2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Real-time Data Driven Precision Estimator for RAVEN-II Surgical Robot End Effector Position

Haonan Peng, Xingjian Yang, Yun-Hsuan Su, Blake Hannaford

Abstract— **Surgical robots have been introduced to operating rooms over the past few decades due to their high sensitivity, small size, and remote controllability. The cable-driven nature of many surgical robots allows the systems to be dexterous and lightweight, with diameters as low as 5mm. However, due to the slack and stretch of the cables and the backlash of the gears, inevitable uncertainties are brought into the kinematics calculation [1]. Since the reported end effector position of surgical robots like RAVEN-II [2] is directly calculated using the motor encoder measurements and forward kinematics, it may contain relatively large error up to 10mm, whereas semi-autonomous functions being introduced into abdominal surgeries require position inaccuracy of at most 1mm. To resolve the problem, a cost-effective, real-time and data-driven pipeline for robot end effector position precision estimation is proposed and tested on RAVEN-II. Analysis shows an improved end effector position error of around 1mm RMS traversing through the entire robot workspace without high-resolution motion tracker. The open source code, data sets, videos, and user guide can be found at //github.com/HaonanPeng/RAVEN_Neural_Network_Estimator.**

I. Introduction

A. Background

Robot-assisted Minimally Invasive Surgery (RAMIS) opens the door to collaborative operations between experienced surgeons and surgical robots with high dexterity and robustness [3]. While surgeons are in charge of decision making and robot manipulation through teleoperation, robots follow the trajectory commands. In abdominal RAMIS, the precision requirement is in millimeter scale. With surgeons manually closing the loop, accuracy of the reported robot end effector pose is not a big concern. In recent years, surgical robot intelligence has emerged in medical robotics research, where repetitive tasks like ablation [4] and debridement [5] can be conducted autonomously under supervision of surgeons. Intelligent robot navigation agents are now being developed to incorporate raw teleoperation commands, tremor canceling [6] and motion compensation of the dynamic surgical scenes [7] [8]. Moreover, vision-based force estimation [9] in RAMIS shows promise. In all these applications, precise end effector positioning of the surgical robot is a requirement.

Many surgical robots are designed with cable transmissions with motors mounted at the base to allow lighter and more compact arms. Cable dynamic properties such as stiffness and internal damping are significant and are known to vary as a function of tension [10]. Therefore, the kinematics based end effector poses reported by the robot from motor sensors are prone to error.

B. Related Work

To compensate for the inaccuracy, an intuitive approach is to take additional sensor measurements by applying a motion tracker at the surgical tooltip or a joint encoder on each robot joint depending on whether to resolve the problem in cartesian or joint level. Drawbacks occur in both solutions. In the former case, complications occur during the required high heat sterilization procedure [10]. The latter, however, introduces complexity keeping the sensor wires and robot cables compact. Alternatively, real-time video streams from endoscopes are used as an additional cue for end effector pose estimates. But extracting pose information from vision alone can be challenging with the highly dynamic and reflective surgical scenes in real-world operations [11]. Recently, online estimation systems are proposed to provide a robust and precise end effector position prediction.

Haghighipanah et al. [1] proposed a joint level model-based approach for RAVEN-II pose estimation using an unscented Kalman filter [12]. Although improved results were shown for the first three joints, the experiments were limited to repeatedly picking up a fixed mass. Also, joint pose estimates for the last four surgical tool joints were beyond the scope of the study and were suggested to be corrected through vision. In 2018, Seita et al. [13] presented a data-driven Cartesian pose calibrator for the daVinci Research Kit experimental surgical robotic platform (dVRK) [14]. That study successfully achieved autonomous surgical debridement through a two-phase calibration procedure. The result shows high precision in both position and orientation, and is tailored to the specific surgical task of debridement. In 2014, Mahler et al. [15] used Gaussian process regression and data cleaning to reduce the error of RAVEN-II end effector, and by including the velocity information, the accuracy could be further improved.

C. Contributions

In this work, the authors built a robot position precision estimator on RAVEN-II that is not task-specific and spans the robot workspace. Thus, a pipeline was built to collect precise RAVEN-II position data traversing the workspace from teleoperation trials by human operators. The ground truth position was carefully derived through calibrated stereo vision. Finally, the dataset was used to train a neural network

Haonan Peng, Xingjian Yang, Yun-Hsuan Su and Blake Hannaford are with the University of Washington Department of Electrical and Computer Engineering, 185 Stevens Way, Paul Allen Center - Room AE100R, Campus Box 352500, Seattle, WA 98195-2500, USA. {penghn, yxj1995, yhsu83, blake}@uw.edu

