Learning Eye-in-Hand Camera Calibration from a Single Image

Anonymous Author(s)
Affiliation
Address
email

Abstract: Eye-in-hand camera calibration is a fundamental and long-studied problem in robotics. We present a study on using learning-based methods for solving the problem online from a single RGB image, whilst training our models with entirely synthetic data. We study three main approaches, one using a direct regression model predicting the extrinsic matrix directly from images, one sparse correspondence model regressing 2D keypoints and using PnP, and one dense correspondence model that uses regressed depth and segmentation maps to enable ICP pose estimation. In our experiments, we benchmark these methods against each other and well established classical methods to find the surprising result that direct regression outperforms other approaches, and perform noise-sensitivity analysis to gain further insights into those results.

Keywords: Camera calibration, Robot manipulation, Sim-to-real

1 Introduction

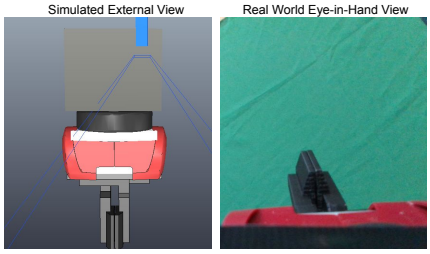

Figure 1: Left: External view of our simulated gripper & camera setup. Right: Real World Eye-in-Hand view.

Eye-in-hand setups, where images from a camera rigidly mounted to the end-effector are used to control a robot, are a popular choice for precise robot manipulation. Compared to control using a fixed external camera, the eye-in-hand setup enables precise control because images can be captured very close to an object, as well as generalisation across the workspace when actions are defined relative to the camera or end-effector [1, 2, 3]. To apply to precise manipulation tasks, these methods require accurate eye-in-hand extrinsic calibration, to ensure that actions or poses inferred in the camera's frame can be transformed to the robot's frame. Whilst calibration-free approaches exist, such as end-to-end control [4, 5], these are only suitable for model-free policy learning, and cannot be used for methods that involve explicit planning or reasoning about 3D space.

Classically, solving the eye-in-hand calibration problem is performed off-line [6, 7, 8, 9] and involves the use of a specific external calibration object, such as a checkerboard or an AprilTag. Poses of the calibration object can be estimated by the camera, and by recording them from several robot configurations, the rigid-body transformation between the camera and end-effector can be estimated analytically. Although with perfect information these methods can yield exact calibrations, they are in practice limited by how precisely the pose of the calibration object can be estimated, and by the accuracy of the recorded end-effector poses. Furthermore, these methods are unsuited for use by non-experts since they require dedicated external calibration objects and a new calibration dataset to be collected for each re-calibration. Finally, they are impractical as re-calibration needs to be performed every time the camera position changes due to (1) re-mounting, (2) wear and tear, and (3) collisions between the camera and the environment. Deep learning methods can alleviate these drawbacks. Given the initial overhead required to generate training data and train a model, a deep learning-based calibration method can be readily used to re-calibrate a camera when necessary. Using parts of the robot as the calibration object, they have the potential to perform calibration on-

Submitted to the 5th Conference on Robot Learning (CoRL 2021). Do not distribute.

line from a single image without the use of an external calibration objects. This potential has been demonstrated for *eye-to-hand* camera calibration, where an external camera observes the robot from a distance [10, 11].

To the best of our knowledge, there are currently no studies on using deep learning methods for *eye-in-hand* calibration, which is the focus of this paper. We propose that this calibration can be estimated directly from the camera's image itself, without requiring any external apparatus, as long as the image can observe part of the end-effector, such as the gripper's fingers (see Fig. 1). Based on this, we identify and benchmark three natural approaches for leveraging the power of deep learning in this way: two based on classical, analytical/geometric calibration where deep learning is used in place of components which are typically manually engineered, and one which uses deep learning to directly regress the camera calibration matrix. We show in our real-world experiments that the two approaches that rely on geometric analysis perform poorly, and we analyse empirically the structural limitations and challenges that cause this. We also show that, perhaps surprisingly, direct regression outperforms all alternatives, including even classical methods based on fiducial markers that collect data automatically. Finally, given that direct regression can estimate the calibration matrix from a single image online, we propose and evaluate a method for fusing multiple estimates of the camera pose, to increase the accuracy of the aggregated estimate.

As such, our work has three key contributions. (1) We investigate the suitability of three natural alternatives of using deep-learning for enabling eye-in-hand calibration from a single image, (2) We show the potential of end-to-end deep learning for online eye-in-hand camera calibration in everyday environments, outperforming typical classical approaches that collect data automatically, and (3) We provide analysis and insights for the two learning-based methods that rely on geometric analysis, through which we discover why, although useful for calibration with an external camera, these methods are not suitable for eye-in-hand calibration. A video and supplementary material for our paper can be found on our website: `https://sites.google.com/view/eye-in-hand` .

