x, y, z\}$. As specified above, we employed $M_x = M_y = M_z = 100$ bins.

2) Baseline + Predicting Future Image (Extended Network): The extended architecture aims to achieve the baseline task and additionally predict future images. The architecture is shown in Fig. 4. On top of the Resnet-18 architecture, a decoder network with skip-connections is added, similar to the U-Net architecture [26]. After encoding the images to 512-feature vectors, six deconvolutions are performed to obtain the future image, which is the same dimension as the input surgical state (224 x 224 x 3). After each deconvolution, the tensors of the same dimension from the encoder network are concatenated, and two additional convolutions are performed to combine the concatenated tensors and refine the up-sampled tensors. Learning to predict future image in addition to the waypoints is intended to learn a richer representation for control than the baseline network. The waypoints were trained using cross-entropy loss similar to the baseline network and the future prediction was trained using RMSE function. The network is trained using Adam optimizer with an initial learning rate of 0.0003 and batch size of 120. The combined loss function is given as

$$L((b, I), (\hat{p}, \hat{I})) = \sum_{j\in\{x,y,z\}} \sum_{c=1}^{M_j} -b_{j,c}\log(\hat{p}_{j,c}) + (I - \hat{I})^2, \tag{3}$$

where $\hat{I}$ denotes the future-image prediction by the network and I denotes the label for the future image. The second term on the right-hand side is a pixel-wise subtraction, and the first term follows as previously defined in the baseline network. To balance the loss functions, drop-out approach was used where we performed back-propagation 70% of the time for the future-image loss term.

E. Data Augmentation

For robust learning, we utilized data augmentation, such as drop-out of pixels (maximum of 2x2 size, less than 70 distributed across the image), added Gaussian noise, and jittered the image (randomize the brightness, contrast, saturation, and hue). These augmentations were enabled 20 percent of the time each epoch and the gain of the jitter was set at 0.08. We also expanded the initialization space so that the network could reach the same target location from various initial positions as shown in Fig. 5. This effectively enabled the network to recover from mistakes when it deviated

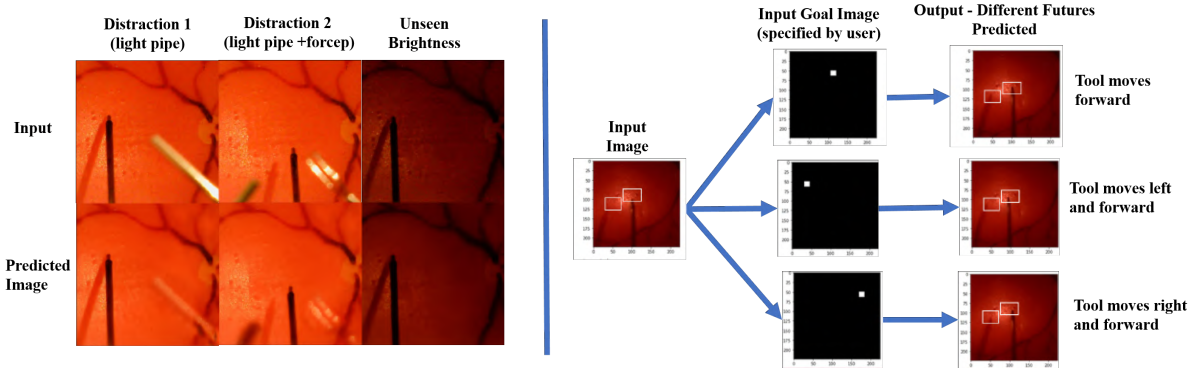

Fig. 7: (Left) Future prediction by the extended network under various unseen conditions (Right) Different futures predicted by changing the goal input (rectangle frames are fixed, added to clarify shifted positions of the tool); visual predictions can be useful to surgeons to foresee the result of the future before executing the predicted waypoint

from its hero path, and it enabled the network to reach various goal locations from any reasonable initialization position. Data augmentation and expanding the initialization region was crucial to achieving good performance in real-life experiments and in simulation.