978-1-5386-2512-5/18/$31.00 ©2020 IEEE

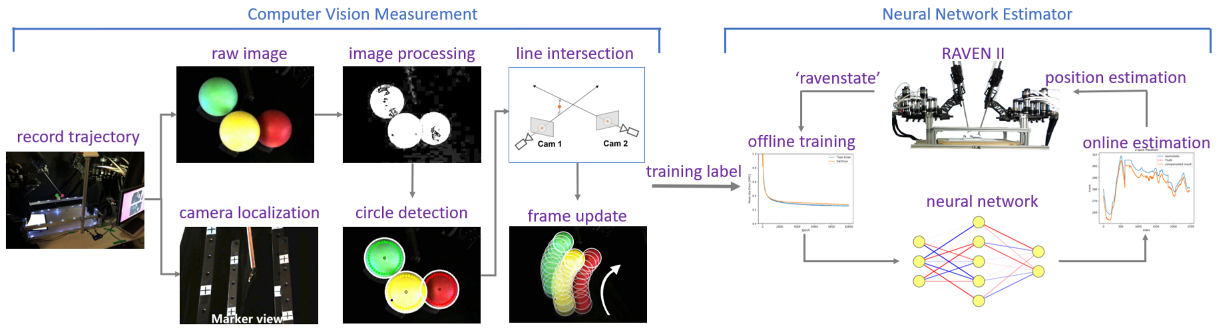

Fig. 1: Workflow for proposed online position precision estimation system. Pipeline entails vision based ground truth measurement (left) and data driven robot position estimator (right). Training labels passing from left to right are true RAVEN-II position information.

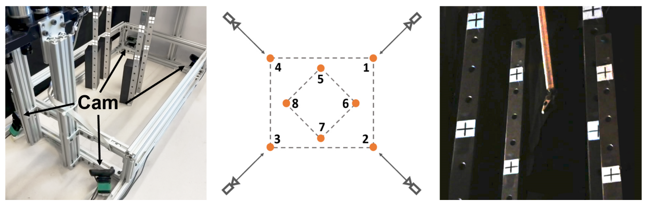

Fig. 2: 16 marker points used for camera pose calibration are placed at 8 ground locations in robot workspace.

model that estimates position error. To the best of the authors' knowledge, this work is first to simultaneously

1) introduce a cost-effective approach for vision-based precise robot position data collection;
2) train a neural network model with 1000Hz detailed sensor and controller state information as input;
3) implement and analytically quantify the performance of a data-driven precision estimator of the robot position in the entire robot workspace.

II. Methods

A. System Workflow

The online end effector pose estimation system consists of two phases - vision-based ground truth measurement and neural network estimator, as shown in Fig. 1.

1) Phase One: 4 webcams are mounted with poses determined by a two-phase calibration procedure (Fig. 2, left). During data recording, 3 distinct colored balls are fixed to the end effector as markers (Fig. 6 left). The balls are only used to collect training data and are removed in real operation. In addition, the balls and the holder are hollow and very light, for a negligible load to the system. For each image frame, preprocessing steps are performed and followed by Hough circle detection [16]. A frame update algorithm is then adopted to prevent false positives by comparing circles detected in subsequent images. Finally, the 3 circle centers yield ground truth end effector position up to 0.5 mm accuracy.

2) Phase Two: There are 2 stages in phase two - offline training and online estimation. The collected position data is used to train a neural network. Next, the trained neural network model provides online estimation of the end effector position with an average precision of around 0.8 mm, a significant improvement from the RAVEN-II feedback position.

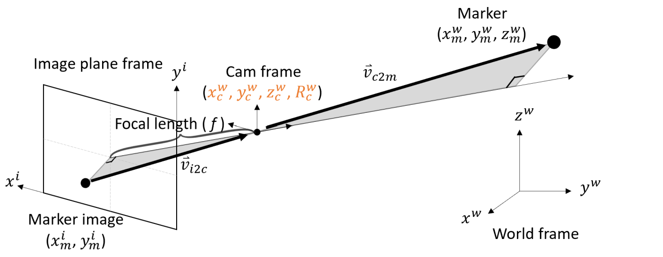

Fig. 3: Geometric relations among image plane, camera and marker.

B. Camera Localization

Since the goal of phase one is to obtain ground truth robot position through vision, accurate camera pose identification in relation to the robot frame is crucial. The existing camera extrinsic matrix solver in MATLAB [17] requires placing a chessboard at a known pose with respect to the world frame. The resultant camera pose from said solution contains error up to 40 millimeters if only 1-5 chessboards are used. Although placing more chessboards (usually more than 20) theoretically improves precision, the problem of acquiring accurate chessboard poses also becomes increasingly difficult due to the constrained workspace. As a practical alternative, we use 1 chessboard and 16 marker points together (Fig. 2). The chessboard provides a rough cameras pose estimate and the 16 markers further refine it.