2 Related Work

Eye-in-hand calibration is a long-studied problem, with significant advances introduced in the 1990's with what are now well-established solutions. The problem consists of inferring the camera to end-effector pose for a camera mounted on a wrist of a robotic manipulator. To do so, several end-effector to robot base poses are recorded, along with corresponding estimates of calibration object to camera poses. The problem is then reduced to solving the $AX = XB$ equation through a formalism introduced by Shiu and Ahmad in [12]. Several works were then followed that mainly differ in their strategy for solving the $AX = XB$ equation. For instance, Tsai and Lenz [7] improve on the efficiency of Shiu and Ahmad's solution by proposing a closed-form solution. Park and Martin [8] and Dornaika and Harod [9] also consider such a closed-form solution while relying on Lie theory and unit quaternions respectively.

While the above approaches solve for rotation and translation separately, Dornaika and Harod [9] also propose a non-linear technique for solving for both simultaneously. Daniilidis proposes a solution to the problem using dual quaternions [6]. Finally, more recent works study various extensions for particular settings such as using structure for motion for calibration [13], using model-based pose estimation and tracking for online calibration [14], simultaneously considering the data time synchronisation problem [15], calibrating depth sensors with non-overlapping views [16], automatically doing both calibration pattern localisation and eye-in-hand calibration [17], and optimising the robot kinematics parameters and the camera to end-effector pose simultaneously [18].

Recently, deep learning methods have had great successes on closely related tasks such as pose estimation [19, 20] and end-to-end robotics manipulation from wrist-mounted cameras [4, 3]. Moreover, sim-to-real transfer has shown promise in alleviating the large data requirements that hinder the scalability of such methods [4, 21, 22, 23]. In this work, we investigate the naturally emerging question of how one might use such methods for eye-in-hand calibration, which is fundamental to a large number of robotics pipelines. Closest to our work, Lee et al. [11] and Labbe et al. [10] investigate two different deep learning approaches for eye-to-hand calibration of a camera looking straight at a robot. [11] is based on keypoint regression followed by Perspective-n-Point (PnP) [24] while [10] adopts a render and compare approach. In our work, we consider the eye-in-hand setup that is more common when precise manipulation needs to be achieved by looking closely at the end-effector

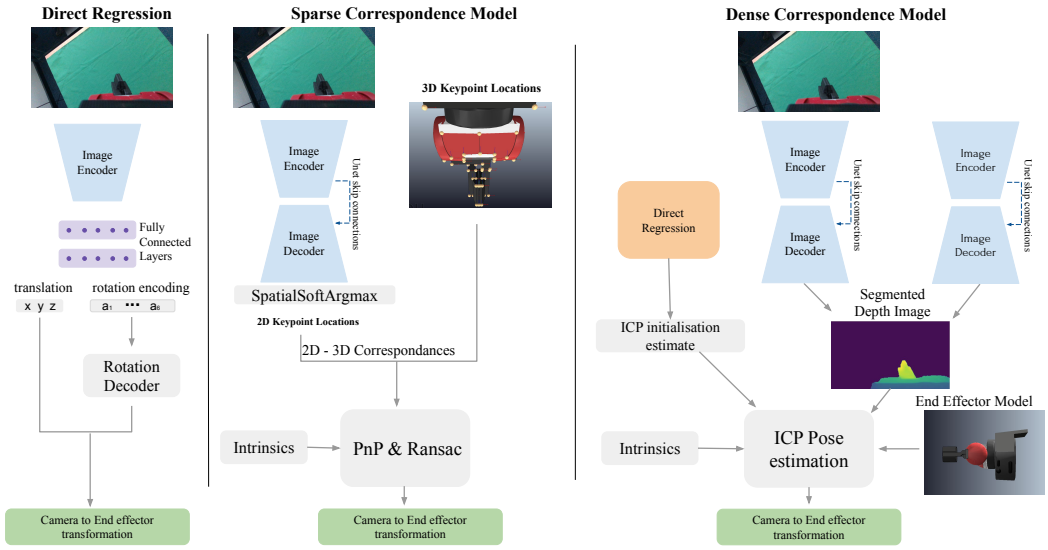

Figure 2: Left: Illustration of the Direct Regression model. Centre: Illustration of the Sparse Correspondence model. Right: Illustration of the Dense Correspondence model.

of the robot [4, 25]. This problem also differs in its properties from the eye-to-hand setup, since in a typical image there is now a much more constrained view that only allows for the tip of the end-effector to be visible and to be used as an anchor for pose estimation.

3 Methods

In this section, we present three methods which are conceptually promising for eye-in-hand calibration from a single RGB image, and which also raised interesting and novel scientific questions that we answer in sections 4 and 5. The first method is an end-to-end direct regression method that estimates the pose of the camera from an RGB image, while the other two methods use deep learning to provide key missing components for well established geometric pose estimation approaches. The latter has been shown to work well in other problem settings [11, 26, 27, 28], but eye-in-hand calibration has the particularity that not only the camera is too close to the end-effector for typical vision sensors to capture a depth image of it, but also the visible geometry varies widely within our inference space. To test whether it is still feasible to use such approaches, our second method uses deep learning to regress 2D locations of predefined keypoints on the end-effector, that are then passed into the PnP [24] algorithm to solve for the pose of the camera. The main challenge with this method is that the large variations in the visible geometry lead to keypoint occlusions and some keypoints being often outside of the image frame. This is illustrated in Fig. 3. Our final method tests whether we can alleviate this pitfall by using deep learning to regress a segmented depth image of the end-effector and an initial guess of the camera pose, and then refine this initial estimate with the Iterative Closest Point (ICP) algorithm [29, 30, 31]. All three methods are illustrated in Fig. 2.

3.1 Problem Setting

Our problem setting consists of the typical eye-in-hand calibration problem, where the aim is to estimate the camera's extrinsic matrix, which is the camera to end-effector pose, $T_{EC} = [R_{EC}|t_{TC}] \in SE(3)$, where $R_{EC} \in SO(3)$ and $t_{EC} \in \mathbb{R}^3$ is the orientation and position of the camera in the end-effector frame, respectively. We also define the end-effector to robot base pose as T_{BE}, the calibration object to camera pose as T_{CO}, and an image captured by the wrist-mounted camera as I. Throughout this paper, we use $\tilde{\cdot}$ to denote an estimated quantity.

As opposed to classical approaches, we constrain ourselves to eye-in-hand calibration from a single RGB image without any external apparatus, enabling our methods to be readily deployed in the wild for online eye-in-hand calibration. We further constrain ourselves to using only synthetic training data. This has the benefit of alleviating the costly data requirements of deep learning approaches while also providing us with ground truth labels that enable our proposed methods to learn effec-

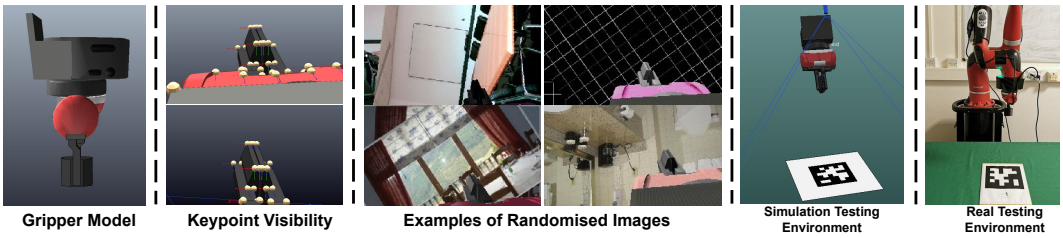

Figure 3: Let to right: Model of the gripper used for data generation. Illustration of keypoint visibility from different view points. Examples of randomised images used for training deep learning-based models in simulation. Simulation testing environment and real testing environment.

tively. Finally, we assume that we have an accurate estimate of the camera intrinsic matrix, which is typically provided by the manufacturers, and access to the CAD model of our robot's end-effector.

3.2 Direct Regression Model

Direct regression (DR) is the simplest of the methods we consider. As illustrated in the right diagram of Fig. 2, a neural network (NN) is tasked with regressing the camera pose parameters from a single RGB image and is trained in simulation using ground truth labels in a supervised setting. The only constraint we introduce into the problem comes from the parameterisation of the orientation, where we use the 6D rotation encoding introduced in [32].

Overall, our network outputs are 9 dimensional, with 3 dimensions representing position and 6 the orientation. It consists of a convolutional encoder followed by linear layers that regress the camera pose. We trained the network on the ground truth labels using the Mean Squared Error (MSE) loss, and we fully breakdown the architecture and training details in our supplementary material.

3.3 Sparse Correspondence Model

Our sparse correspondence model (SC) is illustrated in the middle of Fig. 2. Starting from an RGB image, it uses a U-net [33] type architecture to regress a set of 2D keypoints in image space. The Unet architecture downsamples the input image into some abstract feature space and then upsamples it back to the original resolution, while providing skip connections through concatenation at each resolution level [33]. On the output layer we apply a spatial-soft-argmax [34] activation in order to regress 2D image keypoints. We use the sum of an L1 and MSE losses to train our network on the ground truth labels, where each dimension of the output predicts the 2D location of a predefined 3D keypoint. A detailed breakdown our architecture and training parameters can be found in our supplementary material.