V. Results and Discussion

A. Real-Life Experiment Results

To assess the accuracy of our networks, we performed benchmark experiments where the baseline and the extended network visited 50 predefined locations in the training region in grid-like fashion (5 x 10), starting from 3 different initial locations as shown in Fig. 6. The objective of such experiment was to test how accurately the network could navigate to various targeted locations, given various goal inputs. We tested each network in the following familiar and unseen environments to test their robustness (Table I): two training conditions (low, high brightness settings), in one unseen brightness setting, in the same unseen brightness setting plus with dynamically moving distraction using a light-pipe tool, and in the same unseen brightness setting plus with two dynamically moving distractions using a light-pipe and forceps, both of which are commonly-used retinal surgery tools (Fig.6). For experiments with tool-distractions, we only tested from the right-most initial position out of the three, since a human had to hold the tools throughout the long experiments. The light-pipe was dynamically maneuvered to follow the tool-tip, and the forceps was held by-hand on the opposite side. Both tools occasionally occluded the surgical tool and its shadow.

Our experimental results are summarized in Table I and the executed trajectories are shown in Fig. 10. The table contains numeric xy-error values in reaching the goal position under various test conditions. Since the eye position is fixed during training and experimental validation, the error can be calculated by comparing the input goal-image coordinate (x, y) against the final landing position of the surgical tool $(x^{'}, y^{'})$ after the trajectory execution is complete (e.g. when force is detected using the force sensor). Thus, the error reported in Table I is calculated

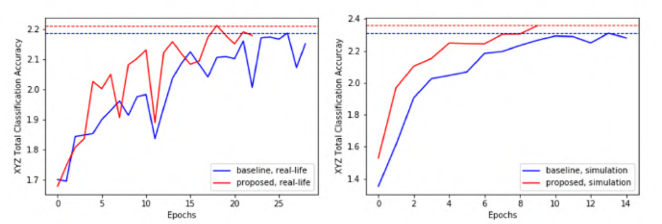

Fig. 8: (Left) Waypoint prediction accuracies on the "real-life" validation dataset and in (Right) simulation. Y-axis is the sum of the classification accuracies for xyz axes (i.e. maximum possible is 3.0). Dashed lines mark the maximum accuracy achieved. It can be seen that the extended network trains faster and is more data-efficient than the baseline network in both simulation and real-life

using the formula $\sqrt{(x-x')^2+(y-y')^2}$. The accuracy reported in Table II are the classification accuracies achieved on the validation dataset offline, not the online benchmark experiments. In Table II, we are able to report errors in the z-axes (depth) because we have ground-truth xyz-values of the full trajectory from the previously collected dataset.

Our results show that the both baseline and extended network generalizes well to unseen scenarios, achieving $166\mu m$ and $137\mu m$ in error, even in the presence of unseen brightness conditions and unseen surgical tools significantly occluding the scene. The extended network also performed marginally better than the baseline network. This result is expected given the higher accuracy achieved by the extended network in the validation dataset, achieving 2.5% higher accuracy than the baseline (Table II). Also, as shown in Fig. 8, the extended network trains faster and is more data-efficient than baseline network, achieving best classification accuracy on the 18th epoch versus 26th epoch by the baseline network. In addition to improving the baseline network performance, the extended network is able to predict clear future images. Clear visual predictions could enable surgeons to intuitively understand the future outcome of the surgery, instead of trying to make sense of numeric outputs from the network. As shown in Fig. 7, the extended network can imagine different futures depending on various goal inputs (e.g. move the tool forward, left, right), recognize the surgical tool as a dynamic object apart from the static

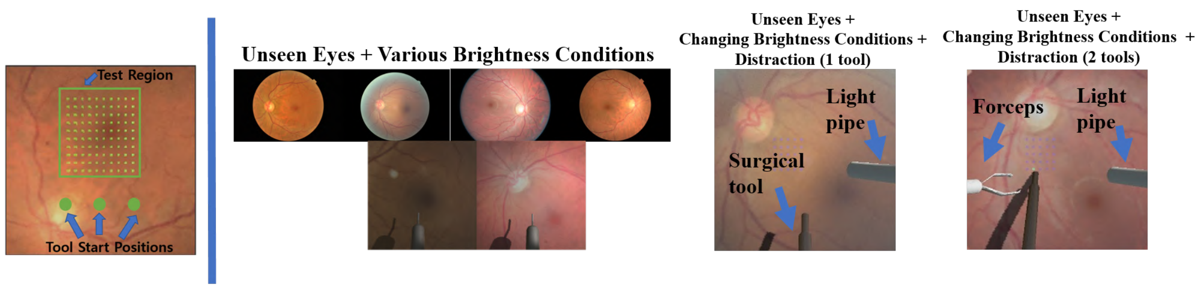

Fig. 9: (Left) Three start positions and test region of the surgical tool in simulation. (Right) Various unseen conditions to test the network

background, and also reliably reconstruct unseen objects (e.g. forceps and light-pipe) and unseen brightness settings. Such visual predictions could be useful in future applications for visual task planning, where a tool trajectory may be visually planned then selected based on recurrent roll-out of various future outcomes.