We define world frame w, chessboard frame b, camera frame c and image frame i. The camera position and orientation with respect to the world coordinate are respectively $p_c^w = [x_c^w \; y_c^w \; z_c^w]^T$ and $R_c^w \in \mathbb{R}^{(3\times3)}$.

Similarly, the marker position in the world coordinate is $p_m^w = [x_m^w \; y_m^w \; z_m^w]^T$ whereas its 2D projection on the image frame is $p_m^i = [x_m^i \; y_m^i]^T$. Let distance vectors

$$\vec{v}_{c2m} = p_m^w - p_c^w \,, \tag{1}$$

$$\vec{v}_{i2c} = R_c^w \begin{bmatrix} p_m^i \\ f \end{bmatrix} = R_b^w \cdot R_c^b \begin{bmatrix} x_m^i \\ y_m^i \\ f \end{bmatrix} \,, \tag{2}$$

where f is the camera focal length. According to the geometric relations illustrated in Fig. 3, the following three equations hold true for each marker from each camera view:

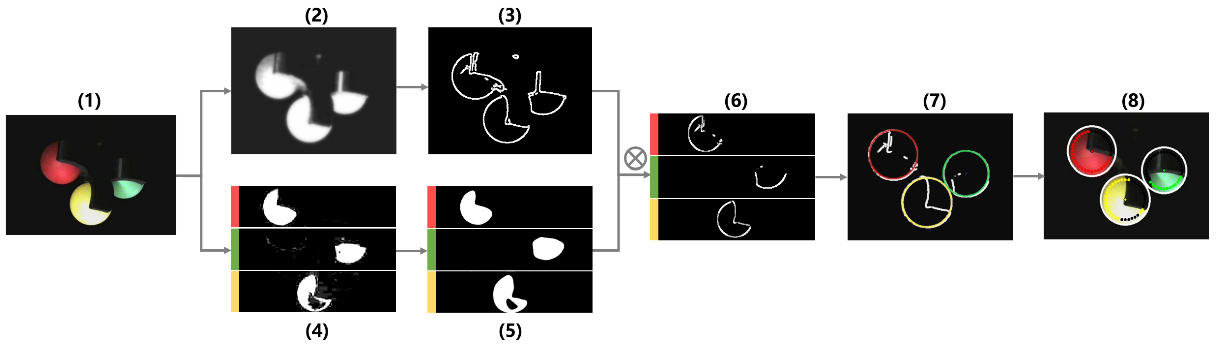

Fig. 4: Circle detection method and the preprocessing steps. Color image (1) is first converted to gray and Gaussian blur is applied (2). Then Canny edge (3) is taken from the gray image. In the other pathway, the color image (1) is divided into red, green, and yellow color channels (4). Then Gaussian blur with σ of 10% radius followed by binarization is applied (5). Image (6) is elementwise multiplication of (3) and (5). Hough circle detection is applied in each channel in image(6), which combines to result in (7). Finally, image (8) is a color check near the circle border to eliminate false positives.

$$\frac{\vec{v}_{c2m}(k)}{\vec{v}_{i2c}(k)} = \frac{\|\vec{v}_{c2m}\|_2}{\|\vec{v}_{i2c}\|_2} \quad \forall k \in [1, 2, 3]\,, \tag{3}$$

where $\vec{v}(k)$ represents the kth entry in the distance vector.

There are a total of 6 unknown variables in p_c^w and R_c^w. With 16 equation sets (3) yield from all 16 marker points, the Levenberg-Marquardt algorithm [18] finds optimal camera pose with initial guess $p_c^{w\prime}$ and $R_c^{w\prime}$ from the chessboard.

C. Auto Hough Circle Detection

Localization of the colored balls assists with end effector position acquisition. 2D circle detection is the first step to localize colored balls. Hough circle detection [16] is chosen due to its robustness to occlusion. Yet, it is sensitive to bright edge noises, so heuristic parameter tuning and image enhancements are necessary. Fig. 4 shows the pre-processing steps with the following design details:

a) Edge Detection (1)-(3): These steps increase the signal to noise ratio and clear up noisy edges.

b) Color Segmentation (1)-(4)-(5): After this, circle segments in the color channels are intentionally enlarged.

c) Border Refinement (3)(5)-(6): Since circles in b) are larger than a). The circle canny edges [19] will be kept, and any edges outside the circles will be removed.

d) Circle Identification (6)-(7)-(8): After circles are detected, sample points near the border are used to ensure the circle color matches the target color with similarity more than 25%. Otherwise, the circle is considered a false detection.

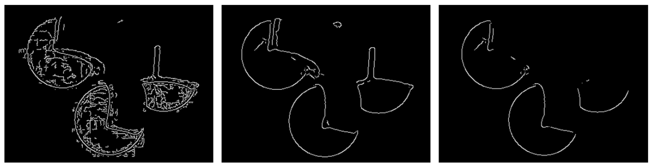

Fig. 5: Comparison of Canny edge on original image (left), normal blurred image (middle) and processed image by our system (right).