At test time, we use the network outputs, the corresponding 3D keypoint locations defined in the object's model, and the camera's intrinsic matrix to estimate the camera pose with PnP and RANSAC [24], which is a popular outlier rejection algorithm.

Finally, in order to alleviate the issue of some keypoints being out of frame depending on the viewpoint, we (1) consider 38 keypoints that are present in at least 70% of the training images, (2) for any keypoint that remains out of frame, we train the model to predict its projection in the image. This way, at test time, we can infer which keypoints are out of frame by looking at the predictions at the rim, and ignore these predictions for the PnP pose estimation.

3.4 Dense Correspondence Model

Our Dense Correspondence Model (DC) is illustrated on the right of Fig. 2. Starting from an RGB Image, it uses three NNs that are trained independently to estimate the pose of the camera in two stages. In the first stage, the three NNs are used to predict (1) a depth map, (2) a segmentation mask of the gripper, and (3) an ICP initialisation. In the second stage, (1) the segmentation mask is used to segment the depth map to only return depth values on gripper pixels, (2) the camera intrinsic matrix is used to project the segmented depth map to a point cloud, and (3) ICP is used to approximate dense correspondences between the estimated point cloud and the model of the end-effector in order to refine the camera pose estimate from the first stage. The network used for

initialisation is the same as the one used for direct regression, and the same UNet architecture as for the sparse correspondence model is used to predict the depth and segmentation maps. See the supplementary material for further architecture design choices and training details.

3.5 Fusing Multiple Estimates

Compared to classical eye-in-hand calibration methods, which require calibration to be done offline and with special apparatus, the proposed deep learning alternatives can estimate the pose of the camera online in unstructured environments, such as in the middle of a manipulation task. This creates the possibility of fusing multiple estimates to increase the overall calibration accuracy. In order to explore this, we implemented a simple aggregation algorithm that rejects 20% of the least likely samples from a set of candidate estimates, under a Gaussian data model. It then averages the remaining samples together to yield the final estimate. Pseudocode for this procedure is available in our supplementary material.

3.6 Dataset Generation and Sim-to-Real Transfer

We train our networks entirely with simulated data using the Coppelia [35] simulator. In order to generate the dataset, we randomise the pose of the camera relative to the end-effector at each simulated timestep and record (1) the current RGB, depth and end-effector segmentation images, (2) the ground truth extrinsic matrix and (3) the 2D keypoint locations in image space.

In order to overcome the "reality gap" we apply visual Domain Randomisation [21, 23, 36]. Specifically, we (1) randomise the colours/textures of the simulated gripper and the light sources on the simulator, (2) replace the background of our images with random images of textures and indoor scenes [37, 38], and (3) apply a post-processing colour jitter operation [39], further randomly perturbing the brightness, contrast, saturation and hue of the whole image. In total, we generated 10000 labelled images in approximately 30 minutes, examples of which can be seen in Fig. 3.

4 Experiments

In our experiments, we evaluated how the different proposed deep learning approaches compare to each other and to various standard off the shelf calibration methods both in simulation and in the real world. From them emerges the surprising result that simple direct regression using end-to-end deep learning outperforms both all classical approaches tested and the correspondence based deep learning methods. As such, we performed a series of analysis experiments that we describe in sec. 5 that are aimed specifically at gaining insights onto this surprising observation.

In our comparisons we include all our deep learning based methods and the following established methods readily available on OpenvCV [40]: Dainiilidis et al. [6], Tsai and Lenz [7], Doraika and Harod [9], and Park and Martin [8]. For all experiments, we use the Realsense D435 [41] (or its simulated counterpart) at a 480×848 resolution, and for the deep learning methods we downsample these images to a resolution of 144×256.

Comparing classical methods to our deep learning ones fairly can be somewhat challenging since classical methods require a training set of several end-effector to robot and calibration object to camera poses to perform a single calibration, while our methods only require a single image. In sections 4.1 and 4.2 we describe our procedures and what we did to mediate this issue, and in our supplementary material we provide pseudocode for each of the procedures.

4.1 Simulation Experiment

Our simulated environment consists of a wrist-mounted camera attached to a Sawyer robot's gripper that is free to move around an AprilTag (see Fig. 3). To benchmark all methods, we (1) Collect a dataset $\{I_i, T^i_{BE}, \tilde{T}^i_{CO}\}_{i=1}^{15}$ of images and corresponding AprilTag and end-effector poses, with the tag poses $\tilde{T}_{CO}$ estimated using the AprilTags3 library [42, 43, 44], (2) Use all of the 15 datapoints in this dataset to estimate the extrinsic matrix using classical methods, and evaluate this estimate, (3) Use each of the 15 datapoints in this dataset independently to estimate the extrinsic matrix using the proposed learned methods, and evaluate each of the single image estimates, and (4) For each

	Simulation		Real World
Method	ϵ_t [mm]	ϵ_R [degrees]	ϵ_{std} [mm]
TSAI [7]	(171.4 ± 271.2)	(15.9 ± 20.6)	(17.5 ± 18.9)
PARK [8]	(68.7 ± 70.1)	(6.3 ± 4.7)	(15.3 ± 14.3)
HORAUD [9]	(68.9 ± 72.2)	(6.3 ± 4.7)	(16.3 ± 16.6)
DANIILIDIS [6]	(110.8 ± 122.8)	(11.5 ± 29.4)	(14.7 ± 13.6)
DR	$\mathbf{(13.4 \pm 4.1)}$	$\mathbf{(4.4 \pm 1.4)}$	(10.6 ± 4.1)
DR (fusion)	$\mathbf{(13.4 \pm 4.1)}$	$\mathbf{(4.4 \pm 1.4)}$	$\mathbf{(10.4 \pm 4.0)}$
SC	(363.6 ± 353.1)	(117.4 ± 48.2)	(516.4 ± 338.1)
SC (fusion)	(338.5 ± 301.0)	(120.2 ± 42.7)	(291.2 ± 129.6)
DC	(93.5 ± 45.9)	(33.9 ± 25.7)	(21.0 ± 10.4)
DC (fusion)	(73.4 ± 21.2)	(23.1 ± 13.9)	(15.0 ± 1.7)

Table 1: Benchmarking results for classical methods and proposed deep learning-based methods: Direct Regression (DR), Sparse Correspondences (SC) and Dense Correspondences (DC), along with their fusion variants where multiple estimates are aggregated together.

of the learned methods, we fuse together all 15 independent predictions using our fusion procedure described in 3.5 and evaluate the fused estimate.

We repeat this procedure for 100 different ground truth extrinsic matrices, average the results and display them in table 1. We evaluate translation and rotation independently: Given an estimate of the extrinsic $\tilde{T}_{EC} = [\tilde{R}_{EC}|\tilde{t}_{EC}]$, and the ground truth $T^*_{EC} = [R^*_{EC}|t^*_{EC}]$, we define the position error as $e_t = ||\tilde{t}_{EC} - t^*_{EC}||_2$, where $|| \cdot ||_2$ is the L2 norm, and the rotational error as $e_R = \theta$, the angle from the axis-angle representation of the rotation matrix $R_\Delta = (w, \theta)$ that satisfies the relationship $R^*_{EC} = R_\Delta \tilde{R}_{EC}$, where w is the axis of rotation, and θ is the angle of rotation.

4.2 Real World Experiment

The real world evaluation environment is analogous to the simulated environment and consists of a Sawyer robot with a wrist-mounted camera moving around an AprilTag (see Fig. 3). In order to evaluate our methods in the real world we (1) Collect a training data bank $\mathcal{D}_{train} = \{I_i, T^i_{BE}, \tilde{T}^i_{CO}\}_{i=1}^{40}$ and an evaluation dataset $\mathcal{D}_{eval} = \{T^i_{BE}, \tilde{T}^i_{CO}\}_{i=1}^{60}$, where AprilTag poses are estimated using an off-the-shelf library [43, 44], (2) We sample 40 training datasets of 15 datapoints each from our training data bank, (3) Use all 15 datapoints in each training set to get an estimate of the camera extrinsic matrix using each classical method, and evaluate that estimate, (4) We use each one of the 15 datapoints in each dataset to get an estimate of the extrinsic matrix using our learned methods, and evaluate each single estimate, and (5) For each of the learned methods, we fuse together all 15 independent predictions using the procedure described in 3.5 and evaluate the fused estimate.

We repeat this procedure for two different ground truth extrinsic, average the results and display them in table 1. Since ground truth extrinsic values are not available in the real world, we take inspiration from [6, 7] and use an indirect error metric: For each corresponding end-effector to robot pose and calibration object to camera pose from the evaluation dataset, $\{T^i_{BE}, \tilde{T}^i_{CO}\} \in \mathcal{D}_{eval}$, we estimate the pose of the calibration object in the robot frame, $\tilde{T}^i_{BO} = T^i_{BE}\tilde{T}_{EC}\tilde{T}^i_{CO} = [\tilde{R}^i_{WO}|\tilde{T}^i_{BO}]$, where $\tilde{T}_{EC}$ is an estimate of the camera to end-effector pose that we are evaluating. We then define the error metric $\epsilon_{std} = \left(1/60 \sum_{i=1}^{60} \left|\left|\tilde{t}^i_{BO} - \mu\right|\right|_2^2\right)^{1/2}$ as the standard deviation of the estimated calibration object position, where $\mu = 1/60 \sum_{i=1}^{60} \tilde{t}^i_{BO}$ is the estimated mean object position.

We emphasise that although the AprilTag is visible in all images in both the simulated and real environments, it is never used to help the calibration of our deep learning methods. We use these images simply to ensure that we have the exact same training and evaluation datasets for both classical and our learned methods.

4.3 Results

All benchmarking results are shown in table 1. Our main observation is that the direct regression method outperforms all others, which is a striking and surprising result. First, it was not expected to outperform the classical methods since these use analytical solutions to estimate the extrinsic, which

in absence of noise in the system should give perfect calibration. This raises the question about the sensitivity of these methods to said noise, which we investigate in section 5.

Second, it was also not expected to outperform the correspondence based learned methods, since those introduce strong geometric constraints that intuitively should help guide the model to good solutions. We thoroughly investigate the reasons behind this observation in sections 5.2 and 5.3.

5 Analysis

In this section, we aim to build a deeper understanding of the surprising results that we observed in our experiments, which showed that a direct regression of calibration parameters outperforms methods which incorporate well-understood, geometric modelling. We split this section into three parts, each analysing the performance of one particular method.

5.1 Classical Methods

In order to better understand the results from our experiments in section 4, we first aim to assess how sensitive classical methods are to sources of noise in the system. Assuming the main source of error stems from the calibration object pose detection, we fix every other source of noise in simulation to the ground truth value. We then vary the noise injected to the calibration object's pose by controlled amounts, and observe how this affects the quality of the calibration result from classical methods.

Precisely, the noise scale added is varied from $0mm$ to $10mm$ and $0°$ to $10°$, in increments of $0.5mm$ and $0.5°$, and for each noise tier calibration is performed with each of the classical methods considered. We then compare the result to the ground truth extrinsic and obtain position errors. This procedure is repeated for 100 camera extrinsic, with the average errors obtained and their standard deviations illustrated in the top graph of Fig. 4.

We can see that as expected with perfect information the classical methods return an exact solution. However, we also see that the quality of the calibration rapidly deteriorates with increased noise in the calibration object's position. This illustrates our motivation for using a deep learning based approach, since our methods are by construction independent from any external pose estimation algorithm's quality.

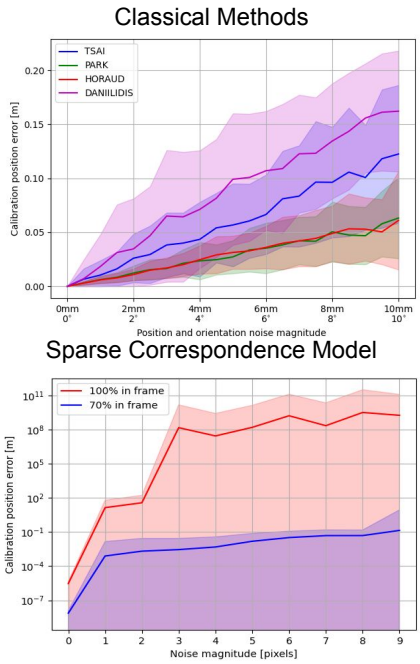

Figure 4: Top: Analysis of sensitivity to noise in estimated tag positions of classical methods. Bottom: Analysis of sensitivity to noise in estimated keypoint positions of PnP+RANSAC.

5.2 Sparse Correspondences

In order to better understand why our sparse correspondence model did not perform strongly we set up another controlled experiment. Starting from ground truth 2D-3D correspondences, we added fixed amounts of noise to the 2D keypoints and observed the effect on the quality of the PnP + RANSAC pose estimation. Specifically, we iterated through 200 simulated datapoints, and for each $\{$2D keypoints, 3D keypoints, extrinsic $\}$ tuple, we perform PnP with the noise-injected 2D keypoints and compared the result to the ground truth. We repeated this for different magnitudes of noise, ranging from 0 to 9 pixels, in increments of 1 pixel, and averaged over all the datapoints considered. We plotted the resulting errors in the "70% in frame" curve in the bottom graph of Fig. 4.

During our investigation we also considered a smaller amount of keypoints, but ones that always remain in frame. With those we observed that our network training performance increased, but our final extrinsic estimation did not improve. In order to understand this behaviour, we repeated the controlled noise experiment but only considering the keypoints that always appearing in frame in the images, which resulted in 12 remaining keypoints. This is plotted in the "100% in frame" curve in the bottom graph of Fig. 4.

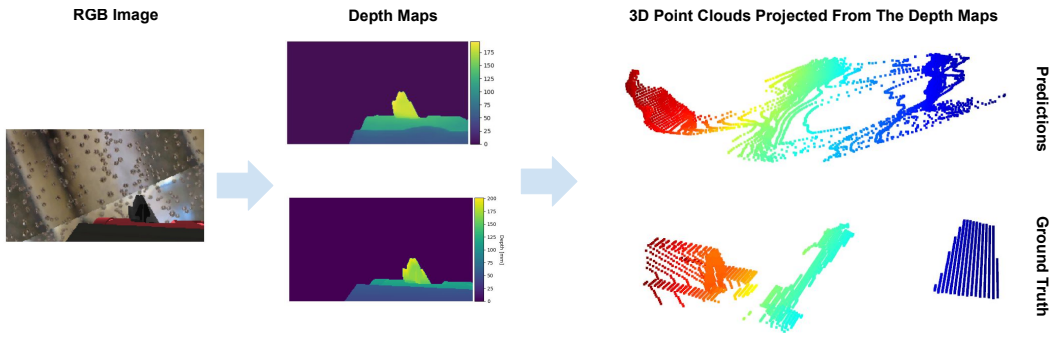

Figure 5: Left: RGB image of the gripper rendered in simulation. Top centre: Depth image predicted by our model from the simulated RGB image. Top right: point cloud projected from the predicted depth image. Bottom centre: Depth image rendered in simulation from the same view point. Bottom right, point cloud projected from the ground truth depth image.

From Fig. 4, we clearly see that there is a very high sensitivity of sparse correspondence pose estimation to errors in 2D keypoint pixel locations. For the keypoints in our training set, in order to get an extrinsic calibration position error of less than 1cm, the average keypoint prediction error from our networks would need to be less than 4 to 5 pixels. This is with the additional challenge of the network having to keep consistent predictions even though sometimes the keypoints appear out of frame. On the other hand, if we consider the easier to learn problem of only predicting keypoints that are always appear in-frame, then the sensitivity skyrockets after single pixel average error in keypoint 2D locations. Overall, we believe that this sensitivity to noise in the keypoints is the primary reason why the sparse correspondence model did not give a strong performance.

5.3 Dense Correspondences

Although conceptually using a dense geometric correspondence approach to the pose estimation should alleviate some of the drawbacks of using sparse correspondences, we found that in practice our model that uses ICP to refine estimates made by the DC method decreases their accuracy.

The reason for this becomes apparent when considering the illustrations shown in Fig. 5: Although the depth values obtained by the neural networks look appealing when projected into the image plane, they fall short when considering the full 3D structure. We conjecture that the main reason for this is that in order to recover a correct 3D structure from such close distances, depth values need to have strong discontinuous jumps that typical neural networks have a lot of trouble modelling. This is supported by the observations in Fig. 5: We can see that the overall depth increase seems to match the ground truth values, but there is an averaging effect that tends to make the depth values increase smoothly, which doesn't allows us to recover the correct shape. As such, we can conclude that using a simple depth regression technique in this setting is inappropriate for this task's requirements, with further research needed to overcome the discontinuity problem.

6 Conclusion

In this work, we presented and evaluated three deep learning based methods for online eye-in-hand calibration from a single image. We trained all our models entirely on synthetic images, and evaluated them both against each other and against long established calibration methods in simulation and the real world. Surprisingly, our experiments indicated that a direct regression method from images to camera extrinsics outperformed other alternatives. In order to better understand this result, we then conducted a series of introspection experiments, which indicated that a strong shortcoming from geometry-based methods seem to be their sensitivity to noise in their input data. Finally, while depth map regression for dense correspondences conceptually could have provided the answer to such shortcomings, our experiments indicated that due to pronounced discontinuities in the depth profiles from such close-up viewpoints, further research is needed into learning-based depth prediction for eye-in-hand robot manipulation.

References

[1] E. Johns, S. Leutenegger, and A. J. Davison. Deep learning a grasp function for grasping under gripper pose uncertainty. In *International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[2] Y. Litvak, A. Biess, and A. Bar-Hillel. Learning pose estimation for high-precision robotic assembly using simulated depth images. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3521–3527, 2019.

[3] E. Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. *arXiv preprint arXiv:2105.06411*, 2021.

[4] E. Valassakis, N. Di Palo, and E. Johns. Coarse-to-fine for sim-to-real: Sub-millimetre precision across the workspace. *arXiv preprint arXiv:2105.11283*, 2021.

[5] E. Y. Puang, K. P. Tee, and W. Jing. Kovis: Keypoint-based visual servoing with zero-shot sim-to-real transfer for robotics manipulation. *arXiv preprint arXiv:2007.13960*, 2020.

[6] K. Daniilidis. Hand-eye calibration using dual quaternions. *The International Journal of Robotics Research*, 18(3):286–298, 1999.

[7] R. Y. Tsai, R. K. Lenz, et al. A new technique for fully autonomous and efficient 3 d robotics hand/eye calibration. *IEEE Transactions on robotics and automation*, 5(3):345–358, 1989.

[8] F. C. Park and B. J. Martin. Robot sensor calibration: solving ax= xb on the euclidean group. *IEEE Transactions on Robotics and Automation*, 10(5):717–721, 1994.

[9] R. Horaud and F. Dornaika. Hand-eye calibration. *The international journal of robotics research*, 14(3):195–210, 1995.

[10] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. Single-view robot pose and joint angle estimation via render & compare. *arXiv preprint arXiv:2104.09359*, 2021.

[11] T. E. Lee, J. Tremblay, T. To, J. Cheng, T. Mosier, O. Kroemer, D. Fox, and S. Birchfield. Camera-to-robot pose estimation from a single image. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9426–9432. IEEE, 2020.

[12] Y. C. Shiu and S. Ahmad. Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax= xb. 1987.

[13] N. Andreff, R. Horaud, and B. Espiau. Robot hand-eye calibration using structure-from-motion. *The International Journal of Robotics Research*, 20(3):228–248, 2001.

[14] K. Pauwels and D. Kragic. Integrated on-line robot-camera calibration and object pose estimation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2332–2339. IEEE, 2016.

[15] F. Furrer, M. Fehr, T. Novkovic, H. Sommer, I. Gilitschenski, and R. Siegwart. Evaluation of combined time-offset estimation and hand-eye calibration on robotic datasets. In *Field and Service Robotics*, pages 145–159. Springer, 2018.

[16] F. Faion, P. Ruoff, A. Zea, and U. D. Hanebeck. Recursive bayesian calibration of depth sensors with non-overlapping views. In *2012 15th International Conference on Information Fusion*, pages 757–762. IEEE, 2012.

[17] M. Antonello, A. Gobbi, S. Michieletto, S. Ghidoni, and E. Menegatti. A fully automatic hand-eye calibration system. In *2017 European conference on mobile robots (ECMR)*, pages 1–6. IEEE, 2017.

[18] X. Liu, H. Madhusudanan, W. Chen, D. Li, J. Ge, C. Ru, and Y. Sun. Fast eye-in-hand 3-d scanner-robot calibration for low stitching errors. *IEEE Transactions on Industrial Electronics*, 68(9):8422–8432, 2020.

[19] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[20] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020.

[21] E. Valassakis, Z. Ding, and E. Johns. Crossing the gap: A deep dive into zero-shot sim-to-real transfer for dynamics. *arXiv preprint arXiv:2008.06686*, 2020.

[22] R. Alghonaim and E. Johns. Benchmarking domain randomisation for visual sim-to-real transfer. *arXiv preprint arXiv:2011.07112*, 2020.

[23] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[24] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[25] R. L. Haugaard, J. Langaa, C. Sloth, and A. G. Buch. Fast robust peg-in-hole insertion with continuous visual servoing. *arXiv preprint arXiv:2011.06399*, 2020.

[26] S. Zakharov, I. Shugurov, and S. Ilic. Dpod: 6d pose object detector and refiner. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1941–1950, 2019.

[27] R. Sheffer and A. Wiesel. Pnp-net: A hybrid perspective-n-point network. *arXiv preprint arXiv:2003.04626*, 2020.

[28] F. Xiong, C. Liu, and Q. Chen. Region pixel voting network (rpvnet) for 6d pose estimation from monocular image. *Applied Sciences*, 11(2):743, 2021.

[29] Y. Chen et al. Object modeling by registration of multiple range images. In *ICRA*, 1991.

[30] P. Besl et al. A method for registration of 3-d shapes. In *TPAMI*, 1992.

[31] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. In *IJCV*, 1994.

[32] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5738–5746, 2019.

[33] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *MICCAI*. Springer International Publishing, 2015.

[34] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[35] E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013. www.coppeliarobotics.com.

[36] R. Alghonaim and E. Johns. Benchmarking domain randomisation for visual sim-to-real transfer. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[37] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420. IEEE, 2009.

[38] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. In *Conference on Robot Learning*, pages 357–368. PMLR, 2017.

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[40] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[41] Intel. Depth camera d435, May 2021. URL https://www.intelrealsense.com/depth-camera-d435/.

[42] pupil-apriltags. URL https://pypi.org/project/pupil-apriltags/.

[43] J. Wang and E. Olson. AprilTag 2: Efficient and robust fiducial detection. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4193–4198. IEEE, oct 2016. ISBN 978-1-5090-3762-9. doi:10.1109/IROS.2016.7759617.

[44] D. Malyuta. Guidance, Navigation, Control and Mission Logic for Quadrotor Full-cycle Autonomy. Master thesis, Jet Propulsion Laboratory, 4800 Oak Grove Drive, Pasadena, CA 91109, USA, Dec. 2017.