B. Simulation Experiment Results

Similar to real-life experiments, we performed benchmark tests where each baseline and extended network visited 100 predefined locations in the training region in grid-like fashion (10 x 10), starting from 3 different initial locations as shown in Fig. 9. We tested each network in the following conditions, which are different from real-life experiments: 9 training eyes under random brightness setting ranging from low to high, 6 unseen eyes under random brightness ranging from the same low to high brightness, in the presence of unseen surgical tool using a light-pipe, and in the presence of two unseen surgical tools using a light-pipe and forceps (Fig.9). For experiment with tool-distractions, we only tested from the middle initial position out of the three. Both the light-pipe and forceps were moved randomly every frame to imitate hand-tremor, and both tools occasionally occluded the surgical tool and its shadow.

The simulation results are summarized in Table III and the executed trajectories are shown in Fig. 10. The errors shown in Table III are calculated using the same formula mentioned in the real-life experiment results, specifically using the formula $\sqrt{(x-x^{'})^2+(y-y^{'})^2}$ where (x, y) denotes input goal-image coordinate and $(x^{'}, y^{'})$ denotes the final landing position of the surgical tool after trajectory execution. Similarly, Table IV shows network results on the validation dataset. Our results show that both baseline and extended networks achieve good performance and can generalize robustly to unseen scenarios, even in the presence of unseen eye backgrounds and unseen surgical tools occluding the scene. Similar to real-life experiments, the extended network also performed marginally better than the baseline network. This result is expected since the extended network achieved 4.9% higher accuracy than the baseline network in the validation dataset (Table IV). Similar to real-life experiments, the extended network is also more data-efficient than the baseline network, achieving maximum accuracy at 13th epoch versus 9th epoch by the baseline network (Fig.8) and achieving significantly higher maximum validation accuracy. Regarding the future predictions generated by the extended network, the future predictions were clear and interpretable, however, it was not able to precisely predict the surgical tool at the expected future position based on changing goal input. We conjecture that using naive squared-error loss function is not sufficient for accurate tool-position reconstruction due to higher sample complexity of the changing backgrounds. One possible solution is to use a weighted squared-error loss where more weight is carried on learning the pixels corresponding to the surgical tool-tip.

TABLE III: Simulation Experiment Results

Testing Condition	Baseline Network Error (mm)	Extended Network Error (mm)
Train	0.107	0.098
Unseen Eyes	0.102	0.096
Unseen Brt. + Distr. (1 tool)	0.140	0.100
Unseen Brt. + Distr. (2 tools)	0.169	0.087
"Unseen" Avg. (above 3 rows)	**0.137**	**0.094**

TABLE IV: Simulation Training Results

Axes	Baseline Val. Acc. (%)	Extended Network Val. Acc. (%)
X	78.9	81.4
Y	84.8	84.0
Z (Depth)	67.3	70.6
XYZ Total Sum	**231.0**	**235.9**

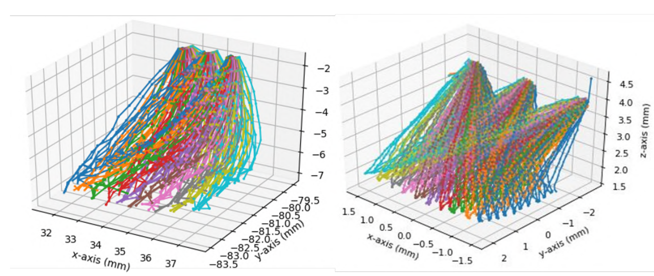

Fig. 10: (Left) Trajectories executed in real-life in unseen brightness condition (Right) trajectories executed in real-life in changing brightness condition + unseen eyes

VI. Conclusions

In this work, we demonstrate end-to-end autonomous navigation of a surgical tool inside the eye using deep networks. We demonstrate a baseline approach and propose an improved network architecture that predicts a future image, which improved the baseline performance. We also show that the network generalizes well to unseen brightness setting and in the presence of unseen distractions, overall achieving $137\mu m$ error on average in real-life experiments and $94\mu m$ error in simulation. In future work, we hope to test our framework in more realistic scenarios. For example, in real surgery, the surgical tool is constrained at a point and can only slide and tilt through a scelerotomy port on the eye-ball. The eye lens also introduces aberrations by distorting the surgical scene. It may also be possible to integrate the tool-navigation task as a sub-task to achieve more complex tasks. We also hope to propose frameworks that enable more efficient learning requiring less demonstration data in the future.

VII. Acknowledgements

This work was supported by U.S. National Institutes of Health under grant 1R01EB023943-01. The authors would also like to thank Gowtham Garimella, Mathew Sheckells, and Niravkumar Patel for helpful discussions.

References

[1] R. Richa, M. Balicki, R. Sznitman, E. Meisner, R. Taylor, and G. Hager, "Vision-based proximity detection in retinal surgery," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 8, pp. 2291–2301, Aug 2012.

[2] T. Tayama, Y. Kurose, M. M. Marinho, Y. Koyama, K. Harada, S. Omata, F. Arai, K. Sugimoto, F. Araki, K. Totsuka, M. Takao, M. Aihara, and M. Mitsuishi, "Autonomous positioning of eye surgical robot using the tool shadow and kalman filtering," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, July 2018, pp. 1723–1726.

[3] M. Ourak, J. Smits, L. Esteveny, G. Borghesan, A. Gijbels, L. Schoevaerdts, Y. Douven, J. Scholtes, E. Lankenau, T. Eixmann, H. Schulz-Hildebrandt, G. Hüttmann, M. Kozlovszky, G. Kronreif, K. Willekens, P. Stalmans, K. Faridpooya, M. Cereda, A. Giani, G. Staurenghi, D. Reynaerts, and E. B. Vander Poorten, "Combined oct distance and fbg force sensing cannulation needle for retinal vein cannulation: in vivo animal validation," *International Journal of Computer Assisted Radiology and Surgery*, vol. 14, no. 2, pp. 301–309, Feb 2019. [Online]. Available: https://doi.org/10.1007/s11548-018-1829-0

[4] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," 2018.

[5] J. Smits, M. Ourak, A. Gijbels, L. Esteveny, G. Borghesan, L. Schoevaerdts, K. Willekens, P. Stalmans, E. Lankenau, H. Schulz-Hildebrandt, G. Httmann, D. Reynaerts, and E. B. Vander Poorten, "Development and experimental validation of a combined fbg force and oct distance sensing needle for robot-assisted retinal vein cannulation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 129–134.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[7] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," 2016.

[8] X. Wang, Q. Huang, A. elikyilmaz, J. Gao, D. Shen, Y. fang Wang, W. Y. Wang, and L. Zhang, "Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation," *ArXiv*, vol. abs/1811.10092, 2018.

[9] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016.

[10] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 3530–3538.

[11] A. Amini, W. Schwarting, G. Rosman, B. Araki, S. Karaman, and D. Rus, "Variational autoencoder for end-to-end control of autonomous driving with novelty detection and training de-biasing," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 568–575.

[12] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–9, 2018.

[13] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," 2018.

[14] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2863–2871. [Online]. Available: http://papers.nips.cc/paper/5859-action-conditional-video-prediction-using-deep-networks-in-atari-games.pdf

[15] O. Gafni, L. Wolf, and Y. Taigman, "Vid2game: Controllable characters extracted from real-world videos," *CoRR*, vol. abs/1904.08379, 2019. [Online]. Available: http://arxiv.org/abs/1904.08379

[16] C. Paxton, Y. Barnoy, K. Katyal, R. Arora, and G. D. Hager, "Visual robot task planning," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 8832–8838.

[17] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2786–2793, 2016.

[18] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 5967–5976.

[19] P. W. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell, "Learning to navigate in complex environments," *ArXiv*, vol. abs/1611.03673, 2016.

[20] A. Üneri, M. A. Balicki, J. Handa, P. Gehlbach, R. H. Taylor, and I. Iordachita, "New steady-hand eye robot with micro-force sensing for vitreoretinal surgery," in *Biomedical Robotics and Biomechatronics (BioRob), 2010 3rd IEEE RAS and EMBS International Conference on*. IEEE, 2010, pp. 814–819.

[21] Bekerman, Gottlieb, Paul, and Michael, "Variations in eyeball diameters of the healthy adults," Nov 2014. [Online]. Available: https://www.hindawi.com/journals/joph/2014/503645/

[22] S. Vurgese, S. Panda-Jonas, and J. B. Jonas, "Scleral thickness in human eyes," *PLOS ONE*, vol. 7, no. 1, pp. 1–9, 01 2012. [Online]. Available: https://doi.org/10.1371/journal.pone.0029692

[23] "Diabetic retinopathy detection." [Online]. Available: https://www.kaggle.com/c/diabetic-retinopathy-detection/overview/description

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.