The 'HoughCircle' function in OpenCV [20] is used to detect circles. Two heuristically chosen parameters are inverse ratio $dp = 1$ and high Canny threshold $para1 = 100$. To improve adaptability, an automated parameter tuning algorithm is designed for the accumulator threshold $para2$, where small values indicate a higher chance of false positives.

Suppose k circles are expected, the algorithm uses bisection to approach the lower bound of $para2$, such that 'HoughCircle' comes close to returning $(k+1)$ circles but still returns k. In the case where $para2$ jumps between $(k-1)$ and $(k+1)$, increasing σ in Gaussian blur helps, but is conducted only if necessary - due to decreased precision.

D. Frame Update Algorithm

There are 2 main purposes of the frame update algorithm. The first is to decide if any camera is returning false circles. The second is to provide other parameters to 'HoughCircle', including the minimum distance between circle centers d_{min}, and extrema of circle radius r_{min} and r_{max}. Setting a low d_{min} and wide range between r_{min} and r_{max} allows more chance to detect circles, but it also increases computational cost and the risk of false positives. Thus, under smooth end effector motion, d_{min}, r_{min} and r_{max} values are bounded by the detection result from the previous frames.

There are a total of 4 cameras. For each ball, there should ideally be 4 circles detected - one corresponding to each of the 4 camera views. After the frame update algorithm, each detected circle is proclaimed as either effective or suspended.

Every circle is initialized as effective. If any of the conditions below holds true, an effective circle will be suspended:

- The movement of the circle center in successive frames exceeds a predetermined motion threshold.
- Color check in fig. 4 (8) fails to match target color by 25%.
- Hough circle detection does not return a circle, even with the largest tolerable σ for Gaussian blur.

On the other hand, if all the following conditions are satisfied for more than 5 frames, a suspended circle will become effective:

- The Hough circle returns result normally.
- Color check in fig. 4 (8) is successful.
- The 3D reconstructed ball center position from effectively detected 2D circles is consistent with that of this suspended 2D circle information.

A line that connects a 2D effective circle center and its associated camera center forms a 3D ray. The 3D position of the ball center is the intersection point of rays. These rays usually do not intersect perfectly, and the midpoint of the common perpendicular of the two rays is selected to be the intersection point. When the number of effective circles is larger than 2, the mean position is taken from each pair of rays. Thus, 2 or more effective circles of the same ball must be detected at any time instance for successful localization. If not, the system skips a few frames and restarts, all while raising the tolerable range for the parameters. Once there are enough effective circles, the parameter tolerance converges back down.

E. End Effector Localization

As Fig. 6 shows, the 3 colored balls are fixed around the end effector center (the black point, which is defined by the RAVEN-II system). The end effector center position shares the same plane with the 3 ball centers. The distances between the end effector center and each ball center are defined as $d = 38$ mm and each ball has radius $r = 20mm$. An end effector coordinate frame is designed for RAVEN-II, where the origin is the end effector point, the X axis points toward the green ball center, and the Y axis points at the yellow ball center, which is also co-linear with RAVEN-II axis X_{6b} [21].

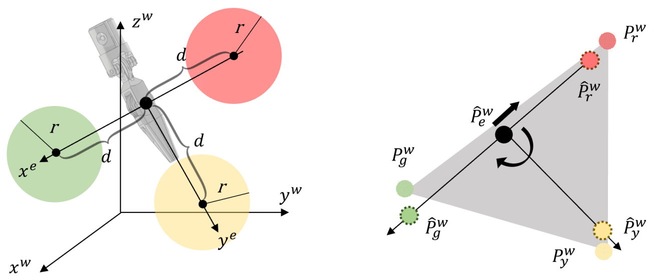

Fig. 6: Geometric relations between ball centers and end effector position.

The detected ball centers in the world coordinate frame are: green $p_g^w = [x_g^w\, y_g^w\, z_g^w]^T$, yellow $p_y^w = [x_y^w\, y_y^w\, z_y^w]^T$ and red $p_r^w = [x_r^w\, y_r^w\, z_r^w]^T$. These 3 points are used to estimate the end effector center $\hat{p}_e^w = [\hat{x}_e^w\, \hat{y}_e^w\, \hat{z}_e^w]^T$ and the orientation $\hat{R}_e^w = Rot(z, \hat{\gamma}) \cdot Rot(y, \hat{\beta}) \cdot Rot(x, \hat{\alpha})$ represented in Euler angles. From the setup of the end effector and three colored balls, the estimated ball centers can be represented by

$$\hat{p}_g^w = \hat{R}_e^w \cdot \begin{bmatrix} d & 0 & 0 \end{bmatrix}^T + \hat{p}_e^w \qquad (4)$$

$$\hat{p}_y^w = \hat{R}_e^w \cdot \begin{bmatrix} 0 & d & 0 \end{bmatrix}^T + \hat{p}_e^w \qquad (5)$$

$$\hat{p}_r^w = \hat{R}_e^w \cdot \begin{bmatrix} -d & 0 & 0 \end{bmatrix}^T + \hat{p}_e^w \qquad (6)$$

To solve for the end effector pose, a cost function C with 6 unknown variables $(\hat{x}_e^w, \hat{y}_e^w, \hat{z}_e^w, \hat{\alpha}, \hat{\beta}, \hat{\gamma})$ can be set up. Minimizing the cost gives an optimal solution of the unknown variables,

$$C = (p_g^w - \hat{p}_g^w)^2 + (p_y^w - \hat{p}_y^w)^2 + (p_r^w - \hat{p}_r^w)^2 \qquad (7)$$

Take the middle point of p_g^w and p_r^w as the initial guess of $\hat{p}_e^w = [\hat{x}_e^w\ \hat{y}_e^w\ \hat{z}_e^w]^T$, denoted as P_{e0}^w. The initial guess of orientation is taken by setting $\overrightarrow{P_{e0}^w P_g^w}$, $\overrightarrow{P_{e0}^w P_y^w}$ as x and y axis of the end effectors frame. Then, the Nelder-Mead Simplex Method [22] is used to find the optimal value of $(\hat{x}_e^w, \hat{y}_e^w, \hat{z}_e^w, \hat{\alpha}, \hat{\beta}, \hat{\gamma})$, which is the pose of the end effector.

F. Neural Network Architecture

The training data for the neural network is represented as:

$$\chi_{NN} = \left\{ \left(\text{ravenstate}^{(k)}, \text{err}^{(k)} \right) \mid k = [1...N] \right\}, \qquad (8)$$

where 'ravenstate' is a ROS topic that contains real-time kinematics and dynamic information of RAVEN-II. And 'err' is the difference between the RAVEN-II reported end effector positions and the ground truth collected through vision-based measurement in phase one. More details follow:

1) The Features: The information provided in a 'ravenstate' message includes: (a) kinematics derived Cartesian pose, (b) desired Cartesian pose, (c) current joint pose, (d) desired joint pose, (e) motor and joint velocities, (f) motor torques, (g) desired grasper pose and other information, all of which are added as features of the neural network.

2) The Labels: We chose the end effector *position error* as the label instead of the ground truth position itself because RAVEN-II already has end effector pose feedback in the 'ravenstate' topic. To get a more accurate estimation, one only needs to correct RAVEN-II pose feedback derived from joint pose and forward kinematics, instead of creating a new estimation. In fact, this design is more robust to static Cartesian positional offsets contained in the 'ravenstate' reported position, which slightly differs every time the system restarts.

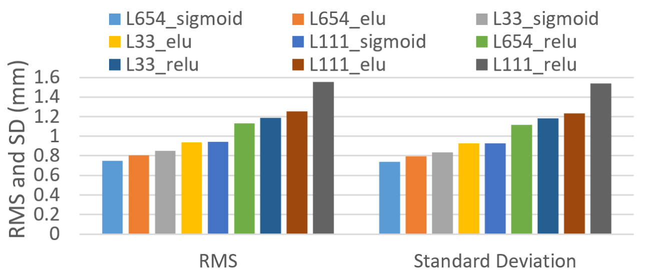

Fig. 7: The performance of different hyperparameter sets. The legend 'L654_sigmoid' means that there are 3 hidden layers of 600, 500 and 400 units with sigmoid activation. '_elu/_relu' refers to ELU [23] and RELU [24] activation functions

3) Network Structure and Parameters: In order to determine the structure and hyperparameters of the neural network, the network was first trained with randomly chosen hyperparameters from a large range and then the hyperparameters narrowed to a smaller range. We evaluated more than 100 sets of hyperparameter values. Nine illustrative values are plotted in Fig. 7. The following hyperparameter values yielded the best obtained performance:

- 3 dense sigmoid layers of 600, 500, 400 units.
- batch normalization [25], batch size = 1024.
- learning rate = 1×10^{-8}, epochs = 10000.
- regularization rate = $5 \times 10^{-6} (L_1)$.
- Adam [26] optimizer: $(\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8})$

III. Experimental Result

A. Experiment Setup

A standard RAVEN II surgical robot was used with a remote controller. Only the left arm was activated and there was no control signal sent to the right arm. The vision-based ground truth measurement system was set up and the environment was surrounded by black cloth to reduce background interference in images. To get data for neural network training, RAVEN-II was manually operated by a remote controller and moved randomly in the workspace for 140 minutes. Due to unstable performance of RAVEN-II near the robot workspace boundary and singular points, the operation workspace defined in this paper is a reasonable large space around initialization center, which is enough for a typical block transfer operation. The data was recorded as time-synchronized pairs of 'ravenstate' and ground truth end effector positions. The recorded trajectories contained a total of $49,407$ data pairs, in which each ravenstate consists of 118 floats for each arm.

$$\left(\text{ravenstate}^{(k)}, \text{err}^{(k)}\right) k = 1 ... 49,407 \, .$$

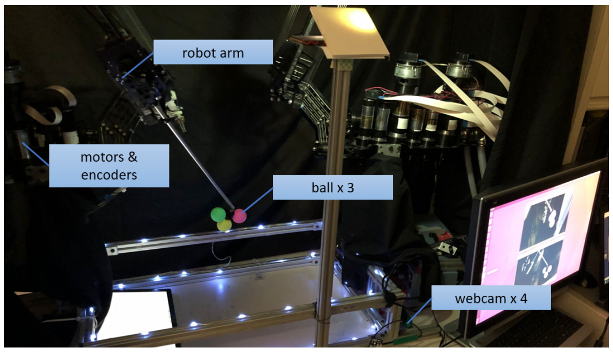

Fig. 8: Experimental setup.

B. Neural Network Offline Training

The training data χ_{NN} was used to train the neural network $f_{NN} : \mathbb{R}^{118} \rightarrow \mathbb{R}^3$ to map input 'ravenstate'$^{(k)}$ to the output end effector position error $err^{(k)}$. Suppose p_e^w and $\hat{p}_e^w$ are the true and 'ravenstate' reported RAVEN-II positions. After training, an online end effector position precision estimator was built, where $p_e^w = \hat{p}_e^w + err$.

The training data comes from recorded trajectories, which are a series of continuous points traversing most of the workspace. However, the neural network estimator is expected to work in the entire workspace. The technique of randomly choosing subsets of points in the recorded trajectory to form the validation set and test sets might only achieve high performance on the training set trajectories, and rapidly decreases its accuracy when the robot leaves the trained trajectories. After several tests, it was found that to prevent poor performance during online estimation at unseen points due to overfitting, the validation set and test set should be carefully selected. In particular, the validation set and test set are randomly chosen *trajectories* in the dataset, instead of random sample points. The purpose is to make sure that the trained neural network has similar accuracy in the whole workspace instead of performing well only along the recorded trajectories.

C. Performance Analysis

First, the accuracy of the measurement system was tested. The error of measurement mainly comes from two aspects: 1) inaccurate circle detection in the images; 2) inaccurate camera localization. The 1^{st} type of error was calculated by assuming camera localization is perfect. 200 frames were chosen, and the centers of the circles were manually marked in the images. Then, the ball centers were calculated using manually marked circle centers and the result was used as the ground truth for accuracy analysis. The 2^{nd} type of error was calculated by assuming circle detection was perfect. 16 marker points with known position inside the workspace were chosen and manually marked in the image plane. After ball centers were detected, the optimization method introduced in II.E was used to solve the end effector position and further improved the accuracy. Finally, the result was compared with the ground truth and analytically shown in TABLE I.

TABLE I: The Error of Measurement System

Axis	Single Ball Center		End Effector Position	
	RMS(mm)	SD(mm)	RMS(mm)	SD(mm)
x	0.6273	0.2891	0.4229	0.0513
y	0.6354	0.3000	0.2896	0.0949
z	0.3004	0.1029	0.1521	0.0718
3D	0.9866	0.2315	0.5346	0.0695

After training the neural network offline, an online neural network estimator was built, which took the 'ravenstate' as input and output the estimated end effector position, $e^w = \hat{p}_e^w + err$, which is roughly 10 times more accurate than the RAVEN-II original estimation calculated by kinematics. The test set, consisting of 10% of the recorded data, was untouched during all training procedures, and was used to test the system performance.

Fig. 9 shows the online estimation result. The actual value was obtained using the measurement system, and the original estimate came from the RAVEN-II forward kinematics, containing errors from cable-driven mechanism *etc*. Fig. 10 shows the RMS error and standard deviation of the error.

IV. Discussions and Future Work

A. Performance of Phase One

The vision-based ground truth position measurement system was used to measure the RAVEN-II end effector position, but can be extended to measuring other object poses. And the system is robust to occlusion with 4 cameras around workspace. The measurement system achieves accuracy below 0.5 mm in each axis. (TABLE.I) In summary, the proposed procedure for acquiring ground truth position is

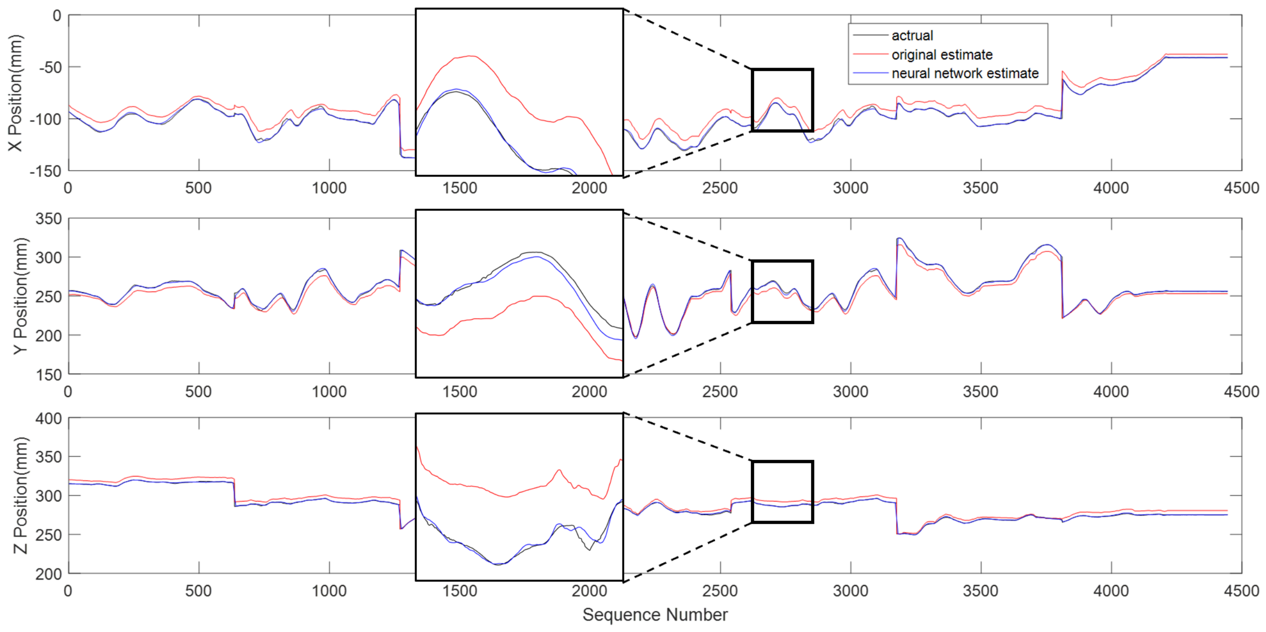

Fig. 9: Comparison of actual position, motor encoder based forward kinematics position, and neural network estimation. The neural network estimator decreases the positional RMS error by 83.6% and the standard deviation of error by 59.4%.

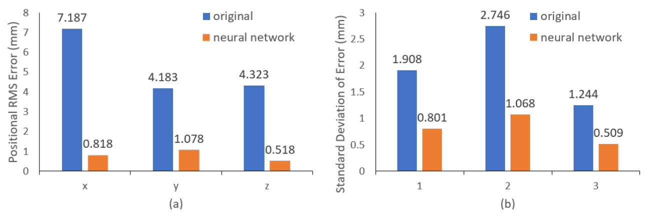

Fig. 10: RMS error and standard deviation of error in end effector position with and without neural network online estimator.

a cost-effective option to satisfy the required accuracy and can further improve precision by increasing the ball count or using higher resolution cameras.

As illustrated in Fig. 9, the data was collected through manual teleoperation of RAVEN-II for 140 minutes through random trajectories traversing the robot workspace. Compared to preprogrammed periodic trajectories, this dataset provides data spanning most of the workspace and the motion pattern is more realistic to teleoperated surgical operations.

B. Advantages of Phase Two

After offline training, we built an online estimator, which gave a significantly more accurate end effector position estimate. Since the neural network input 'ravenstate' is an existing ROS topic from RAVEN-II, the online estimator requires no additional sensor or information and can be applied to any RAVEN-II robot.

Fig. 10 shows that the neural network estimator can not only decrease the RMS error, but also the standard deviation of the error, which means that the online estimator can improve precision beyond applying a static Cartesian offset alone. Moreover, our estimator could potentially be utilized in other robots where transmission compliance and losses are significant influences on precision. Our current work assumes the dynamic property of the system does not change significantly after training, i.e., the testing happens within a short time after training and the end effector is also unloaded. In the future, the influence of payload and time period between training and testing will be further studied.

V. Conclusion

Due to compliance and losses in the transmission mechanism (cable/pulley links in the RAVEN-II), indirect measurement of joints and other external uncertainties, estimation of the precise end effector position is challenging. In this work, a cost-effective online RAVEN-II position precision estimator is implemented and tested on a 140-minute trajectory set. The system entails a vision-based ground truth position measurement system and an online data-driven position estimator based on a neural network. Although the total cost of the measurement is around one hundred dollars (mostly the cost of four webcams), the sub-millimeter accuracy achieved is more than 10 times better than the RAVEN-II position accuracy based on motor-mounted encoders. The neural network estimator decreases the positional RMS error by 83.6% and the standard deviation of error by 59.4%. Furthermore, the estimator requires no additional sensors or information other than the RAVEN-II built-in 'ravenstate' ROS topic (updated at 1000Hz), which contains kinematic and dynamic information of RAVEN-II. Finally, the proposed cost-effective position estimator can be generalized to other RAVEN-II sites, as well as other robots with accuracy affected by compliance and losses in transmission elements between motors and joints. Although robotic surgeons today readily compensate for imperfect position control, as commercial surgical robots incorporate human augmentation and autonomous functions, the need for accurate position estimate and control will increase.

References

[1] M. Haghighipanah, Y. Li, M. Miyasaka, and B. Hannaford, "Improving position precision of a servo-controlled elastic cable driven surgical robot using unscented kalman filter," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015, pp. 2030–2036.

[2] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. N. Kosari, and L. White, "Raven-ii: an open platform for surgical robotics research," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 4, pp. 954–959, 2012.

[3] J. H. Palep, "Robotic assisted minimally invasive surgery," *Journal of Minimal Access Surgery*, vol. 5, no. 1, p. 1, 2009.

[4] D. Hu, Y. Gong, B. Hannaford, and E. J. Seibel, "Semi-autonomous simulated brain tumor ablation with ravenii surgical robot using behavior tree," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3868–3875.

[5] B. Kehoe, G. Kahn, J. Mahler, J. Kim, A. Lee, A. Lee, K. Nakagawa, S. Patil, W. D. Boyd, P. Abbeel *et al.*, "Autonomous multilateral debridement with the raven surgical robot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1432–1439.

[6] C. N. Riviere, W. T. Ang, and P. K. Khosla, "Toward active tremor canceling in handheld microsurgical instruments," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 5, pp. 793–800, 2003.

[7] K. Lindgren, K. Huang, and B. Hannaford, "Towards real-time surface tracking and motion compensation integration for robotic surgery," in *2017 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 2017, pp. 450–456.

[8] S. G. Yuen, D. T. Kettler, P. M. Novotny, R. D. Plowes, and R. D. Howe, "Robotic motion compensation for beating heart intracardiac surgery," *The International journal of robotics research*, vol. 28, no. 10, pp. 1355–1372, 2009.

[9] Y. H. Su, K. Huang, and B. Hannaford, "Real-time vision-based surgical tool segmentation with robot kinematics prior," in *Medical Robotics (ISMR), 2018 International Symposium on*. IEEE, 2018, pp. 1–6.

[10] S. N. Kosari, S. Ramadurai, H. J. Chizeck, and B. Hannaford, "Control and tension estimation of a cable driven mechanism under different tensions," in *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2013, pp. V06AT07A077–V06AT07A077.

[11] B. Lin, Y. Sun, X. Qian, D. Goldgof, R. Gitlin, and Y. You, "Video-based 3d reconstruction, laparoscope localization and deformation recovery for abdominal minimally invasive surgery: a survey," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 12, no. 2, pp. 158–178, 2016.

[17] A. Fetić, D. Jurić, and D. Osmanković, "The procedure of a camera calibration using camera calibration toolbox for matlab," in *2012*

[12] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. Ieee, 2000, pp. 153–158.

[13] D. Seita, S. Krishnan, R. Fox, S. McKinley, J. Canny, and K. Goldberg, "Fast and reliable autonomous surgical debridement with cable-driven robots using a two-phase calibration procedure," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6651–6658.

[14] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci® surgical system," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 6434–6439.

[15] J. Mahler, S. Krishnan, M. Laskey, S. Sen, A. Murali, B. Kehoe, S. Patil, J. Wang, M. Franklin, P. Abbeel *et al.*, "Learning accurate kinematic control of cable-driven surgical robots using data cleaning and gaussian process regression," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2014, pp. 532–539.

[16] H. Yuen, J. Princen, J. Illingworth, and J. Kittler, "Comparative study of hough transform methods for circle finding," *Image and vision computing*, vol. 8, no. 1, pp. 71–77, 1990.
Proceedings of the 35th International Convention MIPRO. IEEE, 2012, pp. 1752–1757.

[18] J. J. Moré, "The levenberg-marquardt algorithm: implementation and theory," in *Numerical analysis*. Springer, 1978, pp. 105–116.

[19] J. Canny, "A computational approach to edge detection," in *Readings in computer vision*. Elsevier, 1987, pp. 184–203.

[20] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[21] H. King, S. Kosari, B. Hannaford, and J. Ma, "Kinematic analysis of the raven-ii research surgical robot platform," *University of Washington, Tech. Rep. UWEETR-2013-0006*, 2012.

[22] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the nelder–mead simplex method in low dimensions," *SIAM Journal on optimization*, vol. 9, no. 1, pp. 112–147, 1998.

[23] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[24] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.