# Learning to Calibrate - Estimating the Hand-eye Transformation without Calibration Objects

Krittin Pachtrachai, Francisco Vasconcelos, Philip Edwards, and Danail Stoyanov

***Abstract*—Hand-eye calibration is a method to determine the transformation linking between the robot and camera coordinate systems. Conventional calibration algorithms use a calibration grid to determine camera poses, corresponding to the robot poses, both of which are used in the main calibration procedure. Although such methods yield good calibration accuracy and are suitable for offline applications, they are not applicable in a dynamic environment such as robotic-assisted minimally invasive surgery (RMIS) because changes in the setup can be disruptive and time-consuming to the workflow as it requires yet another calibration procedure. In this paper, we propose a neural network-based hand-eye calibration method that does not require camera poses from a calibration grid but only uses the motion from surgical instruments in a camera frame and their corresponding robot poses as input to recover the hand-eye matrix. The advantages of using neural network are that the method is not limited by a single rigid transformation alignment and can learn dynamic changes correlated with kinematics and tool motion/interactions. Its loss function is derived from the original hand-eye transformation, the re-projection error and also the pose error in comparison to the remote centre of motion. The proposed method is validated with data from da Vinci Si and the results indicate that the designed network architecture can extract the relevant information and estimate the hand-eye matrix. Unlike the conventional hand-eye approaches, it does not require camera pose estimations which significantly simplifies the hand-eye problem in RMIS context as updating the hand-eye relationship can be done with a trained network and sequence of images. This introduces a potential of creating a hand-eye calibration approach that is capable of accurately updating the hand-eye matrix according to the changes in RMIS setup.**

***Index Terms*—Calibration and Identification; Deep Learning Methods; Computer Vision for Medical Robotics**

## I. INTRODUCTION

The use of miniaturised surgical tools and camera to provide access to an operative site has transformed surgical practice such that the incision and trauma are minimised. When combining with robotic setups, the tele-manipulation technique called RMIS is created, which provides accurate instrument articulation as well as enhancing ergonomics for the surgeon. The configuration also introduces a potential to increase safety by programmatically providing guidance in an operation through computer-assisted interventions such as virtual fixtures [1] to guide tools' trajectory or augmented reality to enhance visualisation and localisation of anatomical information [2]. However, as the camera is controlled by the robot, the implementation of such systems is highly dependent on determining the link between the camera coordinate frame and the kinematic of the robot, which is called hand-eye calibration [3].

Conventional hand-eye calibration methods require a calibration grid as the calibration target. Apart from differences in the problem formulations and optimisation algorithms on the original hand-eye equation, almost all of the conventional methods use camera calibration algorithm to refine camera poses with respect to the calibration target and forward kinematics to calculate the end-effector poses with respect to the robot base, as shown in Figure 1(a). While they yield accurate calibration results in many different robotics applications, the calibration procedure is more challenging in RMIS as one of the criteria to accurately calibrate the transfomation requires maximising rotational motion range [4], i.e. the changes in the camera orientation between any two selected frames must be as large as possible, to mitigate influence from noise in the system which is not easy to achieve. Moreover, any change in the setup needs yet another calibration procedure and it is not practical to reuse a calibration grid to recreate the scene for the calibration which could cause a disruption in the surgical workflow. Although it is arguable that due to the robot design, changes can be made insignificant and it is not necessary to re-calibrate the link, the experiments presented in the literature indicate that even a very small error can be translated to an observable discrepancy in poses in a chain of transformations [5], [6].

In this paper, we propose a neural network-based hand-eye calibration approach to recover the hand-eye matrix from a sequence of images without using a calibration grid. The method uses LSTM to detect temporal information in the features extracted from CNN [7] and poses determined by forward kinematics. The calibrated transformation is validated with the data from da Vinci Si and shows promising results. The advantages and contributions of this approach are listed as follows.

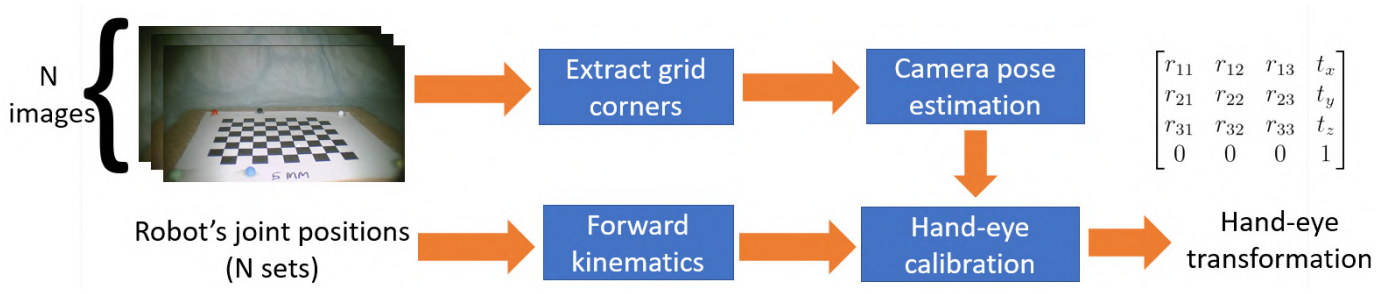
- The method and the loss function are not derived from the original hand-eye equation. Therefore, the solution does not suffer from the criteria requiring to achieve the

Manuscript received: February, 24, 2021; Revised May, 23, 2021; Accepted June, 22, 2021.

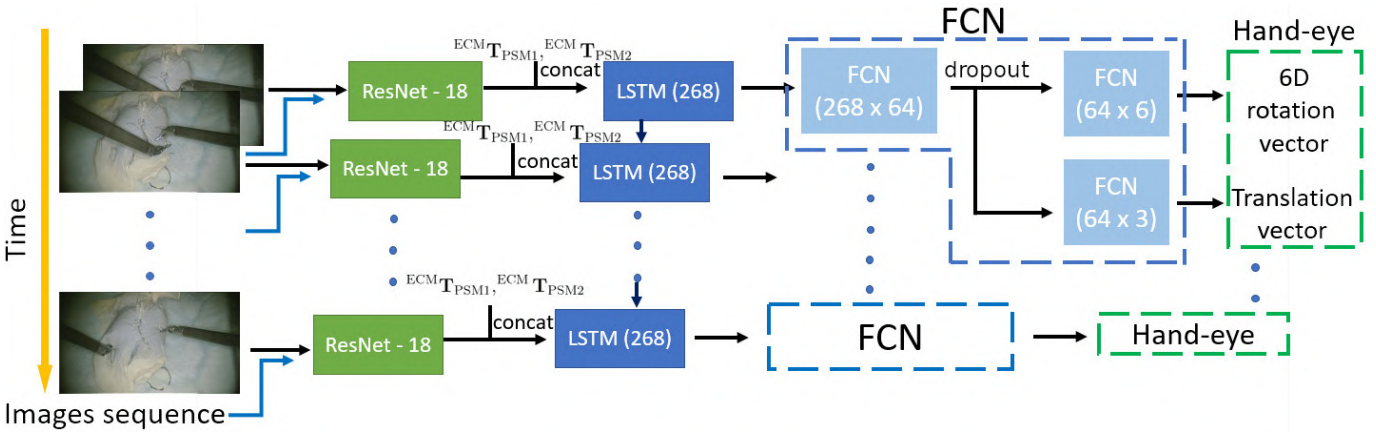
This paper was recommended for publication by Editor Lucia Pallottino upon evaluation of the Associate Editor and Reviewers' comments. The work was supported by the Wellcome/EPSRC Centre for Interventional and Surgical Sciences (WEISS) [203145Z/16/Z], Engineering and Physical Sciences Research Council (EPSRC) [EP/P027938/1, EP/R004080/1, EP/P012841/1], and The Royal Academy of Engineering Chair in Emerging Technologies scheme.

Krittin Pachtrachai, Francisco Vasconcelos, Philip Edwards and Danail Stoyanov are with the Wellcome / EPSRC Centre for Interventional and Surgical Sciences (WEISS) and the Department of Computer Science, University College London, Gower Street, WC1E 6BT, UK. {krittin.pachtrachai.13; f.vasconcelos; eddie.edwards; danail.stoyanov}@ucl.ac.uk

Digital Object Identifier (DOI): see top of this page.




(a)




(b)

Fig. 1. (a) The schematic shows the conventional procedure to calibrate the hand-eye matrix. The main variation in most of hand-eye calibration methods is at the last step where the problem is formulated and optimised differently, while all the steps before remain unchanged. The images are selected based on the criteria listed in [4] to maximise the calibration accuracy and the robot's joint positions must be corresponded to the selected images. In some application, robot joints' positions and forward kinematics are replaced by an external tracking system, providing poses with respect to its reference frame, which essentially is the same as using a robot and forward kinematics. (b) The proposed neural network architecture to solve the hand-eye problem in RMIS using Recurrent Neural Network. The inputs to the network are a sequence of images and their corresponding end-effector poses of the tool arms (PSM1 and PSM2) with respect to the end-effector pose of the camera arm (ECM), ${}^{\text{ECM}}\mathbf{T}_{\text{PSM1}}$ and ${}^{\text{ECM}}\mathbf{T}_{\text{PSM2}}$. The outputs are a translation vector and a 6-D rotation vector. The rotation vector is converted to a rotation matrix using Gram-Schmidt orthogonalisation and combined with the translation part to form the hand-eye matrix.

maximum calibration accuracy.

- It does not need a calibration grid as the network directly estimates the hand-eye matrix from the changes in the camera view with respect to the kinematic. Therefore, it does not disrupt the surgical workflow and greatly simplifies the calibration procedure in RMIS.
- It is a neural network-based calibration approach which deals with dynamic sources of calibration errors by avoiding the explicit use of a fixed rigid tranformation. This is useful to deal with both changes in calibration (from passive joints), and more complex and dynamic kinematic errors (tendon driven actuators and forces on the end-effector).
- The method uses CNN to process images and RNN to deal with the extracted features, kinematic and temporal information without having to solve camera poses estimation problem.
- The designed loss function combines the difference between the pre-calibrated hand-eye matrix and the predicted transformation, the re-projection error, and the orientation defined the remote centre of motion (RCM), so the recovered hand-eye matrix is close to the original hand-eye matrix and also adaptive to changes in the camera arm.

*Notation:* Matrices are represented by a bold capital letter, e.g. $\mathbf{K}$. Bold upper-case letters represent a point or a vector in 3D space in relation to the frame in the subscription, e.g. $\mathbf{P}_{\text{world}}$ represents a point $\mathbf{P}$ in the world frame and a bold lower-case letter represents a point in images corresponding to its upper-case letter, e.g. $\mathbf{p}$ is $\mathbf{P}_{\text{world}}$ in an image. The transformation from frame $i$ to frame $j$ is denoted by the $4 \times 4$ matrix ${}^{j}\mathbf{T}_{i}$, which consists of the rotation matrix ${}^{j}\mathbf{R}_{i}$ and the translation vector ${}^{j}\vec{t}_{i}$. The following notations are also used throughout the paper to denote the frames on the da Vinci robot, $F_0$ denotes the base frame of the robot, Endoscope Control Manipulator (ECM), Patient-Side Manipulator 1 (PSM1), Patient-Side Manipulator 2 (PSM2) are denoted in Figure 2, and CAM is used to denote the pose of the camera frame.

## II. Related Work

The solution to the hand-eye problem in robotic systems was first introduced in [8], and it is the answer to the homogeneous matrix equation written as follows,

$$\mathbf{AX} = \mathbf{XB} \tag{1}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the relative transformations in different coordinate frames and $\mathbf{X}$ is the hand-eye transformation. The

main requirement to solving the problem, that is at least two motions with different rotation axes are required [3], and the characterisation of camera-robot motions for maximising the calibration accuracy detailed in [4] can be easily achieved with the present robotic systems.

The solutions with different parameterisations to Eq. 1 have been proposed [6], [8], [9]. The constraint such as epipolar constraint [10], re-projection error [11] are also developed to further optimise the problem in addition to Eq. 1. However, as these solutions are developed from the original equations, they suffer with the same degeneracy and ill-posed configurations. There is also another hand-eye formulation ($\mathbf{AX} = \mathbf{YB}$) [12]–[14] which is applicable to the RMIS setup, but [12] shows that the formulation also has similar characteristics to Eq. 1 and it is also later shown by [15] that when the motion range is smaller, the solution degenerates at a faster rate than the original equation.

A few works in the literature propose the hand-eye solution without a calibration grid by using Structure-from-Motion (SfM) [16], or an EndoWrist surgical tool as a calibration target instead of a static calibration grid [17]–[19]. While removing the calibration target simplifies the procedure, the SfM approach is still not suitable to RMIS as it has a very high computational cost so updating the transformation can be challenging. On the other hand, the methods using a surgical tool as a calibration target can introduce an additional source of calibration error when solving Eq. 1 as it still heavily relies on accurate tracking results. However, it is undeniable that hand-eye calibration requires camera pose estimation from images sequence.

Despite successes in solving computer vision problems [20] using deep neural networks such as classification [21], [22], localisation [23]–[25], sensors calibration [26], [27], and segmentation [28], [29], there are only a few papers proposing automatic hand-eye calibration method utilising deep neural networks [30]. The paper proposes a network only using hidden layers to process robot poses to solve both inverse kinematics and the hand-eye problem at the same time, but the method does not consider the continuity of data or use CNN to process images. On the other hand, our method (see Figure 1(b)) fully utilises CNN and RNN to deal with both images and temporal information to solve the problem without extrinsically solving for camera poses. Furthermore, it does not rely on the original hand-eye equation, therefore a constrained motion around the RCM in RMIS that causes an ill-posed in Eq. 1, does not worsen the calibration performance as it is shown later in Section IV.

## III. Methods

In most surgical robots, for example, the da Vinci system (see Figure 2), every arm is connected at the base frame $F_0$, e.g. the kinematic chain defining the end-effector poses are connected and the transformations between ECM, PSM1 and PSM2 are known. The transformations are calculated from

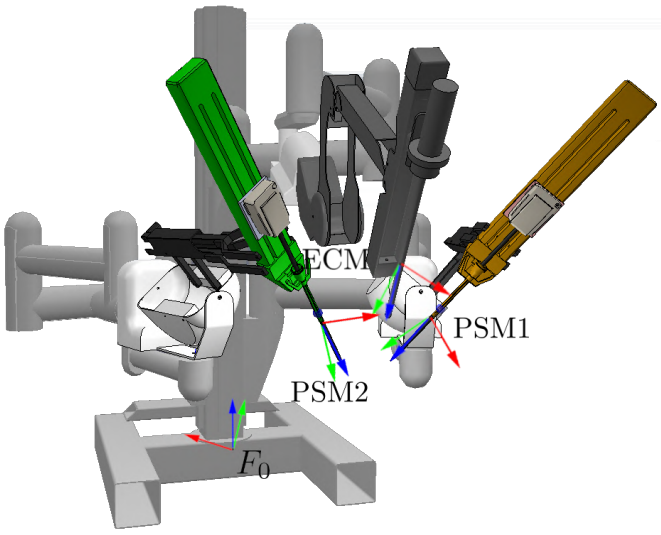



Fig. 2. A version of da Vinci Surgical Robot in a simulation platform, CoppeliaSim [31]. The model has two PSM arms and one ECM arm, and every arm is connected to the robot base $F_0$.

forward kinematics,

$$ {}^{F_0}\mathbf{T}_{\text{PSM1}} = {}^{F_0}\mathbf{T}_{\text{R1}}\ {}^{\text{R1}}\mathbf{T}_{\text{R2}} \cdots {}^{\text{R14}}\mathbf{T}_{\text{PSM1}} \tag{2} $$

$$ {}^{F_0}\mathbf{T}_{\text{PSM2}} = {}^{F_0}\mathbf{T}_{\text{L1}}\ {}^{\text{L1}}\mathbf{T}_{\text{L2}} \cdots {}^{\text{L14}}\mathbf{T}_{\text{PSM2}} \tag{3} $$

$$ {}^{F_0}\mathbf{T}_{\text{ECM}} = {}^{F_0}\mathbf{T}_{\text{C1}}\ {}^{\text{C1}}\mathbf{T}_{\text{C2}} \cdots {}^{\text{C14}}\mathbf{T}_{\text{ECM}} \tag{4} $$

where $\text{R}i, \text{L}i$ and $\text{C}i$ represent the frames in between the base and the respective end-effectors. The hand-eye matrix ${}^{\text{CAM}}\mathbf{T}_{\text{ECM}}$ is defined as a transformation in addition to the ECM frame to represent the pose of the camera with respect to ECM frame.

$$ {}^{F_0}\mathbf{T}_{\text{CAM}} = {}^{F_0}\mathbf{T}_{\text{ECM}}({}^{\text{CAM}}\mathbf{T}_{\text{ECM}})^{-1} \tag{5} $$

Although the end-effector frame ECM is defined at the tip of the scope, it is not necessarily the same frame as the camera principal point and the purpose of this network is to find this transformation. This section shows the theory behind the proposed algorithm, data preprocessing, the network architecture and loss functions.

### A. Data preprocessing

Although the method does not require extracting grid corners and camera poses estimation as images are directly fed into the network, it still needs forward kinematics to determine corresponding poses to each image. Forward kinematics is the method to translate from robot's joints positions to poses assigned at several parts of a robot and the most common approach is to use Denavit-Hartenberg convention [32], [33].

The DH parameters for computing the transformations ${}^{F_0}\mathbf{T}_{\text{ECM}}, {}^{F_0}\mathbf{T}_{\text{PSM1}}$ and ${}^{F_0}\mathbf{T}_{\text{PSM2}}$ are taken from [34] and modified to match the da Vinci model. Once the transformations are calculated, the transformation from ECM to PSM1 and PSM2 can simply be determined as follows.

$$ {}^{\text{ECM}}\mathbf{T}_{\text{PSM1}} = ({}^{F_0}\mathbf{T}_{\text{ECM}})^{-1}\ {}^{F_0}\mathbf{T}_{\text{PSM1}} \tag{6} $$

$$ {}^{\text{ECM}}\mathbf{T}_{\text{PSM2}} = ({}^{F_0}\mathbf{T}_{\text{ECM}})^{-1}\ {}^{F_0}\mathbf{T}_{\text{PSM2}} \tag{7} $$

Apart from the above transformations, the network requires an initial estimation of camera intrinsic parameters (also distortion parameters), and the hand-eye matrix obtained from hand-eye calibration using a calibration grid [6] as these will be used to calculate loss for training the network. A number of frames that clearly visualising both of the PSMs' end-effector are also selected and the end-effector positions in the frames are manually labelled.

### B. Network architecture

The proposed network consists of a pre-trained ResNet-18 network [7] to extract features in each frame, an LSTM [35] and fully-connected layers as shown in Figure 1(b). The output of ResNet-18 is concatenated with the the transformations calculated from Eq. 6 and 7 and fed into the LSTM layer to extract temporal features between consecutive frames. Finally, fully-connected layers are used to compute the rotation and the translation components of the predicted hand-eye matrix. The dropout is used before estimating the translation and the rotation component to prevent an overfitting. The translation is a $3 \times 1$ vector ${}^{\text{CAM}}\vec{t}_{\text{ECM}}$ and can be used in the hand-eye matrix directly, but the rotation vector is of 6-D format, $[\vec{a}_1^T, \vec{a}_2^T]^T$, which needs to be converted using Gram-Schmidt orthogonalisation into three column vectors as follows.

$$\begin{aligned} {}^{\text{CAM}}\mathbf{R}_{\text{ECM}} &= \begin{bmatrix} \vec{r}_1 & \vec{r}_2 & \vec{r}_3 \end{bmatrix} \\ \vec{r}_1 &= \frac{\vec{a}_1}{||\vec{a}_1||} \\ \vec{r}_2 &= \frac{\vec{a}_2 - (\vec{a}_1^T\vec{a}_2)\vec{a}_1}{||\vec{a}_2 - (\vec{a}_1^T\vec{a}_2)\vec{a}_1||} \\ \vec{r}_3 &= \vec{r}_1 \times \vec{r}_2 \end{aligned} \tag{8}$$

where $\vec{r}_1, \vec{r}_2$ and $\vec{r}_3$ are column vectors that satisfy orthogonal constraints in a rotation matrix. This representation does not suffer from discontinuity like quaternion, or Euler angles and it is empirically shown that it yields a more accurate pose estimation [36]. Combining with the translation vector yields the hand-eye matrix defined as,

$${}^{\text{CAM}}\mathbf{T}_{\text{ECM}} = \begin{bmatrix} {}^{\text{CAM}}\mathbf{R}_{\text{ECM}} & {}^{\text{CAM}}\vec{t}_{\text{ECM}} \\ \vec{0}^T & 1 \end{bmatrix} \tag{9}$$

### C. Loss function

*1) Hand-eye loss:* The estimated hand-eye matrix is compared with the initial hand-eye calibration using a calibration grid, ${}^{\text{CAM, grid}}\mathbf{T}_{\text{ECM}}$. This is obtained beforehand at the data pre-processing step. Given that the scope is not changed, the hand-eye matrix linking the robot and camera coordinate systems should be close to the original estimated by the conventional method [34]. The loss is computed from the error between the two transformations in its Lie algebra domain [37].

$$\mathfrak{L}_{\text{HE}} = ||\text{logm}(({}^{\text{CAM, grid}}\mathbf{T}_{\text{ECM}})({}^{\text{CAM}}\mathbf{T}_{\text{ECM}})^{-1})|| \tag{10}$$

where $\text{logm}(\cdot)$ represents a matrix logarithm and the matrix ${}^{\text{CAM, grid}}\mathbf{T}_{\text{ECM}}$ is the hand-eye matrix calculated from any conventional calibration method.

Fig. 3. The 0 degree scope used in the experiment. The kinematic chain defined in Eq. 4 assigns the final frame at the centre of the tip of the scope (ECM frame), which is not the same as the camera position and orientation. The CAM frame is typically defined at somewhere in front of the scope, and hand-eye calibration is the process to identify the exact pose of the CAM frame.

*2) Re-projection loss:* As the frames assigned on a PSM's shaft are defined in [34] and they have been labelled, we can use this to evaluate re-projection loss. The matrices in Eq. 6 and 7 can be transformed to the camera frame by simply pre-multiplying ${}^{\text{CAM}}\mathbf{T}_{\text{ECM}}$ to the terms. Let $\mathbf{p}_{1,i}$ and $\mathbf{p}_{2,i}$ be the labelled point on PSM1 and PSM2 in the frame $i$, respectively. The re-projection loss can be written as

$$\begin{aligned} \mathfrak{L}_{\text{PROJ}} = \sum_{i=1}^{N} &||\mathbf{p}_{1,i} - f({}^{\text{CAM}}\mathbf{T}_{\text{ECM}}\ {}^{\text{ECM}}\mathbf{T}_{\text{PSM1}}, \mathbf{P}_{PSM1})|| \\ &+ ||\mathbf{p}_{2,i} - f({}^{\text{CAM}}\mathbf{T}_{\text{ECM}}\ {}^{\text{ECM}}\mathbf{T}_{\text{PSM2}}, \mathbf{P}_{PSM2})|| \end{aligned} \tag{11}$$

where $f(\cdot)$ denotes the re-projection function including both intrinsic parameters and the distortion parameters and $N$ is the number of labelled frames.

*3) RCM loss:* As suggested in [38], one of the axis assigned at the camera frame is assumed to be parallel with one of the axis at ECM frame. The paper shows that the aligned axis assumption is sufficiently accurate for relative motion estimation. As the position of the RCM is by the kinematic, this loss is adaptive to the changes in the kinematic. The loss calculated from the RCM can be written as follows.

$$\mathfrak{L}_{\text{RCM}} = ||\arccos(\vec{r}_3^T\vec{z})|| \tag{12}$$

where $\vec{z}$ is the third column vector in the transformation ${}^{F_0}\mathbf{T}_{RCM}$ and $\vec{r}_3$ is defined in Eq. 8. Using this loss constrains the degree-of-freedom of the problem.

The final loss function consists of a weighted summation of the defined losses:

$$\mathfrak{L}_{\text{final}} = c_1\mathfrak{L}_{\text{HE}} + c_2\mathfrak{L}_{\text{PROJ}} + c_3\mathfrak{L}_{\text{RCM}} \tag{13}$$

where $c_i$ define the weight for each loss. In this paper, the weights are empirically set as 1, 0.1 and 0.1 respectively.

## IV. EXPERIMENTS AND RESULTS

We tested the proposed method with the data collected from the da Vinci system to see how the network behaves in terms of training and predicting the hand-eye matrix. The main criteria when testing hand-eye calibration algorithm are the number of included motions in the calibration, motion range to see

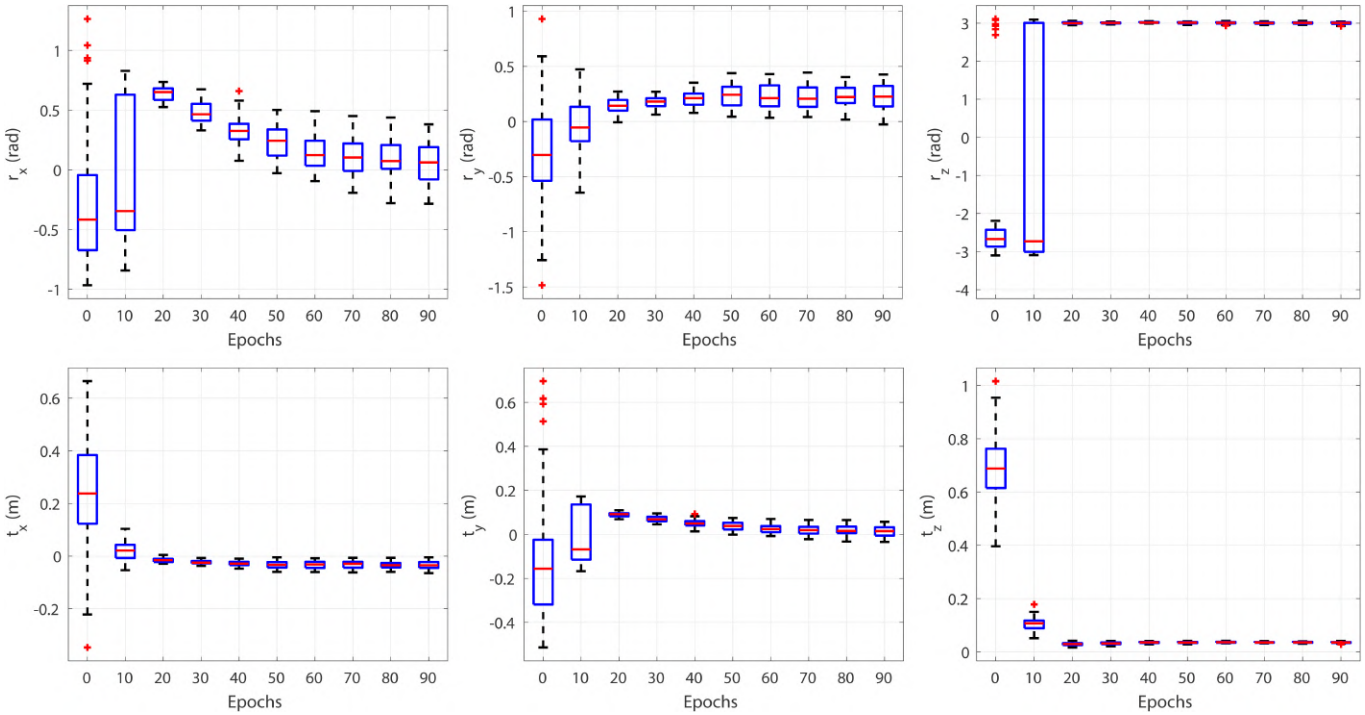



Fig. 4. The distribution of the elements in the hand-eye matrix after training for several epochs. The rotation matrix is converted from a rotation matrix using Rodrigues' formula to a rotation vector $[r_x, r_y, r_z]^T$ and the translation vector is directly extracted from the transformation. After training for several epochs, the estimation becomes more stable.

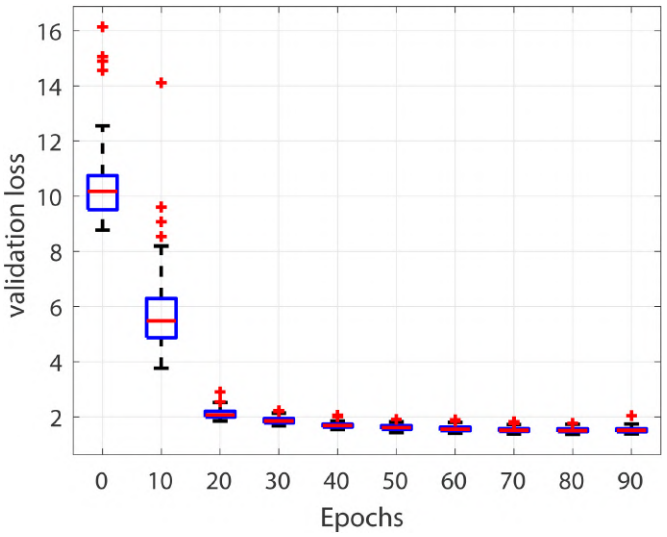



Fig. 5. Validation loss after training for several epochs, calculated from Eq. 13. The loss starts to become stable after 20 epochs.

how well an algorithm optimises when the given poses are ill-posed, and increasing input noise to check its robustness. These testing criteria do not apply to this paper because the proposed method directly uses images instead of geometric features that can be easily simulated.

### A. Training and testing

*1) Data pre-processing and collecting dataset:* Before starting the training, the network requires the hand-eye matrix obtained from any conventional calibration method for Eq. 10. In this paper, the method presented in [6] was used. The camera used in the procedure (see Figure 3) must be calibrated as the loss function utilises its intrinsic and distortion parameters. Furthermore, we select 100 images of PSM1 and PSM2 which clearly visualise their end-effector to manually label the positions as noted in [34]. The labelled images are shown in Figure 6 with the re-projected end-effectors using ${}^{\text{CAM, grid}}\mathbf{T}_{\text{ECM}}$ and ${}^{\text{CAM}}\mathbf{T}_{\text{ECM}}$.

For the network input, the synchronised joint data and images are collected when PSM1, PSM2 and ECM are moving around (this dataset is different from those 100 images we manually label), while the end-effector of PSM1 and PSM2 remain in the scene. The joint data is then converted to poses using Eq. 2 - 7. The size of the training data and validation set are 6,760 and 1,742 frames, together with their corresponding transformations, respectively. The size of the original images is 1920×1080 pixels, but an interpolation is applied to resize them to 640×360. An example of images input into the network are shown in Figure 6 with the re-projected PSM1's and PSM2's end-effectors.

In addition to sample frames shown in Figure 6, we also test the network with robotic prostatectomy video dataset which contains more scenes with occlusion, blurry and unclear features to test the stability of the proposed network. The size of training set is 15,002 frames. Figure 7 shows sample frames from tested video. The number of labelled images to be evaluated by re-projection error is also 100 images.

*2) Hyperparameters:* The network is trained using Adam Optimiser [39] with an initial learning rate of $1e^{-4}$. The

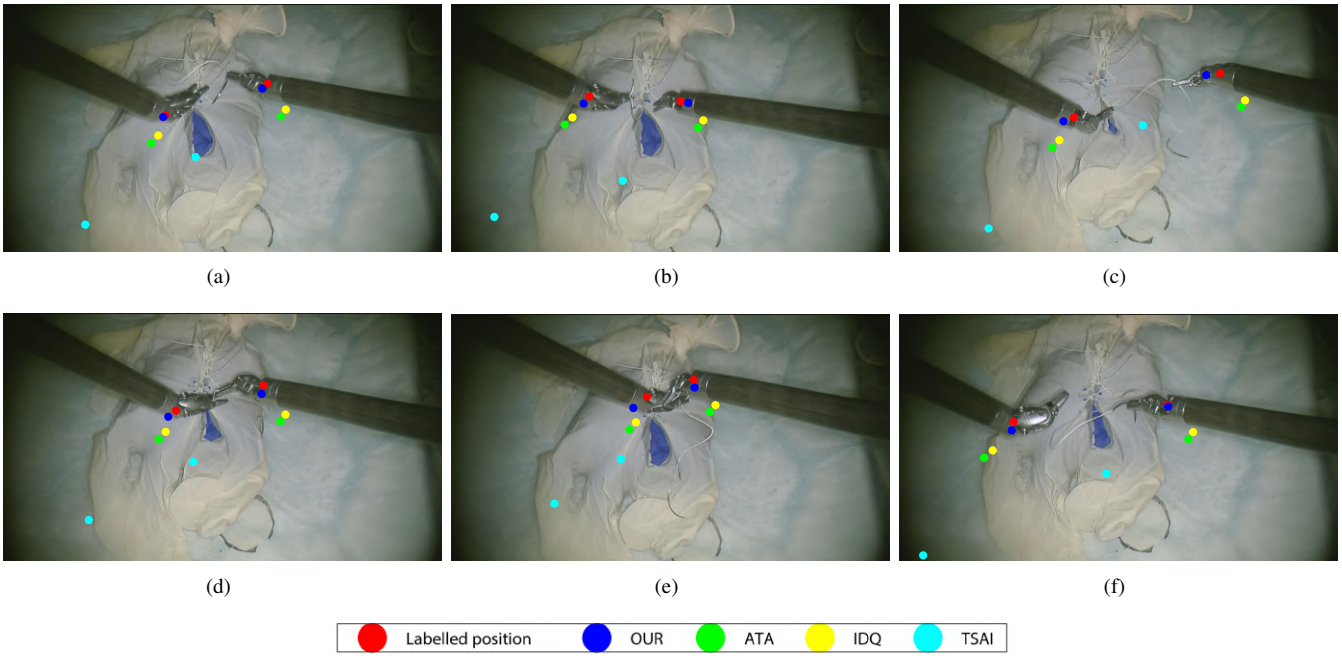



Fig. 6. Examples of re-projected PSM1's and PSM2's end-effector positions in the unseen test data. The red dots show the labelled position, and blue dots are the re-projected position using the calibration result from the proposed method (using all of the loss equations), ${}^{\text{CAM}}\mathbf{T}_{\text{ECM}}$. Green, yellow and cyan dots are the positions re-projected from using the transformation ${}^{\text{CAM, grid}}\mathbf{T}_{\text{ECM}}$ determined by different algorithms [3], [5], [6], respectively. The re-projected points using partial loss functions are not shown here as it is evident in Table I that training with all the loss functions yield a better calibration accuracy.

learning rate is decreased by 0.5 every 10 epochs. The size of each layer is shown in Figure 1(b). The training runs for 100 epochs to see the stability and feasibility of the neural network approach in solving the hand-eye problem.

### B. Results

As the ground truth of the hand-eye transformation is not known, we determine if it is possible to solve the problem using the proposed method from checking the stability of the calibration result and validation loss. For the calibration performance aspect, since we can determine the poses of PSM1's and PSM2's end-effectors with respect to the camera frame by calculating ${}^{\text{CAM}}\mathbf{T}_{\text{ECM}}\ {}^{\text{ECM}}\mathbf{T}_{\text{PSM1}}$ and ${}^{\text{CAM}}\mathbf{T}_{\text{ECM}}\ {}^{\text{ECM}}\mathbf{T}_{\text{PSM2}}$, we can compare the re-projected points on 100 unseen labelled images calculated by the proposed method and different algorithms.

Figure 4 and 5 show the distribution of hand-eye parameters and the validation loss. After several epochs, we can see that the calibration and the loss become stable which implies that the hand-eye problem in RMIS can be generalised using neural network and the solution does not overfit to the training data or the labelled images. Unlike the conventional hand-eye approaches where the rotation component is more stable than the translation component as illustrated in Figure 4, $r_x$ and $r_y$ estimated by our method are fluctuating despite the convergence of the other parameters. One of the reasons can be that the constraints on the rotation parameters in Eq. 13 are not as well-defined as the conventional method. However, it is shown in re-projection results that even with this uncertainty, the proposed method can still map the targeted points more accurately than the conventional methods.

TABLE I
AVERAGE ($\pm$ STANDARD DEVIATION) RE-PROJECTION ERROR ACROSS 100 LABELLED IMAGES, CALCULATED BY EUCLIDEAN DISTANCE BETWEEN TWO POINTS. 6 OF THEM ARE SHOWN IN FIGURE 6 (PIXEL). "OUR" REPRESENTS THE PROPOSED METHOD THAT USES DIFFERENT LOSS FUNCTION.

| Methods\Point | PSM1 | PSM2 |
|---|---|---|
| OUR (Eq. 10) | 162.29 $\pm$ 39.32 | 169.36 $\pm$ 59.34 |
| OUR (Eq. 10 and 11) | 102.67 $\pm$ 43.79 | 64.67 $\pm$ 62.95 |
| OUR (Eq. 10 and 12) | 163.08 $\pm$ 21.31 | 149.80 $\pm$ 15.19 |
| OUR (Eq. 13) | 46.83 $\pm$ 18.90 | 31.26 $\pm$ 17.20 |
| ATA [6] | 171.63 $\pm$ 21.32 | 155.44 $\pm$ 15.24 |
| IDQ [5] | 125.34 $\pm$ 20.79 | 141.31 $\pm$ 15.60 |
| TSAI [3] | 643.58 $\pm$ 42.74 | 432.31 $\pm$ 24.41 |

Figure 6 shows re-projection results on example images using different hand-eye algorithms and Table I shows average re-projection errors in pixel across all 100 unseen labelled images. The re-projected points using our method with Eq. 13 are much closer to the ground truth and the errors are the smallest.

When the network is trained and tested with a more challenging dataset (see Figure 7, the re-projection errors grow significantly and the re-projected points are not as close to the ground truth as they are in Figure 6. One possible explanation is that there are occlusions, specularity (Figure 7(b)) and blurry features in the dataset which LSTM may fail to detect, since we do not provide any priori knowledge on the occlusions or when some object is out of focus. Furthermore, the only input motions are the kinematic of ECM, PSM1 and PSM2, but the motions of features in the dynamic environment in the dataset are left unaccounted for. These could lead to an erroneous feature detection and pose estimation and result in a

TABLE II
AVERAGE ($\pm$ STANDARD DEVIATION) RE-PROJECTION ERROR ACROSS 100 LABELLED IMAGES, CALCULATED BY EUCLIDEAN DISTANCE BETWEEN TWO POINTS. 3 OF THEM ARE SHOWN IN FIGURE 7 (PIXEL).

| Methods\Point | PSM1 | PSM2 |
|---|---|---|
| OUR (Eq. 13) | 143.01 $\pm$ 59.40 | 128.49 $\pm$ 50.91 |
| ATA [6] | 217.14 $\pm$ 62.35 | 272.82 $\pm$ 59.92 |
| IDQ [5] | 249.07 $\pm$ 88.59 | 250.13 $\pm$ 109.41 |
| TSAI [3] | 608.65 $\pm$ 84.40 | 664.79 $\pm$ 119.14 |

miscalibration. However, the re-projected PSM1 and PSM2 are still somewhat close to the ground truth which signifies that the network can detect PSM1's and PSM2's motions, even though there are unaccounted changes in the environment.

### C. Discussion

The results show that the neural network approach with appropriate objective function $\mathfrak{L}$ can estimate the hand-eye matrix from the motions in images and corresponding poses. This is unlike the conventional algorithms that require a known-dimension calibration target (Figure 1(a)) and formulate the problem solely based relative motions in various mathematical domain [3], [5], [6] which yield different calibration accuracy. However, in addition to the criteria on the type of camera and robot motions to yield an accurate calibration accuracy for this formulation is already difficult to achieve in RMIS environment, there are other possible sources of errors such as the miscalibration of the camera parameters, or uncalibrated robot kinematic are not considered in the formulation. It is arguable that these errors are not significant but it is shown in [6] that a small inaccuracy in the chain of transformations can produce a noticeable calibration error. Therefore, the conventional calibration algorithms may not be the right direction for solving the hand-eye problem in RMIS. On the other hand, the proposed method can generalise the hand-eye problem and it uses both kinematic and imaging information in the loss function, therefore resulting in a more accurate calibration.

However, as the neural network is a data-driven approach, it requires a tremendous amount of data that is diverse enough for the network to generalise. If the input data is outside the training range or the data does not cover the variety of the input data, it is likely that the network will yield the wrong output. This is one of the limitations of the proposed method. For example, there are specularities and dynamic environment unaccounted for in Figure 7(b). There is not enough data to train the network to support these features, and yields a less accurate hand-eye matrix as shown in the images. Therefore, to prevent this miscalibration, the trained dataset must contain all special cases and higher variety of background appearances.

## V. CONCLUSION

We propose a novel deep convolutional network to automatically solve the hand-eye problem in RMIS. It estimates the hand-eye transformation from temporal information between frames and kinematic data. Unlike conventional hand-eye algorithms, the proposed algorithm does not require a calibration grid in the process, as it uses CNN to directly extract features and this replaces the camera pose estimation process. This enables an online update of the hand-eye matrix when there is a change in the system and therefore greatly simplifies the whole calibration procedure. Moreover, the algorithm does not suffer from the ill-posed cases occuring in RMIS as it does in the conventional approaches. The network is validated with the data from da Vinci Si and it shows that the proposed method can solve the hand-eye problem and the re-projected PSM1 and PSM2 are also more accurate than the ones estimated from the conventional hand-eye. This implies that the inaccuracy that is not accounted for in the original hand-eye equation in RMIS such as the miscalibration of the intrinsic parameters, the erroneous nominal robot kinematic, or the camera pose estimation are propagated to the final calibration result. However, as our method does not rely on the original equation and it is a neural network-based approach, it does not require the criteria from the original equation and can generalise the optimisation to mitigate the influence from these discrepancies and therefore yield a more accurate calibration result.

In the future, as it is shown that the hand-eye problem can be solved using a neural network-based approach, a more adaptive-to-robot-kinematic loss function will be investigated so that the model less depends on the original equation and more on the changes in the kinematic model and camera. Furthermore, in order to improve the generalising capability of the network, the network must be trained with more data with various background and different lighting conditions.

## ACKNOWLEDGEMENT

We would like to thank Intuitive Surgical, CA for providing us the da Vinci API and the support. The work was supported by the Wellcome/EPSRC Centre for Interventional and Surgical Sciences (WEISS) [203145Z/16/Z]; Engineering and Physical Sciences Research Council (EPSRC) [EP/P027938/1, EP/R004080/1, EP/P012841/1]; The Royal Academy of Engineering Chair in Emerging Technologies scheme.

## REFERENCES

[1] S. A. Bowyer, B. L. Davies, and F. R. y Baena, "Active constraints/virtual fixtures: A survey," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 138–157, Feb 2014.

[2] F. O. Matu, *et al.*, "Stereoscopic augmented reality system for supervised training on minimal invasive surgery robots," in *Proc. 2014 Virt. Real. Int. Conf. (VRIC'2014)*, 2014, pp. 33:1–33:4.

[3] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *IEEE Tran. Robot. Autom.*, vol. 5, no. 3, pp. 345–358, Jun 1989.

[4] J. Schmidt and H. Niemann, "Data selection for hand-eye calibration: A vector quantization approach," *Int. J. Robot. Res. (IJRR)*, vol. 27, no. 9, pp. 1027–1053, 2008.

[5] A. Malti and J. P. Barreto, "Robust hand-eye calibration for computer aided medical endoscopy," in *2010 IEEE Int. Conf. Robot. Autom. (ICRA'2010)*, May 2010, pp. 5543–5549.

[6] K. Pachtrachai, *et al.*, "Adjoint transformation algorithm for hand–eye calibration with applications in robotic assisted surgery," *Ann. Biomed. Eng.*, vol. 46, no. 10, pp. 1606–1620, Jul 2018.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[8] R. Y. Tsai and R. K. Lenz, "Real time versatile robotics hand/eye calibration using 3d machine vision," in *Proc. 1988 IEEE Int. Conf. Robot. Autom. (ICRA'1988)*, Apr 1988, pp. 554–561.

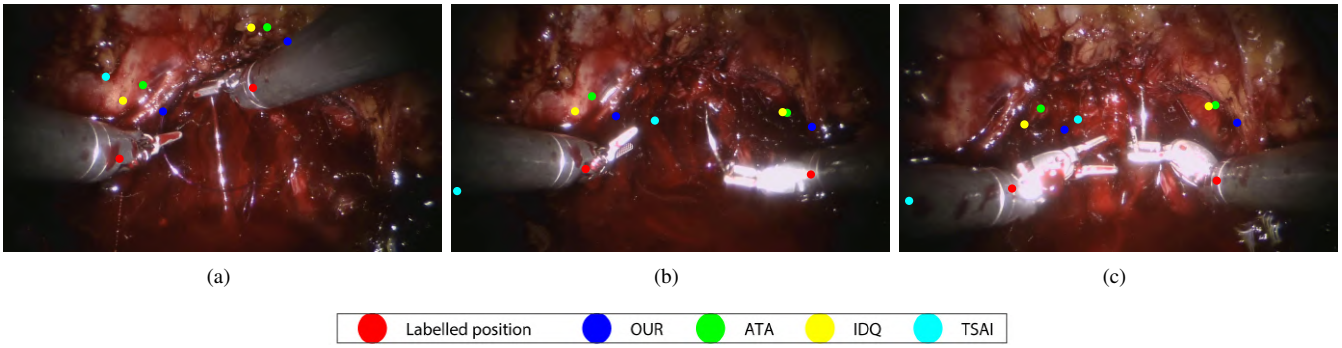



Fig. 7. Examples of re-projected PSM1's and PSM2's end-effector positions in a real surgery. The red dots show the labelled position, and the other dots are shown like Figure 6. The average re-projection errors are shown in Table II. The in-vivo dataset contains specularities and dynamic environment which cannot be accounted solely from kinematic data, and therefore present difficulties in calibrating the hand-eye matrix.


[9] J. Heller, D. Henrion, and T. Pajdla, "Hand-eye and robot-world calibration by global polynomial optimization," in *2014 IEEE Int. Conf. Robot. Autom. (ICRA'2014)*, May 2014, pp. 3157–3164.

[10] A. Malti, "Handeye calibration with epipolar constraints: Application to endoscopy," *Robot. Autom. Syst.*, vol. 61, no. 2, pp. 161 – 169, 2013.

[11] A. Malti and J. P. Barreto, "Hand-eye and radial distortion calibration for rigid endoscopes," in *Int. J. Med. Robot. Comput. Assist. Surg.*, Jan 2013, pp. 441–454.

[12] H. Zhuang, S. Roth, and R. Sudhakar, "Simultaneous robot/world and tool/flange calibration by solving homogeneous transformation equations of the form ax=yb," *IEEE Tran. Robot. Autom.*, vol. 10, no. 4, pp. 549–554, Aug 1994.

[13] A. Tabb and K. M. Ahmad Yousef, "Solving the robot-world hand-eye(s) calibration problem with iterative methods," *Mach. Vision. Appl.*, vol. 28, no. 5, pp. 569–590, Aug 2017.

[14] K. Koide and E. Menegatti, "General handeye calibration based on reprojection error minimization," *IEEE Robot. Autom. Lett. (RAL)*, vol. 4, no. 2, pp. 1021–1028, April 2019.

[15] K. Huang and C. Stachniss, "On geometric models and their accuracy for extrinsic sensor calibration," *2018 IEEE Int. Conf. Robot. Autom. (ICRA'2018)*, pp. 1–9, 2018.

[16] J. Heller, M. Havlena, A. Sugimoto, and T. Pajdla, "Structure-from-motion based hand-eye calibration using $l_\infty$ minimization," in *IEEE Conf. Comput. Vis. Pattern Recognit. 2011 (CVPR'2011)*, June 2011, pp. 3497–3503.

[17] Z. Wang, *et al.*, "Vision-based calibration of dual rcm-based robot arms in human-robot collaborative minimally invasive surgery," *IEEE Robot. Autom. Lett. (RAL)*, vol. 3, no. 2, pp. 672–679, April 2018.

[18] K. Pachtrachai, *et al.*, "Hand-eye calibration for robotic assisted minimally invasive surgery without a calibration object," in *2016 IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'2016)*, Oct 2016, pp. 2485–2491.

[19] L. Zhang, M. L. Ye, P. L. Chan, and G. Z. Yang, "Real-time surgical tool tracking and pose estimation using a hybrid cylindrical marker," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 12, pp. 921–930, 2017.

[20] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, Apr 2020.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 8490, May 2017.

[22] J. Donahue, *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," *CoRR*, vol. abs/1411.4389, 2014. [Online]. Available: http://arxiv.org/abs/1411.4389

[23] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," *CoRR*, vol. abs/1505.07427, 2015. [Online]. Available: http://arxiv.org/abs/1505.07427

[24] I. Melekhov, J. Kannala, and E. Rahtu, "Relative camera pose estimation using convolutional neural networks," *CoRR*, vol. abs/1702.01381, 2017. [Online]. Available: http://arxiv.org/abs/1702.01381

[25] Y. Shavit and R. Ferens, "Introduction to camera pose estimation with deep learning," *CoRR*, vol. abs/1907.05272, 2019. [Online]. Available: http://arxiv.org/abs/1907.05272

[26] J. Shi, *et al.*, "Calibrcnn: Calibrating camera and lidar by recurrent convolutional neural network and geometric constraints," in *2020 IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'2020)*, Oct 2020.

[27] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," *CoRR*, vol. abs/1709.08429, 2017. [Online]. Available: http://arxiv.org/abs/1709.08429

[28] M. Attia, M. Hossny, S. Nahavandi, and H. Asadi, "Surgical tool segmentation using a hybrid deep cnn-rnn auto encoder-decoder," in *2017 IEEE Int. Conf. Sys. Man. Cyber. (SMC' 2017)*, pp. 3373–3378.

[29] L. C. García-Peraza-Herrera, *et al.*, "Real-time segmentation of non-rigid surgical tools based on deep learning and tracking," in *Comput. Assist. Robot. Endo.*, Feb 2017, pp. 84–95.

[30] H. Wu, *et al.*, *Hand-Eye Calibration and Inverse Kinematics of Robot Arm Using Neural Network*. Springer International Publishing, 2014, pp. 581–591. [Online]. Available: https://doi.org/10.1007/978-3-319-05582-4_50

[31] G. A. Fontanelli, *et al.*, "A v-rep simulator for the da vinci research kit robotic platform," in *2018 IEEE Int. Conf. Biomed. Robot. Biomech. (Biorob)*, Aug 2018, pp. 1056–1061.

[32] S. H. Mark W. Spong and M.Vidyasagar, *Robot modeling and control*. John Wiley & Sons, Ltd., 2006.

[33] J. J. Craig, *Introduction to Robotics: Mechanics and Control, 3rd Edition*. Pearson, 2005.

[34] *ISI API User Guide: da Vinci Research Kit*, Intuitive Surgical Inc., 2010.

[35] F. A. Gers, J. A. Schmidhuber, and F. A. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Comput.*, vol. 12, no. 10, p. 24512471, Oct. 2000.

[36] Y. Zhou, *et al.*, "On the continuity of rotation representations in neural networks," *CoRR*, vol. abs/1812.07035, 2018. [Online]. Available: http://arxiv.org/abs/1812.07035

[37] V. Balntas, S. Li, and V. Prisacariu, "Relocnet: Continuous metric learning relocalisation using neural nets," in *Europe. Conf. Comput. Vis. (ECCV)*, Sept 2018.

[38] F. Vasconcelos, E. Mazamenos, J. Kelly, and D. Stoyanov, "Rcm-slam: Visual localisation and mapping under remote centre of motion constraints," *2018 IEEE Int. Conf. Robot. Autom. (ICRA'2019)*, pp. 9278–9284, 2019.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2017. [Online]. Available: https://arxiv.org/abs/1412.6980