Foundation Feature-Driven Online End-Effector Pose Estimation: A Marker-Free and Learning-Free Approach

Tianshu Wu*, Jiyao Zhang*, Shiqian Liang, Zhengxiao Han, Hao Dong

***Abstract*— Accurate transformation estimation between camera space and robot space is essential. Traditional methods using markers for hand-eye calibration require offline image collection, limiting their suitability for online self-calibration. Recent learning-based robot pose estimation methods, while advancing online calibration, struggle with cross-robot generalization and require the robot to be fully visible. This work proposes a Foundation feature-driven online End-Effector Pose Estimation (FEEPE) algorithm, characterized by its training-free and cross end-effector generalization capabilities. Inspired by the zero-shot generalization capabilities of foundation models, FEEPE leverages pre-trained visual features to estimate 2D-3D correspondences derived from CAD models and reference images, enabling 6D pose estimation via the PnP algorithm. To resolve ambiguities from partial observations and symmetry, a multi-historical key frame enhanced pose optimization algorithm is introduced, utilizing temporal information for improved accuracy. Compared to traditional hand-eye calibration, FEEPE enables marker-free online calibration. Unlike robot pose estimation, it generalizes across robots and end-effectors in a training-free manner. Extensive experiments demonstrate its superior flexibility, generalization, and performance. Additional demonstrations are available at `https://feepose.github.io/`**

I. INTRODUCTION

Consider a robot performing a task where perception results are obtained in the camera space. How can the robot execute actions based on these perception results? This requires an accurate transformation between the camera space and the robot space. Traditional methods employ augmented reality (AR) tags [1], [2], [3] as markers attached to the end-effector and solve a homogeneous matrix equation [4] to determine the transformation. However, this approach requires an offline collection of images of the robotic arm in different states for optimization, making it unsuitable for online robot self-calibration [5], [6]. This limitation restricts the rapid deployment of robotic systems.

Recent advancements in learning-based robot pose estimation algorithms [7], [8], [9] have shown promise for enabling online self-calibration. These algorithms [10], [11] aim to use data-driven methods to estimate a robot's pose from a single image or a sequence of images. However, these methods have notable limitations: they cannot generalize across different robots and require the robot to be fully visible in the image. These constraints significantly limit the applicability of such methods. Our goal is to develop an **online**, **marker-free**, **highly generalizable** and **training-free** end-effector pose estimation algorithm for robot self-calibration, which presents the following challenges:

1) End-effector diversity in appearance and geometry poses a challenge for cross-end-effector generalization.
2) The ambiguity issue caused by partial observations and end-effector symmetry.

To address these, we propose **FEEPE**, a **F**oundation feature-driven online **E**nd-**E**ffector **P**ose **E**stimation algorithm. Inspired by the zero-shot generalization ability [12], [13] of visual foundation models [14] like DINOv2 [15], the algorithm leverages pre-trained visual features for end-effector 6D pose estimation, effectively handling the appearances and geometry variation. Specifically, with the CAD model of the end-effector known, multiple reference images from different perspectives are pre-rendered first. The DINOv2 algorithm then extracts features from these images and the target image to establish 2D-2D correspondences. These correspondences facilitate the creation of 2D-3D mappings, linking the target image to the 3D model. This facilitates an initial 6D pose estimation using the PnP algorithm [16].

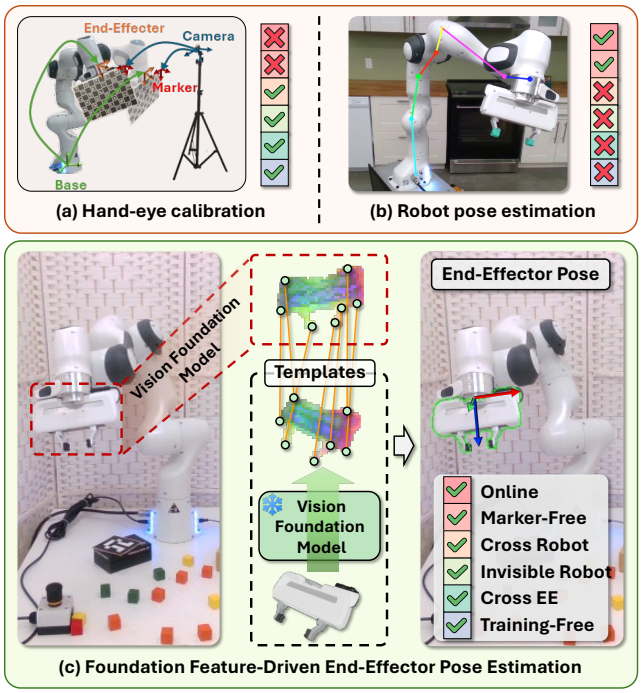

Fig. 1: We introduce **F**oundation feature-driven online **E**nd-**E**ffector **P**ose **E**stimation (**FEEPE**). The proposed FEEPE achieves online, marker-free, training-free end-effector pose estimation with cross-robot and cross-end-effector generalization without requiring the robot to be fully visible.

All authors are with CFCS, School of CS, Peking University and National Key Laboratory for Multimedia Information Processing.
* indicates equal contribution
Corresponding to hao.dong@pku.edu.cn

Nevertheless, the ambiguity from partial observations and the end-effector's symmetry can lead to inaccuracies when predicting from a single image. To address this, we introduce a multi-historical key frame enhanced pose optimization that utilizes temporal information and robot priors to resolve symmetry ambiguities and enhance accuracy.

Overall, as shown in Figure 1, our approach enables marker-free online calibration compared to traditional hand-eye calibration. In contrast to learning-based robot pose estimation, our method achieves cross-robot and cross-end-effector generalization in a training-free manner and does not require the robot arm to be visible. To the best of our knowledge, this is the first online, marker-free, generalizable, and training-free end-effector pose estimation algorithm for robot self-calibration. Extensive experimental validation demonstrates the convenience, robustness, and high precision (1mm) of our approach, whether compared to learning-based methods or traditional hand-eye calibration techniques.

II. RELATED WORK

A. Camera-to-Robot Pose Estimation.

Traditionally, camera-to-robot pose is estimated by hand-eye calibration [4], using markers like ARTag [2] or AprilTag [3]. Recently, learning-based methods [8], [17] have emerged, utilizing deep neural networks for online calibration. DREAM [7] leverages a CNN to detect pre-defined robot keypoints and solve the camera-to-robot pose using a PnP solver. CtRNet [10] further improves the performance by utilizing a differentiable renderer and a segmentation objective. RoboKeyGen [11] uses a diffusion model to lift 2D keypoints into 3D, jointly estimating robot joint angles and camera pose. RoboPose [8] adopts a render-and-compare approach. Current methods have taken a step towards online calibration, but they require the full robot to be visible and are robot-specific, whereas our method generalizes to unseen robots. There have also been works aiming to perform online calibration by end-effector pose estimation. [18] directly regresses the end-effector pose from a pointcloud. [19] predict 3D positions of the end-effector and solve for the effector pose, then calibrate using multiple estimations. Such methods are all end-effector specific, while our method can generalize to unseen end-effectors without training.

B. CAD Model-based Object Pose Estimation.

Early methods of object pose estimation [20], [21] estimate the 6D pose of a known object. These methods are instance-level [22], [23], meaning that the test object is seen in training [24], [25], [26], and the method is unable to generalize to unseen objects. To relax this constraint, recent efforts aim to estimate the unseen object pose with the textured CAD model known. [27] proposed the challenge of novel object pose estimation, providing a method that established correspondences between the object pointcloud and the scene pointcloud. [28], [29], [30] follow this path, seeking to directly extract object-agnostic features [31] from the CAD model and match them with features of the scene to obtain 3D-3D matches. Other works [32], [33], [34] use a render-and-compare approach, iteratively refining a coarse estimate by rendering the object in different poses and comparing it with the target image. Template-based methods [35], [36] render templates [37], [38], and retrieve the nearest template during test time [39], [40], and perform further refinement or optimization. While progressing towards estimating the unseen object pose, all current approaches struggle with ambiguity caused by partial observations and symmetry. Our approach enhances pose estimation by addressing symmetry ambiguities and improving accuracy through the integration of temporal information and robot priors.

III. METHOD

Problem Formulation. We assume the 3D model of end-effector known and take a sequence of observations $\{\boldsymbol{O}_i\}_{i=1}^{t}$, where $\boldsymbol{O} = \{\boldsymbol{I}_c, \boldsymbol{I}_d, \boldsymbol{s}\}$. Here, $\boldsymbol{I}_c$ represents the RGB image, $\boldsymbol{I}_d$ represents the depth image, and $\boldsymbol{s}$ represents the states of the robotic arm. The goal is to predict the pose of the end-effector at time t, denoted as $\boldsymbol{p}_t \in SE(3)$.

Overview. Figure 2 illustrates our pipeline. With the 3D model known, we first use rendered images as references and employ foundation features to establish 2D-3D matches between the reference and target images for pose estimation, as detailed in Section III-A. Then, to address ambiguities arising from partial observations and end-effector symmetry, we incorporated a multi-historical key frame enhanced pose optimization algorithm, as detailed in Section III-B.

A. Foundation Feature-Driven 2D-3D Matching

This section elaborates on the procedures for generating reference views, establishing 2D-3D correspondences, and estimating initial pose candidates. We first render K_t template images from the 3D model of the end-effector. We then extract foundation features using DINOv2 [15] and identify the top K_r reference views that exhibit the highest similarity to the target image. Based on these features, we establish 2D-3D matches to compute pose candidates with the Perspective-n-Point (PnP) algorithm [16]. In this manner, for every target image, we obtain $\mathcal{S} = \{\hat{\mathcal{M}}_i, \boldsymbol{p}_i\}_{i=1}^{K_r}$, where $\hat{\mathcal{M}}_i$ represents the inlier 2D-3D matches and $\boldsymbol{p}_i \in SE(3)$ denotes the pose candidates. The index $i \in \{0, 1, ..., K_r\}$ represents each reference view. In the following, we will delineate the specific details.

1) Templates generation and reference views selection: We employ Fibonacci Sphere [41] to sample 80 positions on a unit sphere. For each viewpoint, we uniformly sample 12 in-plane rotations, resulting in 960 sampled viewpoints. Using Blender, we render the templates with ray tracing and extract pixel-level visual features using DINOv2-ViTL [15], denoted as Ψ. The visual features $\mathcal{F}$ for each template image are computed as $\mathcal{F} = \Psi(\boldsymbol{I}_c)$, where $\boldsymbol{I}_c$ represents the RGB images. This process results in the template set $\{\boldsymbol{I}_i^c, \boldsymbol{I}_i^d, \mathcal{F}_i\}_{i=1}^{K_t}$, where $\boldsymbol{I}_i^c$ represents the RGB images, $\boldsymbol{I}_i^d$ represents the depth images, $\mathcal{F}_i$ represents the visual features and $K_t = 960$. Further, we construct Bag-of-Words descriptors (BoW) using the template features $\{\mathcal{F}_i\}_{i=1}^{K_t}$, following [39]. Based on these descriptors, we select the top K_r views that are most similar to the target image as

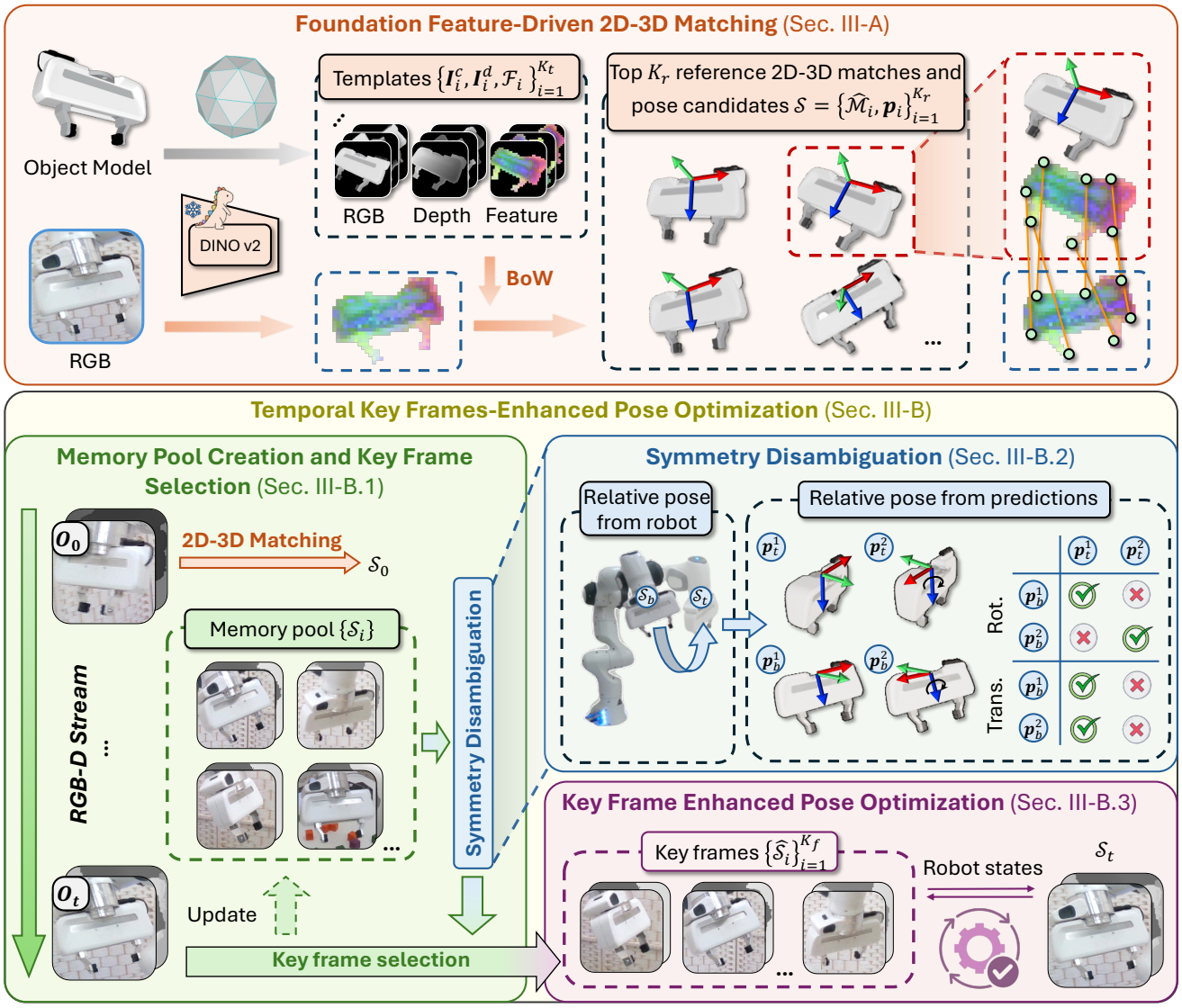

Fig. 2: Overview of our method. Given the 3D model of the end-effector and a target image, we first render templates from various viewpoints. Using foundation features, we find the top K_r references most similar to the target image and compute 2D-3D matches and pose candidates (Section III-A). To address ambiguities from partial observations, we introduce a global memory pool (Section III-B.1) that records key frames and robot states for pose optimization (Section III-B.3). To resolve ambiguities from symmetry, we propose a symmetry disambiguation module (Section III-B.2) to eliminate incorrect matches.

reference views. In this way, for each target image, we can identify K_r reference images for subsequent calculations. In this paper, we set the number of reference images K_r to 5.

2) 2D-3D matching and pose candidate estimation: Although we identified reference views from the templates that closely match the target image using Bag-of-Words descriptors, the discrete sampling within the templates still results in significant pose errors between the reference views and the target image. To address this issue, we utilize foundation features to establish pixel-level correspondences between the reference views and the target image. In this way, for each reference view, we can further establish 2D-3D matches between the target image and the 3D model, denoted as $\mathcal{M} = \{\boldsymbol{u} \leftrightarrow \boldsymbol{X}\}$, where $\boldsymbol{u}$ and $\boldsymbol{X}$ represent 2D image points and 3D model points, respectively. Specifically, we first compute 2D-2D matches by the cosine similarity between $\mathcal{F}_{ref}$ and $\mathcal{F}_{tar}$, where $\mathcal{F}_{ref}$ indicates the pixel-level features of the reference views and $\mathcal{F}_{tar}$ indicates the pixel-level features of the target image. By incorporating the rendered depth maps, for each reference view, we convert these 2D-2D matches into 2D-3D matches $\mathcal{M}$ and recover the pose candidates $\boldsymbol{p}$ with PnP. To remove mismatches in $\mathcal{M}$, we eliminate outliers in the PnP solving process, resulting in refined matches $\hat{\mathcal{M}}$. Through this way, for each target frame, we obtain the results $\mathcal{S} = \{\hat{\mathcal{M}}_i, \boldsymbol{p}_i\}_{i=1}^{K_r}$.

B. Multi Historical Key Frame Enhanced Pose Optimization

In the preceding discussion, we established the formation of 2D-3D matches and corresponding pose candidates $\mathcal{S} = \{\mathcal{M}_i, \boldsymbol{p}_i\}_{i=1}^{K_r}$ between target frame and reference images. However, estimating pose from a single frame introduces ambiguities due to partial observations and symmetry. To

address these, this section introduces the temporal data and robotic priors enhanced pose optimization. Specifically, Section III-B.1 discuss the memory pool creation and key frame selection, Section III-B.2 introduce the symmetry disambiguation with temporal information and robotic priors, and Section III-B.3 propose a key frame-enhanced joint pose optimization approach for accurate pose estimation.

1) Memory Pool Creation and Key Frame Selection: To enhance the accuracy and robustness of pose estimation, we maintain a global memory pool $\{\mathcal{S}_i\}$, where $\mathcal{S} = \{\hat{\mathcal{M}}_i, \boldsymbol{p}_i\}_{i=1}^{K_r}$ as described in Section III-A. The memory pool is used to store historical data for subsequent optimization (Section III-B.3). To ensure that only significantly different poses are retained, we define a criterion based on the angular distance. The angular distance function Ω is defined as follows:

$$\Omega(\boldsymbol{p}_1, \boldsymbol{p}_2) = \arccos\left(\frac{tr(\boldsymbol{p}_1^T \boldsymbol{p}_2) - 1}{2}\right) \tag{1}$$

where $\boldsymbol{p}_1, \boldsymbol{p}_2 \in SO(3)$. We implement the updating criterion for the memory pool following [42]: a new frame $\boldsymbol{O}_t$ is added only if the angular distance between the estimated $\boldsymbol{p}_t$ and the closest existing poses $\{\boldsymbol{p}\}$ in the memory pool satisfies:

$$\min_{\boldsymbol{p}_j \in \{\boldsymbol{p}\}} \Omega(\boldsymbol{p}_j, \boldsymbol{p}_t) > \theta \tag{2}$$

where θ indicates a predefined threshold, which set to 10° in this paper. To balance computational efficiency and estimation accuracy in pose estimation, we select key frames from the memory pool for subsequent optimization. Initially, we estimate the pose $\hat{\boldsymbol{p}}_t$ for the new frame $\boldsymbol{O}_t$ using the end-effector pose $\boldsymbol{p}_{t-1}$ of $\boldsymbol{O}_{t-1}$ and the transformation $\delta_{\boldsymbol{p}}$ derived from the forward kinematics of robot:

$$\hat{\boldsymbol{p}}_t = \boldsymbol{p}_{t-1} \cdot \delta_{\boldsymbol{p}} \tag{3}$$

We utilize the Farthest Point Sampling (FPS) [43] method to select key frames from the memory pool, starting from the estimated pose $\hat{\boldsymbol{p}}_t$. This method aims to effectively cover the pose space by maximizing the angular distances between selected frames. The objective for selecting key frames is succinctly formulated as:

$$\mathcal{P}^* = \underset{\mathcal{P} \subset \mathcal{P}_m, |\mathcal{P}| = K_f}{\arg\max} \min_{\boldsymbol{p}_i, \boldsymbol{p}_j \in \mathcal{P}, i \neq j} \Omega(\boldsymbol{p}_i, \boldsymbol{p}_j) \tag{4}$$

where $\mathcal{P}_m$ is the set of poses corresponding to each frame in the memory pool $\{\mathcal{S}\}$, and $\mathcal{P}$ is the subset of chosen from $\mathcal{P}_m$, with K_f being the desired number of frames. This ensures the diversity of selected frames, thereby enhancing the robustness and efficiency of the pose estimation process.

2) Symmetry Disambiguation: While the memory pool provides valuable historical data for pose estimation, the presence of symmetric end-effectors poses significant challenges. Incorrect symmetric predictions can greatly affect optimization processes. To mitigate these issues, we introduce a *symmetry disambiguation* module with temporal information and robot priors. Specifically, we select a base frame $\mathcal{S}_b$ from the memory pool, which exhibits a bimodal distribution due to symmetry, as $\{\boldsymbol{p}_b^1, \boldsymbol{p}_b^2\}$. For the current frame $\mathcal{S}_t$, denote as $\{\boldsymbol{p}_t^1, \boldsymbol{p}_t^2\}$. The optimal pose combination follows:

$$\boldsymbol{p}_b^*, \boldsymbol{p}_t^* = \underset{\boldsymbol{p}_b \in \{\boldsymbol{p}_b^1, \boldsymbol{p}_b^2\}, \boldsymbol{p}_t \in \{\boldsymbol{p}_t^1, \boldsymbol{p}_t^2\}}{\arg\min} \Omega(\delta\boldsymbol{p}, \delta\boldsymbol{p}') \tag{5}$$

where $\delta\boldsymbol{p}$ denotes the relative pose from forward kinematics, and $\delta\boldsymbol{p}' = \boldsymbol{p}_b^T \cdot \boldsymbol{p}_t$ denotes the relative pose from prediction.

3) Key Frame Enhanced Pose Optimization: Upon obtaining key frames, the relative pose between these frames and the current frame is established with forward kinematics, defined as $\{\delta\boldsymbol{p}_i\}_{i=1}^{K_f}$, for optimizing $\boldsymbol{p}_t$. As a result, we have a set of observations, $\{\boldsymbol{O}_i\}_{i=1}^{K_f}$, where $\boldsymbol{O} = \{\boldsymbol{I}_c, \boldsymbol{I}_d, \mathcal{M}, \delta\boldsymbol{p}\}$, where $\mathcal{M} = \{\boldsymbol{u} \leftrightarrow \boldsymbol{X}\}$. For optimization, the data from all reference views in each key frame with the correct symmetry estimation are applied. The optimization objectives include two main parts. First, we aim to minimize the 2D reprojection error. The loss function is defined as:

$$\mathcal{L}_{2D} = \sum_{i=1}^{K_f} \sum_{j=1}^{K_r} \rho\left(\|\pi(\boldsymbol{p}_t \cdot \delta\boldsymbol{p}_{ij} \cdot \boldsymbol{X}_{ij}) - \boldsymbol{u}_{ij}\|_2\right) \tag{6}$$

where $\pi(\cdot)$ represents the projection function, and $\rho(\cdot)$ is a robust cost function, Caushy loss, which mitigates the influence of outliers. Second, we aim to minimize the 3D distance error, and the loss function is defined as:

$$\mathcal{L}_{3D} = \sum_{i=1}^{K_f} \sum_{j=1}^{K_r} \rho\left(\|\pi^{-1}(\boldsymbol{u}_{ij}) - \boldsymbol{p}_t \cdot \delta\boldsymbol{p}_{ij} \cdot \boldsymbol{X}_{ij}\|_2\right) \tag{7}$$

where $\pi^{-1}(\cdot)$ serves as the back-projection function that transforms 2D image coordinates $\boldsymbol{u}$ into 3D spatial coordinates $\boldsymbol{X}$ with the depth known. Combining the two items, the complete loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_{2D} + \lambda \mathcal{L}_{3D} \tag{8}$$

where λ is weighting factors, which is set to 1 in this paper.

IV. EXPERIMENTAL RESULTS

In this section, we compare our method with CAD model-based object pose estimation methods and camera-to-robot pose estimation methods. Additionally, we conducted a quantitative comparison with traditional Marker-based hand-eye calibration, showcasing the high precision of our approach.

A. Experiment Setup

Dataset. We consider two datasets: RealSense-Franka [11] and our synthetic dataset SynEEPose. RealSense-Franka comprises 4 video sequences, recording the movements of a Franka Panda. We extracted segments from each video that conformed to our settings, yielding segments of 565 frames, 500 frames, 451 frames, and 480 frames, respectively. SynEEPose is a synthetic dataset generated with Blender by ray-tracing [44]. We selected six commonly used end-effectors and three robot arms, combining them to form various robot configurations. Each configuration was rendered in 10 video segments, each containing 300 frames. In total, SynEEPose comprises 180 video segments, amounting to 54,000 frames. The selected robot arms are Franka Panda, UR10e, and UR5e, while the end-effectors include Franka Panda, Robotiq-2F85, Robotiq-2F140, Kinova-3F, Robotiq-3F, and Shadow Hand. These combinations cover a

	MegaPose[32]		MegaPose†[32]		SAM6D[29]		FoundationPose[33]		FEEPE(Ours)	
Traning-free	✗		✗		✗		✗		✓	
	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S	ADD	ADD-S
Franka Panda	59.21	68.10	65.09	70.63	78.22	80.90	81.48	83.79	**85.24**	**86.29**
Kinova-3F	20.02	26.85	46.29	50.58	80.41	81.91	**87.13**	**87.69**	85.24	86.36
Shadow Hand	2.89	9.29	45.01	46.03	71.77	74.34	59.51	65.99	**84.53**	**85.23**
Robotiq-3F	29.66	37.98	63.62	66.88	79.97	86.26	82.83	87.85	**88.73**	**91.67**
Robotiq-2F85	22.79	51.80	23.42	56.63	38.79	77.27	41.12	79.41	**84.33**	**86.11**
Robotiq-2F140	22.12	52.29	42.85	64.05	39.35	75.84	35.96	69.54	**84.87**	**86.15**
RealSense-Franka	0.19	1.03	13.88	38.11	27.48	47.68	19.94	47.17	**29.27**	**48.71**
Average	22.41	35.33	42.88	56.13	59.43	74.89	58.28	74.49	**77.46**	**81.50**

TABLE I: Model-based object pose estimation results. Values represent the AUC of ADD and ADD-S metrics with a threshold of 1 cm, where values to the left of '/' are ADD metrics and values to the right are ADD-S metrics. MegaPose corresponds to MegaPose-RGBD [32], and MegaPose† represents MegaPose-RGB [32] + multi-hypothesis + ICP. The first six rows of the test dataset represent different end-effectors from the SynEEPose dataset.

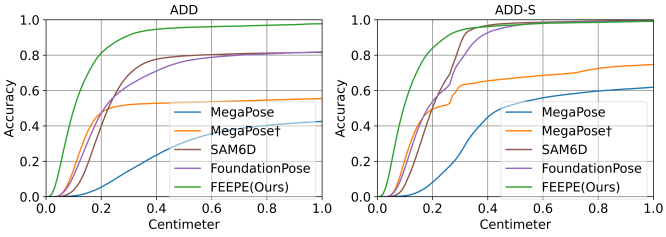

Fig. 3: Accuracy curves. MegaPose† represents MegaPose-RGB [32] + multi-hypothesis + ICP, and MegaPose corresponds to MegaPose-RGBD [32]. The results shown are statistical averages of performance across all datasets.

range of features such as weak textures, symmetric designs, two-finger and three-finger grippers, and dexterous hands. During the generation of SynEEPose, we applied domain randomization to the background, ambient lighting, camera poses, and the states of the robot arms to better evaluate the generalization capability of models. Please refer to the supplementary materials for a visualization of data samples.

Metric. We follow instance-level pose estimation, using the Average Distance (ADD) and ADD-S [45] as metrics.

B. Comparison with CAD Model-based Pose Estimation

We compare our method with FounationPose [33], MegaPose [32], and SAM-6D [29], the CAD model-based pose estimation methods. To ensure a fair comparison, we provide all methods with ground truth segmentation masks. As shown in Figure 3, the accuracy curves for ADD and ADD-S metrics demonstrate that our method consistently outperforms the others, particularly achieving a significant leading advantage in the low-distance threshold region where high accuracy is required. This superior performance is further validated by the quantitative results in Table I, which reports the AUC values for ADD and ADD-S metrics with a threshold of 1 cm. Notably, while other methods have been trained on large 6D pose estimation datasets, our method achieves superior performance without training on such datasets. The results highlight the robustness and effectiveness of our approach, particularly in scenarios involving symmetrical end-effectors. It should be clarified that the performance difference between RealSense-Franka and SynEEPose is due to RealSense-Franka being a manually annotated dataset for robot pose estimation with errors in end-effector pose annotations. For detailed visualizations, refer to the supplementary materials.

C. Ablation Study

Ablation study of critical design choices. We test four different setups of our method: **A:** Only using DINOv2 2D-3D matching and solving for the end-effector pose through PnP; **B:** Performing optimization with only one frame on top of A; **C:** Performing temporal key frames-enhanced pose optimization on top of A without symmetry discrimination; **D:** Our full method, adding symmetry discrimination on top of C. Results are shown in Table II. Notably, method B achieves comparable to SAM6D [29] and FoundationPose [33] by only using single frame observations. Results of methods C and D also show that multi-frame optimization and symmetry discrimination both boost our performance significantly, validating the effectiveness of these designs.

Method	ADD	ADD-S
A	7.50	19.49
B	54.28	70.67
C	74.40	79.04
D	77.46	81.50

TABLE II: Ablation study of critical design choices.

Effects of the Layers of DINOv2. We visualize the features extracted at different layers of DINOv2 for reference and target image pairs in Figure 4. The visualized results reveal that with increasing network depth, semantic features are enhanced at the expense of positional features [13]. This loss of positional or semantic information can adversely affect matching accuracy, especially for symmetric end-effectors. As shown in Figure 4, intermediate layers offer a effective trade-off. Table III shows results when using feature descriptors from different layers. Intuitively, the intermediate features perform best.

Layer	ADD	ADD-S
11	73.05	77.84
19	77.46	81.50
23	67.86	73.60

TABLE III: Effects of different DINOv2 layers.

Effects of Hyperparameters. We study how certain hyperparameters' choices influence our method's performance. Key hyperparameters include: the number of rendered templates, the number of reference views selected for matching, the number of key frames selected for multi-frame optimization. Quantitative results are shown in Table IV.

Hyperparameters	Templates				Reference views				Key frames			
	240	480	**960**	1920	1	3	**5**	10	2	4	**8**	16
ADD	73.08	75.06	77.46	77.66	64.21	75.40	77.46	77.98	73.75	76.26	77.46	77.72
ADD-S	77.82	79.53	81.50	81.64	69.10	79.57	81.50	81.86	78.54	80.60	81.50	81.71
Speed(ms)	63	65	67	71	55	61	67	81	54	59	67	86

TABLE IV: Effects of hyperparameters. **Bold** numbers indicate hyperparameters selected in the end.

Intuitively, more templates, more reference views and more key frames provide more information useful for end-effector pose estimation, and also result in longer inference time. Our experiments confirm this intuition, also showing that performance gradually saturates with larger hyperparameters. As a balance between performance and inference speed, we use 960 total templates, 5 reference views, and 8 keyframes for optimzation as our final method setup.

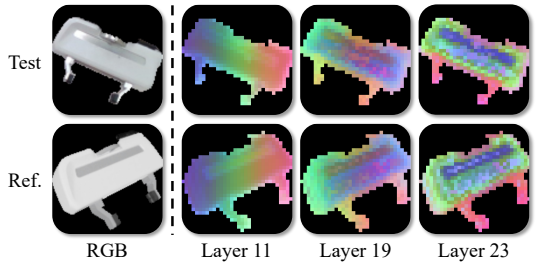

Fig. 4: Features visualization from various DINOv2 layers.

Method	Training-free	ADD@2cm	ADD-S@2cm
RoboPose[9]	✗	0.68	6.94
CTRNet[10]	✗	5.46	22.57
RoboKeyGen[11]	✗	9.35	29.99
FEEPE(Ours)	✓	**60.5**	**75.22**

TABLE V: Comparison with robot pose estimation methods on RealSense-Franka dataset.

D. Comparison with Robot Pose Estimation

Our end-effector pose estimation algorithm ultimately serves as a **robot self-calibration algorithm**. Consequently, we compare our approach with learning-based camera-to-robot pose estimation algorithms, specifically RoboPose [9], CTRNet [10], and RoboKeyGen [11]. Table V presents comparison results on the RealSense-Franka dataset. To obtain the end-effector pose from camera-to-robot pose estimation methods, we use the robot joint angles and forward kinematics to calculate the pose of the end effector in the robot space, and then transfer it to the camera space using the predicted camera-to-robot pose. The AUC threshold is raised to 2cm because all robot pose estimation methods yield near-zero results on the 1cm threshold. Even though the robot pose estimation methods have undergone robot-specific training on large amounts of data, our method outperforms all three baselines by a very large margin.

E. Real-world high-precision targeting experiment

Following [46], we conduct a high-precision targeting experiment to compare with marker-based hand-eye calibration method [47]. Both our method and the marker-based method use the images and the robot arm's position to calculate the transformation from the camera coordinate system to the robot base coordinate system. We selected 20 positions for calibration, covering as much of the workspace as possible. For each specified number of views, we sample different views from the 20 positions and experiment 3 times to reduce variance. Additionally, the marker-based method requires a marker board to be attached to the end-effector, and our method uses Track-anything [48] for segmentation with a manually specified prompt for the first frame.

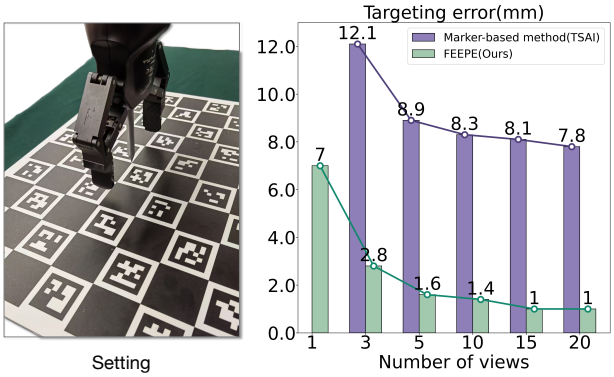

Fig. 5: Results in the real-world high-precision targeting.

As shown in Figure 5, we detect the 5 different corners of the marker board using OpenCV [49] and transform their positions to the robot base coordinate system via the calibration results from the process mentioned above. Then we use a pointer attached to the end-effector to tip these positions and manually measure the targeting error. The results in Figure 5 demonstrate that our method consistently outperforms traditional marker-based calibration across all numbers of views. Moreover, our approach achieves an accuracy within 3mm using just three frames and reaches 1mm accuracy with 15 frames. Overall, compared to conventional hand-eye calibration methods, our method is not only marker-free and online but also significantly more precise. Additionally, we also conduct real-world grasping experiments. Please refer to the supplementary materials for details.

V. CONCLUSION

In conclusion, we introduced FEEPE, a foundation feature-driven, online, training-free, and generalizable end-effector pose estimation method. Extensive experiments demonstrate that FEEPE outperforms learning-based and traditional methods, and provides superior flexibility and generalization, enabling effective online robot self-calibration.

Limitations. Although our method shows effectiveness in end-effector pose estimation, its reliance on prior knowledge of the relative end-effector's pose provided by the robot limits its applicability to general instance-level 6D object pose estimation. One of the future works could focus on adapting our approach to broader scenarios.

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] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 590–596.

[3] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 3400–3407.

[4] I. Fassi and G. Legnani, "Hand to sensor calibration: A geometrical interpretation of the matrix equation ax= xb," *Journal of Robotic Systems*, vol. 22, no. 9, pp. 497–506, 2005.

[5] G. Du and P. Zhang, "Online robot calibration based on vision measurement," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 6, pp. 484–492, 2013.

[6] G. Du, P. Zhang, and D. Li, "Online robot calibration based on hybrid sensors using kalman filters," *Robotics and Computer-Integrated Manufacturing*, vol. 31, pp. 91–100, 2015.

[7] 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.

[8] 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.

[9] 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.

[10] J. Lu, F. Richter, and M. C. Yip, "Markerless camera-to-robot pose estimation via self-supervised sim-to-real transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 296–21 306.

[11] Y. Tian, J. Zhang, G. Huang, B. Wang, P. Wang, J. Pang, and H. Dong, "Robokeygen: Robot pose and joint angles estimation via diffusion-based 3d keypoint generation," *arXiv preprint arXiv:2403.18259*, 2024.

[12] J. Zhang, C. Herrmann, J. Hur, L. Polania Cabrera, V. Jampani, D. Sun, and M.-H. Yang, "A tale of two features: Stable diffusion complements dino for zero-shot semantic correspondence," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[13] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel, "Deep vit features as dense visual descriptors," *arXiv preprint arXiv:2112.05814*, vol. 2, no. 3, p. 4, 2021.

[14] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[15] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[16] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Ep n p: An accurate o (n) solution to the p n p problem," *International journal of computer vision*, vol. 81, pp. 155–166, 2009.

[17] Y. Tian, J. Zhang, Z. Yin, and H. Dong, "Robot structure prior guided temporal attention for camera-to-robot pose estimation from image sequence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8917–8926.

[18] H. Cheng, Y. Wang, and M. Q.-H. Meng, "Real-time robot end-effector pose estimation with deep network," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 921–10 926.

[19] 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.

[20] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

[21] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7668–7677.

[22] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3343–3352.

[23] Z. Li, G. Wang, and X. Ji, "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7678–7687.

[24] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 632–11 641.

[25] Y. He, H. Huang, H. Fan, Q. Chen, and J. Sun, "Ffb6d: A full flow bidirectional fusion network for 6d pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3003–3013.

[26] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*. Springer, 2020, pp. 574–591.

[27] M. Gou, H. Pan, H.-S. Fang, Z. Liu, C. Lu, and P. Tan, "Unseen object 6d pose estimation: a benchmark and baselines," *arXiv preprint arXiv:2206.11808*, 2022.

[28] F. Hagelskjær and R. L. Haugaard, "Keymatchnet: Zero-shot pose estimation in 3d point clouds by generalized keypoint matching."

[29] J. Lin, L. Liu, D. Lu, and K. Jia, "Sam-6d: Segment anything model meets zero-shot 6d object pose estimation," *arXiv preprint arXiv:2311.15707*, 2023.

[30] J. Huang, H. Yu, K.-T. