EasyHeC: Accurate and Automatic Hand-eye Calibration via Differentiable Rendering and Space Exploration

Linghao Chen[1,2]   Yuzhe Qin[2]   Xiaowei Zhou[1]   Hao Su[2]

[1]Zhejiang University   [2]University of California, San Diego

Abstract— Hand-eye calibration is a critical task in robotics, as it directly affects the efficacy of critical operations such as manipulation and grasping. Traditional methods for achieving this objective necessitate the careful design of joint poses and the use of specialized calibration markers, while most recent learning-based approaches using solely pose regression are limited in their abilities to diagnose inaccuracies. In this work, we introduce a new approach to hand-eye calibration called EasyHeC, which is markerless, white-box, and offers comprehensive coverage of positioning accuracy across the entire robot configuration space. We introduce two key technologies: differentiable rendering-based camera pose optimization and consistency-based joint space exploration, which enables accurate end-to-end optimization of the calibration process and eliminates the need for the laborious manual design of robot joint poses. Our evaluation demonstrates superior performance in synthetic and real-world datasets, enhancing downstream manipulation tasks by providing precise camera poses for locating and interacting with objects. The code will be available upon publication of this paper.

I. INTRODUCTION

Hand-eye calibration, a fundamental problem in robotics with a long history, addresses the problem of determining the transformation between a camera and a robot. It aligns the two most important modalities for robot sensing: proprioception and perception. Thus, hand-eye calibration serves as a crucial first step for various visual robotics applications.

Traditional calibration methods rely on markers, such as Aruco tags [1] or checkerboard patterns [2, 3], to establish a connection between the robot and camera frames. Despite being grounded in robust theory, numerous practical challenges arise when implementing these methods in real-world robotic systems. Research indicates that the accuracy of traditional approaches is considerably influenced by factors such as marker fixture quality, marker localization accuracy, and the choice of joint poses. Crucially, varying camera placements for distinct tasks require the creation of different joint pose trajectories, a task that is highly demanding for humans. While industry users may achieve satisfactory accuracy through extensive tuning for specific tasks, traditional marker-based methods fall short in the quest to develop versatile robots with high accuracy and minimal human intervention in daily environments. Consequently, there is an urgent demand for an accurate, automated hand-eye calibration solution that is both easy to implement and user-friendly. This necessity is especially pertinent given the
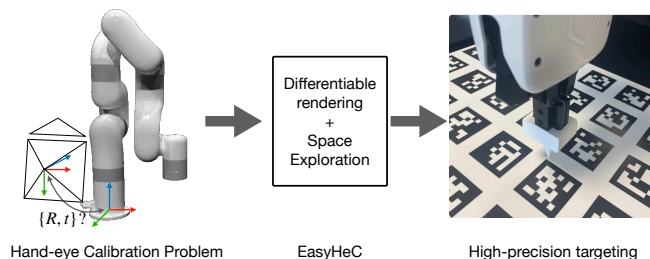
Corresponding author: Hao Su.

Fig. 1: **We propose EasyHeC for hand-eye calibration.** Left: Hand-eye calibration problem. We aim to estimate the relative pose between the camera and the robot's base. Middle: The main techniques of our approach EasyHeC are differentiable rendering IV-B and space exploration IV-C. Right: High-precision targeting with a real robot arm.

growing application of computer vision algorithms across a broad spectrum of robotics tasks.

More recently, efforts have been made to develop marker-less hand-eye calibration techniques [4]–[6], utilizing learning-based methods to minimize the cost and labor associated with calibration. However, the performance of these approaches largely depends on the scale and quality of their training data. Both traditional and learning-based methods exhibit uneven hand positioning error distribution in the camera space across the robot's configurations. Greater accuracy is achieved for configurations closely resembling the sampled ones, while larger discrepancies are observed in areas of the configuration space that remain unexplored during calibration. Additionally, most learning-based methods lack transparency due to their reliance on direct regression, making it challenging to diagnose calibration errors in real-world applications.

In this work, we aim to simplify hand-eye calibration for robotics researchers. To this end, we propose a system called EasyHeC, which offers several key features:

- No need for markers to be attached to the robot;
- Easy diagnosis with a white-box design;
- Elimination of laborious manual design of robot joint poses for calibration;
- Comprehensive coverage of positioning accuracy across the entire robot configuration space.

To achieve our goal, we propose two key technologies in our EasyHeC: (i) differentiable rendering-based camera pose optimization, and (ii) consistency-based joint space exploration.

First, we substitute the widely used objective function based on transformation equivalence, such as $AX = XB$ [3, 7, 8], with a loss function grounded in per-pixel segmentation of the robot arm. We assume that the 3D model of the robot arm is readily available, which is a reasonable assumption as most commonly used arms can be found online. In our formulation, the transformation between the camera and the robot base frames is optimized by minimizing the discrepancy between the projection of the 3D arm model and the observed segmentation mask. This white-box approach establishes a direct connection between the camera observations and camera poses, enabling end-to-end optimization of the calibration process and rendering our system highly diagnosable. Moreover, by leveraging the shape of the robot arm itself, we eliminate the need for calibration markers and reduce the financial cost of the hand-eye-calibration process. As the robot arm is considerably larger than the markers attached to it, the arm's mask offers substantially more evidence for estimating the transformation than traditional marker points.

Second, we develop a consistency-based joint space exploration module that facilitates the selection of informative joint poses. This module identifies the highest informative robot joint poses by evaluating the consistency with a set of camera pose candidates. This automatic selection enables our method to eliminate the need for the manual design of the robot joint poses and provide comprehensive coverage of accuracy across the entire configuration space. Moreover, since this process is performed in simulation, it is free of cost and can be performed many times to improve the accuracy of the calibration.

We evaluated our method on several synthetic and real-world datasets and provide ablation studies of the different design choices of the proposed system. When evaluated on synthetic datasets, EasyHeC **outperforms a traditional-based baseline by 4 times in accuracy** with a single view, and achieves an accuracy of around 0.2 cm with 5 views. Upon evaluation using a publicly available real-world dataset, EasyHeC consistently demonstrates superior performance across all metrics. Furthermore, we show that our proposed hand-eye calibration system can enhance downstream manipulation tasks by providing precise camera poses for locating and interacting with objects. Our system is easy to deploy and user-friendly, and we are dedicated to open-source it for the benefit of the robotics community.

II. RELATED WORK

Traditional Hand-eye Calibration. The Hand-eye Calibration (HeC) problem encompasses two settings: (i) eye-in-hand, where "eye", the camera is fixed to the world; and (ii) eye-to-hand, where the camera is fixed to "hand", commonly the end-effector. The traditional method for HeC involves explicit tracking of a marker [1]. Depending on the HeC setting, the marker is attached to the robot end-effector for eye-in-hand or fixed to the ground for eye-to-hand. The camera pose is then obtained by solving a linear system [2, 3, 7, 8]. However, these methods require the marker to be fully visible, which makes designing a calibration trajectory a challenging task. To address this limitation, marker-free methods [9]–[11] have been proposed, which use Structure-from-Motion (SfM) to estimate camera motion in the eye-in-hand setting. However, such methods rely on rich visual features for correspondence and are not suitable for tracking robot motion in the eye-to-hand setting. Unlike the aforementioned methods, our work tackles the eye-to-hand HeC problem in a marker-less manner. Instead of solving a linear system, our approach formulates HeC as an iterative optimization problem by minimizing the per-pixel discrepancy of the robot mask.

Learning-based Hand-eye Calibration. Several recent methods have proposed learning-based approaches to solve the HeC problem. For example, DREAM [12] uses a deep neural network to predict robot joint keypoints and then recovers camera poses using the Perspective-n-point (PnP) algorithm [13]. Another approach [4] uses a neural network as a refinement module to iteratively predict robot delta states from rendered images and observations. However, these regression-based methods do not establish a direct connection between camera pose and observation, limiting practical diagnosis. Another approach [14] directly regresses camera pose from images but lacks robustness in practical scenarios. Another recent work [5] uses neural networks for end-effector segmentation and the iterative closest point (ICP) algorithm [15] for camera pose estimation, but relies on depth sensors and may not work well without high-quality depth data. Another proposal [6] tracks a reference point on the robot and regresses camera pose, while a recent work [16] uses differentiable rendering but jointly optimizes camera pose and robot shape with single-image input, leading to ambiguities and reduced performance. In contrast, our approach leverages multi-view constraints for disambiguation and achieves superior performance.

Differentiable Rendering. Differentiable rendering has been widely utilized in various 3D computer vision tasks to optimize 3D properties using 2D observations. For instance, methods such as [17]–[21] have proposed differentiable rendering techniques for tasks such as single image 3D reconstruction [17]–[19], joint shape and pose learning [20], and novel object pose estimation [21]. These methods employ differentiable rendering to bridge the gap between 2D and 3D data, allowing for effective optimization of 3D properties using neural networks and 2D supervision. In our work, we leverage the concept of inverse rendering to optimize the transformation between the camera and the robot. Our approach is capable of utilizing a single image for HeC, but it achieves improved performance with the availability of more images.

III. PROBLEM FORMULATION

Given a robot arm with a known 3D model and a camera with known intrinsic parameters, the objective is to solve the eye-to-hand calibration problem, which is to estimate the
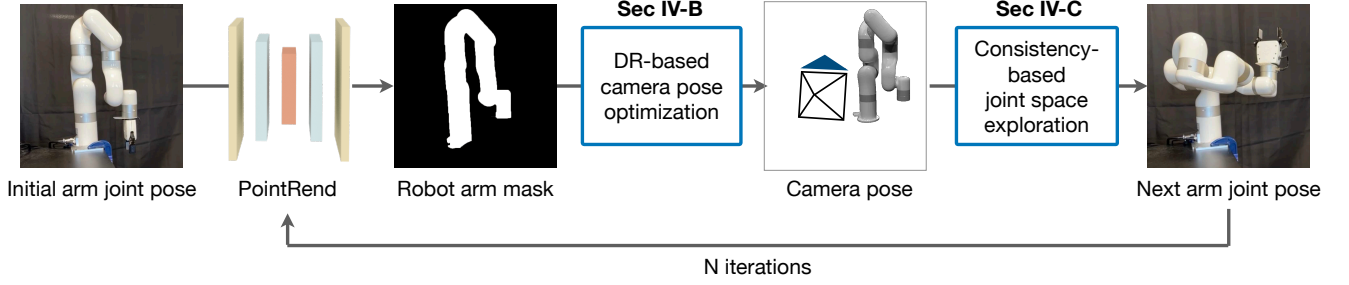
Fig. 2: **EasyHeC architecture.** The architecture of EasyHeC is composed of three main components that are executed iteratively. First, the RGB images are fed into a mask prediction network (PointRend) to acquire the observed robot arm mask. Second, a differentiable rendering (DR)-based camera pose optimization module is used to produce the camera pose by minimizing the difference between the rendered mask and the observed mask. Next, the robot arm is driven to the next arm joint pose generated by the consistency-based joint space exploration module. We then repeat the above steps for N iterations.

transformation between the camera c and the robot base b. In this paper, the transformation is represented as a 4×4 matrix $T_{cb} \in SE(3)$ that satisfies:

$$P^c = T_{cb}P^b, \qquad (1)$$

where P^c and P^b are the 3D points in the camera coordinate system and the robot base coordinate system, respectively.

IV. METHOD

A. Overview

As shown in Fig. 2, we optimize the camera pose in an iterative manner. In each iteration, given the image of the robot arm at a specific joint pose, we first predict the observed mask using PointRend [22]. Following this, we utilize differentiable rendering (Sec. IV-B) to refine the camera pose by minimizing the discrepancy between the rendered mask and the observed mask. Next, to optimize the T_{cb} more efficiently, we utilize the consistency-based joint space exploration module (Sec. IV-C) to select the most informative joint pose for the next iteration. The robot arm is then driven to this informative joint pose and the proposed method proceeds to the next iteration. Note that in later iterations, the camera pose is optimized with the masks at all the observed joint poses.

B. Differentiable rendering-based camera pose estimation

Different from the equation $AX = XB$ [3] that is widely used in marker-based hand-eye calibration, our approach employs a novel objective function that operates at the pixel level. Assuming that the 3D model of the robot arm is known, we can obtain the camera pose by minimizing the difference between the rendered robot arm and the camera observation, which is similar to the render-and-compare technique used in [20]. Unlike marker-based hand-eye calibration methods and regression-based methods, the pixel-wise objective function provides a direct connection between the camera pose and the observed image. This feature makes our approach considerably more diagnosable.

Specifically, we use the objective function to minimize the difference between the rendered mask and the observed mask:

$$L(T_{cb}) = (\min(1, \sum_l \pi(T_{cb}T_{bl}l)) - M)^2, \qquad (2)$$

where π is the differentiable mask renderer, l represents the 3D shape of a robot link, M is the mask of the robot, T_{cb} is the transformation between the robot base b and the camera c, and T_{bl} is the transformation between the link l and the robot base b, which can be obtained through forward kinematics with joint poses.

As the optimization target T_{cb} belongs to the $SE(3)$ space, we transform the transformation matrix to the Lie algebra space [23] and then perform the optimization using the following objective function:

$$L(\xi_{cb}) = (\min(1, \sum_l \pi(\exp(\xi_{cb})T_{bl}l)) - M)^2, \qquad (3)$$

The above loss function is fully differentiable, we could use the gradient descent method with PyTorch [24] Autograd to perform the optimization.

Mask prediction. To obtain the robot arm mask M as the supervision for the differentiable rendering-based optimization, we leverage the off-the-shelf learning-based segmentation network PointRend [22] to predict the mask from the RGB image. This obviates the need for additional manual annotation efforts, making the entire pipeline fully automated. Specifically, we first train the PointRend model on synthetic data generated by the SAPIEN [25] robot simulator and then fine-tune it on real data to enhance its generalization ability.

Camera pose initialization. There exist various methods for initializing the camera pose, such as traditional marker-based techniques [2, 3, 7, 8], learning-based methods [4, 12], or random initialization of the object pose within the frustum of the camera [16]. In our work, we adopt the PVNet [26] to perform the pose initialization. Given an RGB image of the robot arm with the initial joint pose, PVNet estimates

the pre-defined 2D keypoint locations (obtained with farthest point sampling on the 3D object model) through pixel-wise voting and subsequently solves the transformation using PnP [13]. By leveraging pixel-wise voting, PVNet can tackle the challenges of occlusion and truncation, which are often encountered in real-world hand-eye calibration scenarios. This strategy enables us to obtain an accurate pose initialization that serves as the starting point for the optimization process.

C. Consistency-based joint space exploration

In this subsection, a novel approach is presented that alternates between identifying the most informative supplementary joint pose for camera pose estimation and recalculating the camera pose by incorporating this new joint pose.

Recall that, our camera pose estimation is achieved by solving Eq. 3, which aims to align the mask of the projected 3D model of the arm with the mask of the observed arm. We can rewrite the objective function in Eq. 3 in a more simplified manner:

$$L(t; J) = \|\pi(J, t) - M_J\|^2, \quad (4)$$

where π is a mask renderer, J is a joint pose, t is a camera pose, M_J is the observed robot arm mask at joint pose J. This 2D matching constraint is often insufficient in constraining the solution space of the 6D camera pose given limited joint pose observations. For example, accurately estimating the camera's distance to the robot base can be challenging with only one joint pose observation. In other words, there might be a set of reasonable camera pose candidates T that satisfies $L(t, J) < \epsilon$, where ϵ is a small value.

The goal is to shrink T as much as possible by obtaining a new $T' \subseteq T$ as the solution set of $L(t; J') < \epsilon$ with a new joint pose J'. The next question is, how to choose J' so that T can be shrunk as soon as possible? Equivalently, how to choose J' so that $L(t, J') > \epsilon$ for as many $t \in T$ as possible?

Our approach is to sample a representative set of $\{t_i\}_{i=1}^K \in T$, and based upon which, we use $\sum_i L(t_i, J')$ as the surrogate to help us to choose J'. In fact, it can be proved that:

$$\sum_i L(t_i, J') \geq \text{Var}(m_1, m_2, \ldots, m_K), \quad (5)$$

where $m_i = \pi(J', t_i)$. This inequality then tells us that, if we choose J' to maximize the variance of the masks, we can shrink T as soon as possible.

Based on this observation, we develop a joint space exploration strategy that operates solely in simulation, allowing us to generate informative joint poses before executing motion commands on the robot. The algorithm is summarized in Alg.1, where we first sample a large number of joint poses in the simulator and then evaluate the variance of the rendered masks over the camera pose candidates. Then, we move the robot to the joint pose with the highest variance and perform the next iteration of DR-based optimization. Unlike traditional marker-based methods that necessitate the manual

design of joint poses, our approach is fully automatic and requires no human intervention.

In the subsequent DR-based optimization iterations, we include all previously observed masks rather than using only the most recent observation. The final optimization objective is defined as follows:

$$L(\xi_{cb}) = \frac{1}{K} \sum_{i=1}^K (\min(1, \sum_l \pi(\exp(\xi_{cb})T_{bl,i}l)) - M_i)^2, \quad (6)$$

where $K = \{1, 2, \ldots, N\}$ is the number of images, $T_{bl,i}$ is the transformation between the link l and the robot base b at the i-th joint pose, and M_i is the mask of the robot in the i-th image.

Algorithm 1: Joint space exploration.

Input: Camera pose candidates $T = \{t_i\}_{i=1}^K$;
Joint pose sampling number N_s;
Output: The next joint pose J'
1 Randomly sample N_s joint poses $J_1, J_2, \ldots, J_{N_s}$.
2 **for** each joint pose J_i **do**
3   **for** each camera pose t_k **do**
4   // Render the robot mask with the joint pose J_i and the camera pose t_k.
5         $m_{ij} = \pi(J_i, t_k)$
6     $V_i = \text{Var}(m_{i1}, m_{i2}, \ldots, m_{iK})$
7 $J' = J_{argmax(V)}$

D. Implementation details

In our implementation, we employ nvdiffrast [27] as the differentiable mask renderer. During the optimization process, we utilize the Adam optimizer with a learning rate of 3e-3 and perform 1000 steps.

For the space exploration module, we randomly choose 50 camera poses from the range between step 200 and 1000 as camera pose candidates. In practice, the number of sampled camera pose candidates and the range may vary depending on the number of optimization iterations. Next, we randomly sample 2000 joint poses and filter out any invalid poses that result in self-intersection of the robot arm, links exceeding a pre-defined distance threshold from the robot arm base, or collisions with the surrounding environment. Subsequently, we identify the joint pose with the largest variance as the next joint pose to explore.

V. EXPERIMENTS

A. Evaluation on synthetic dataset

Data generation. In our first experiment, we use SAPIEN [25] to synthesize 100 photo-realistic scenes with different camera poses using a xArm robot arm and compare our method with the traditional marker-based method. We use 20 manually-designed joint poses for each scene for
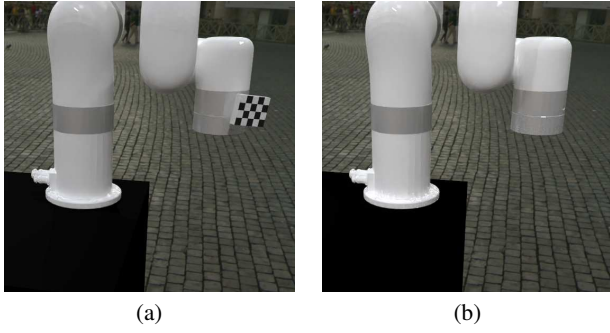
(a)   (b)

Fig. 3: **Example images in the xArm synthetic dataset.** (a) is for the marker-based method and (b) is for our method.
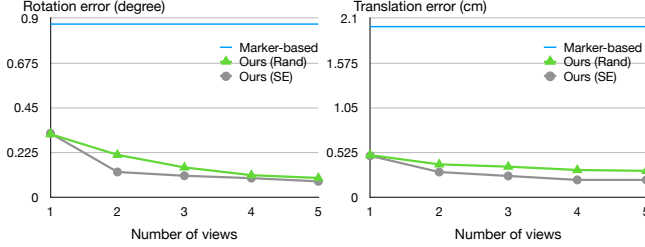

Fig. 4: **Pose estimation results on the xArm synthetic dataset.** We report the rotation error and translation error in degrees and centimeters, respectively. The number of views is used only for our methods but not for the marker-based method. Ours (SE) and Ours (Rand) represent our method with the next joint pose generation using consistency-based space exploration and random sampling, respectively.

the marker-based method. Our method only required the image with the initial joint pose and the subsequent views are generated online. An example of the synthetic dataset is shown in Fig. 3, where we attach a chessboard marker on the robot arm for the marker-based method. The size of the marker is 4cm×5cm, with each grid being 1cm×1cm. We rendered 10,000 images for the PointRend [22] and another 10,000 images for the PVNet [26] training to obtain the observed segmentation mask and initial camera pose for our method.

Evaluation of hand-eye calibration. We evaluate both methods based on the translation error and rotation error between the ground-truth camera poses and the solved camera poses. Note that the manually-designed joint poses do not always ensure that the marker is visible in all scenes, especially if there is occlusion. To ensure fairness, we also consider randomly sampling a large number of joint poses for each scene and selecting only those where the marker is visible. However, we find that only 1.9% of the sampled joint poses result in clear marker visibility, which makes this approach impractical. Thus, we use the same joint poses for all scenes but only analyze those scenes where more than 6 markers are detected. The results are shown in Fig. 4, Tab. I, and Tab. II, where our method outperforms the

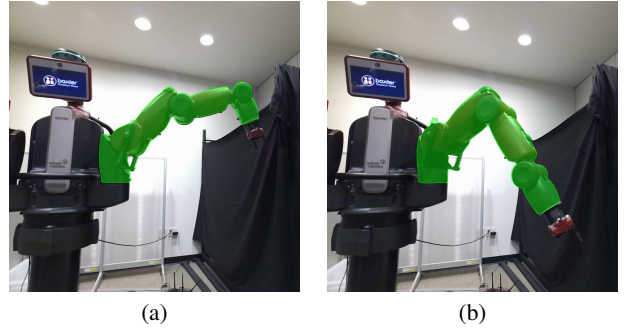
(a)   (b)

Fig. 5: **Qualitative results on the Baxter dataset.** We render the links of the Baxter robot arm with the solved camera poses and hover the masks on the RGB images.

marker-based method [3] by a large margin. Specifically, the average rotation and translation errors of the marker-based method are 0.9 degrees and 2.1 cm, respectively. For the proposed method, the evaluation is performed using different numbers of views ranging from 1 to 5. The errors consistently decrease with an increase in the number of views. Specifically, when there is only 1 view, the error of our method is 0.32 degrees and 0.5 cm, whereas, with 5 views, the error decreases to 0.08 degrees and 0.2 cm.

Method	1	2	3	4	5
Marker-based	0.87				
Ours (Rand)	0.318	0.214	0.151	0.112	0.098
Ours (SE)	0.322	0.128	0.109	0.097	0.081

TABLE I: **Rotation error evaluation results on the xArm synthetic dataset.** For our method, we report errors in degrees with the number of views from 1 to 5. Ours (SE) and Ours (Rand) represent our method with the next joint configuration generation using consistency-based space exploration and random sampling, respectively.

Method	1	2	3	4	5
Marker-based	2.00				
Ours (Rand)	0.496	0.388	0.361	0.322	0.312
Ours (SE)	0.488	0.298	0.252	0.206	0.206

TABLE II: **Translation error evaluation on the xArm synthetic dataset.** We report errors in centimeters with the number of views from 1 to 5.

B. Evaluation on Real-world Baxter dataset

Dataset introduction. In this section, we conduct experiments on the real-world Baxter dataset [28] and compare the efficacy of our method against state-of-the-art learning-based methods. The Baxter dataset contains 100 images of a real Baxter robot, with a single camera pose and 20 distinct joint poses. In light of the single camera pose, we manually initialize the camera pose for simplicity.

Implementation details and evaluation metrics. To train the segmentation network, we train the PointRend first on the synthetic DREAM dataset [12] with domain randomization and then fine-tune on some realistic images rendered with SAPIEN [25]. The experimental results are presented in Tab.III and Tab.IV, where the 2D and 3D percentage of correct keypoints (PCK) are reported. We also provide some visualizations of the results in Fig. 5. In addition to the thresholds employed in [16, 28], we report results with smaller thresholds to further elucidate the accuracy of our method since our method nearly reaches 100% at the thresholds used in previous works.

Following the conventions in [16, 28], we present the evaluation results on the training images in Tab. III and Tab. IV, while in Fig. 7, we present the results evaluated on the entire dataset since it is more representative to evaluate the performance at other joint poses.

Method	20px	30px	40px	50px	100px	150px	200px
DREAM [12]	0.16	0.23	0.29	0.33	0.52	0.62	0.64
OK [28]	0.34	0.54	0.66	0.69	0.88	0.93	0.95
IPE [16] (box)	-	-	-	0.65	0.94	0.95	0.95
IPE [16] (cylinder)	-	-	-	0.80	0.91	0.93	0.95
IPE [16] (CAD)	-	-	-	0.74	0.90	0.94	**1.00**
Ours (1view)	0.35	0.55	0.75	0.90	0.95	0.95	**1.00**
Ours (2views)	0.40	0.75	0.95	**1.00**	**1.00**	**1.00**	**1.00**
Ours (3views)	**0.55**	**0.85**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**

TABLE III: **2D PCK evaluation results on the Baxter dataset.**

Method	2cm	5cm	10cm	20cm	30cm	40cm
DREAM [12]	0.01	0.08	0.32	0.43	0.54	0.66
OK [28]	0.10	0.34	0.54	0.66	0.69	0.88
IPE [16] (box)	-	-	0.8	0.95	0.95	0.95
IPE [16] (cylinder)	-	-	0.71	0.93	0.94	0.95
IPE [16] (CAD)	-	-	0.78	0.93	0.97	**1.00**
Ours (1view)	0.10	0.65	0.90	**1.00**	**1.00**	**1.00**
Ours (2views)	**0.15**	**0.80**	**0.95**	**1.00**	**1.00**	**1.00**
Ours (3views)	**0.15**	**0.80**	0.90	**1.00**	**1.00**	**1.00**

TABLE IV: **3D PCK evaluation results on the Baxter dataset.**

Evaluation of hand-eye calibration. For our method, we perform evaluations with 1, 2, and 3 views. The results demonstrate that our method outperforms the previous state-of-the-art methods by a significant margin. It is worth noting that the thresholds of all methods are relatively large. This can be attributed to two factors. First, the ground-truth positions of the end-effector are collected by detecting a marker attached to it, which is unreliable due to the marker's small size and the significant distance between the marker and the camera. Second, the provided joint poses are not reliable, as evidenced by inconsistencies observed in images with identical joint poses. Nonetheless, our method delivers the most accurate results on this dataset, demonstrating the effectiveness and robustness of our method.
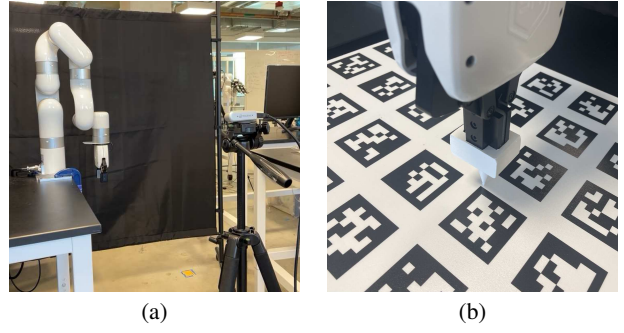

(a)　　　　　　　　(b)

Fig. 6: **Real-world high-precision targeting experiment using a xArm robot arm.** (a) shows the setup of the camera and the robot arm, and (b) shows the real-world experiment where a pointer is used to tip the corners of an Aruco marker board.
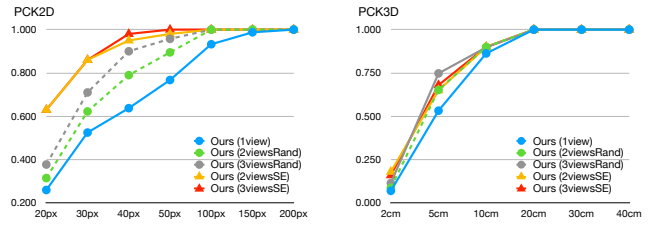

Fig. 7: **Ablation study on the Baxter dataset.** We report the PCK2D (left) and PCK3D (right) on the Baxter dataset with different thresholds. Ours (SE) and Ours (Rand) represent our method with the next joint pose generation using consistency-based space exploration and random sampling, respectively.

C. Evaluation on the real-world xArm data

To assess the effectiveness of our proposed hand-eye calibration method in real-world scenarios, we conduct a high-precision targeting experiment using an Aruco marker board and a pointer attached to the robot's end effector.

We detect the corners of the marker board using OpenCV [29] and transform their positions to the robot arm base coordinate system via our hand-eye calibration results. Next, we employ the pointer to tip the marker board corners and manually measure the distance between the pointer and the actual corner position. A representative illustration is visualized in Fig. 6. Our results exhibit that the average error in our hand-eye calibration system is roughly 0.4cm. Although this outcome does not manifest significant precision, it offers a preliminary indication of the system's accuracy and convincingly demonstrates the effectiveness of our proposed approach.

D. Ablation studies

Joint space exploration. To demonstrate the effectiveness of the joint space exploration module, we conduct an ablation study by comparing it with the random selection approach. The experiments were performed on the xArm synthetic dataset and the Baxter dataset. For the xArm synthetic
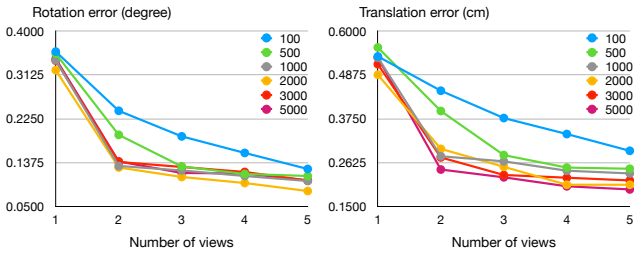
Fig. 8: **Pose estimation results on the xArm synthetic dataset with different numbers of sampled joint poses in the joint space exploration process.** We report the rotation (left) and translation (right) errors in degrees and centimeters, respectively.
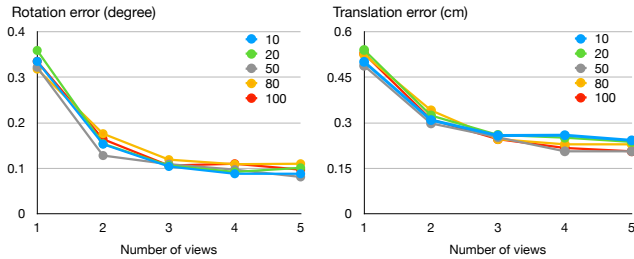

Fig. 9: **Pose estimation results on the xArm synthetic dataset with different numbers of sampled camera pose candidates in the joint space exploration process.** We report the rotation (left) and translation (right) errors in degrees and centimeters, respectively.

dataset, the joint pose selection is from the sampled joint pose, whereas for the Baxter dataset, the joint pose selection is from the joint poses provided by the dataset. The results are shown in Fig. 4 and Fig. 7, respectively. The results on the xArm synthetic dataset show that using space exploration to select the next joint pose consistently provides faster convergence speed and produces smaller errors than random selection. The results on the Baxter dataset show that the space exploration-based approach produces higher PCK than the random selection approach, especially for the PCK2D.

Different number of sampled joint poses. To assess the influence of varying numbers of joint poses on our method's performance in the joint space exploration process, we conducted experiments on the xArm synthetic dataset by sampling different numbers of joint poses. The outcomes of these experiments are presented in Fig. 8, where it is demonstrated that the errors decrease as the number of sampled joint poses increases, particularly when the number of joint poses is below 1000. As soon as the number of joint poses exceeds 1000, however, the errors become virtually unchanged. This occurs since the number of sampled joint poses is already extensive enough to cover the entire joint space. Therefore, we set the number of sampled joint poses to 2000 in other experiments.

Different number of sampled camera candidates. To verify the influence of different numbers of sampled camera
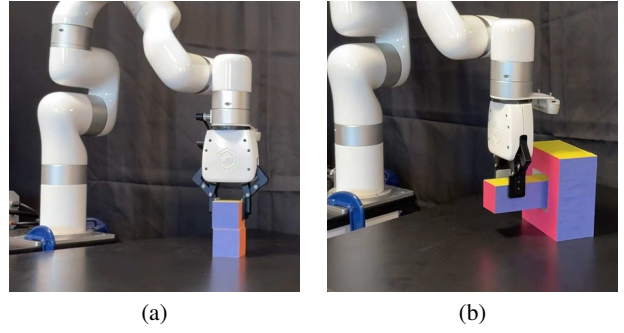

(a)   (b)

Fig. 10: **Real-world applications powered by our method:** screenshots from stacking cube (a) and peg insertion (b) experiments, where object poses in the camera coordinate system are transformed to the robot coordinate system using our hand-eye calibration result. See the supplementary video for the whole process.

candidates in the joint space exploration process on the performance of our method, we conduct an ablation study on the xArm synthetic dataset by varying the number of sampled camera candidates. The results are shown in Fig. 9, which shows that the errors are roughly the same as the number of sampled camera candidates varying. This indicates that the number of sampled camera candidates does not affect the performance of our method, thus we can use a small number of camera pose candidates in practice to save computation time. In other experiments, we set the number of sampled camera candidates to 50.

E. Real-world applications

In this section, we present some real-world applications to exemplify the efficacy of our proposed approach. Suppose we intend to employ a state-based policy with solely visual inputs, in that case, it becomes imperative to convert object states from the camera coordinate system to the robot base coordinate system. In this case, the proposed EasyHeC could be leveraged to perform this transformation. In our experiment, we choose CoTPC [30] as the state-based algorithm, which takes states as input and outputs the next robot action. We perform two experiments using the CoTPC algorithm: stacking cube and peg insertion. Specifically, we use PVNet [26] to estimate the object poses in the camera coordinate system and then use the hand-eye calibration results to transform the object poses from the camera coordinate system to the base coordinate system as the input states to the CoTPC network. Fig.10 shows the screenshots captured during the execution process, which demonstrate that the cubes are accurately aligned and the peg is successfully inserted, thereby affirming the effectiveness of our method in state-based algorithms.

VI. LIMITATION

There are several limitations to our work. **(1)** Our methodology necessitates a precise robot model, which may not enjoy universal applicability due to the presence of protective shells enveloping certain robots. Nonetheless, the

object shape of the robot links can be effectively optimized without ambiguity in the context of multiview calibration. Alternatively, one can employ advanced techniques such as [31] to reconstruct high-fidelity link shapes before using our methodology. **(2)** Our methodology exclusively caters to the eye-to-hand calibration scenario. However, for eye-in-hand configurations, the same pipeline can be leveraged for calibration. Although the restricted field of view for the camera in this scenario may introduce inaccuracies when using the PVNet [26] for initialization, the distribution range of the initial camera pose will be greatly constrained for eye-in-hand configurations, enabling manual initialization or the selection of one of several predefined camera poses for initialization.

VII. CONCLUSIONS

In this paper, we have presented a novel approach to hand-eye calibration. Our method leverages the differentiable rendering technique to optimize the camera pose by minimizing the difference between the rendered mask and the observed mask. The joint space exploration strategy is employed to find the informative joint poses for the camera pose optimization, which reduces the number of samples needed for achieving accurate results. Our experimental results demonstrate that our approach outperforms traditional methods and several state-of-the-art methods on both synthetic and real-world datasets. Overall, our approach provides a practical solution for camera pose estimation in robotics and can be applied to a wide range of applications, such as object manipulation, grasping, and pick-and-place tasks. We believe that our work can reduce the sim-to-real gap and contribute to the development of more efficient and accurate robotic systems.

REFERENCES

[1] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[2] J. Ilonen and V. Kyrki, "Robust robot-camera calibration," in *2011 15th International Conference on Advanced Robotics (ICAR)*. IEEE, 2011, pp. 67–74.

[3] 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*, vol. 5, no. 3, pp. 345–358, 1989.

[4] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Single-view robot pose and joint angle estimation via render & compare," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1654–1663.

[5] B. C. Sefercik and B. Akgun, "Learning markerless robot-depth camera calibration and end-effector pose estimation," in *Conference on Robot Learning*. PMLR, 2023, pp. 1586–1595.

[6] O. Bahadir, J. P. Siebert, and G. Aragon-Camarasa, "A deep learning-based hand-eye calibration approach using a single reference point on a robot manipulator," in *2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2022, pp. 1109–1114.

[7] F. C. Park and B. J. Martin, "Robot sensor calibration: solving ax=xb on the euclidean group," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, pp. 717–721, 1994.

[8] K. Daniilidis, "Hand-eye calibration using dual quaternions," *The International Journal of Robotics Research*, vol. 18, no. 3, pp. 286–298, 1999.

[9] N. Andreff, R. Horaud, and B. Espiau, "Robot hand-eye calibration using structure-from-motion," *The International Journal of Robotics Research*, vol. 20, no. 3, pp. 228–248, 2001.

[10] J. Heller, M. Havlena, A. Sugimoto, and T. Pajdla, "Structure-from-motion based hand-eye calibration using l_∞ minimization," in *CVPR 2011*. IEEE, 2011, pp. 3497–3503.

[11] X. Zhi and S. Schwertfeger, "Simultaneous hand-eye calibration and reconstruction," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1470–1477.

[12] 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)*. IEEE, 2020, pp. 9426–9432.

[13] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, pp. 155–166, 2009.

[14] E. Valassakis, K. Dreczkowski, and E. Johns, "Learning eye-in-hand camera calibration from a single image," in *Conference on Robot Learning*. PMLR, 2022, pp. 1336–1346.

[15] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. Spie, 1992, pp. 586–606.

[16] J. Lu, F. Liu, C. Girerd, and M. C. Yip, "Image-based pose estimation and shape reconstruction for robot manipulators and soft, continuum robots via differentiable rendering," *arXiv preprint arXiv:2302.14039*, 2023.

[17] H. Kato, Y. Ushiku, and T. Harada, "Neural 3d mesh renderer," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3907–3916.

[18] S. Liu, W. Chen, T. Li, and H. Li, "Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction," *arXiv preprint arXiv:1901.05567*, 2019.

[19] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3504–3515.

[20] A. Kundu, Y. Li, and J. M. Rehg, "3d-rcnn: Instance-level 3d object reconstruction via render-and-compare," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3559–3568.

[21] Y. Labbé, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic, "Megapose: 6d pose estimation of novel objects via render & compare," *arXiv preprint arXiv:2212.06870*, 2022.

[22] A. Kirillov, Y. Wu, K. He, and R. Girshick, "Pointrend: Image segmentation as rendering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9799–9808.

[23] D. Gabay, "Minimizing a differentiable function over a differential manifold," *Journal of Optimization Theory and Applications*, vol. 37, pp. 177–219, 1982.

[24] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[25] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "SAPIEN: A simulated part-based interactive environment," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[26] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4561–4570.

[27] S. Laine, J. Hellsten, T. Karras, Y. Seol, J. Lehtinen, and T. Aila, "Modular primitives for high-performance differentiable rendering," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–14, 2020.

[28] J. Lu, F. Richter, and M. C. Yip, "Pose estimation for robot manipulators via keypoint optimization and sim-to-real transfer," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4622–4629, 2022.

[29] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.

[30] Z. Jia, F. Liu, V. Thumuluri, L. Chen, Z. Huang, and H. Su, "Chain-of-thought predictive control with behavior cloning," in *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*.

[31] K. Yang, X. Zhang, Z. Huang, X. Chen, Z. Xu, and H. Su, "Movingparts: Motion-based 3d part discovery in dynamic radiance field," *arXiv preprint arXiv:2303.05703*, 2023.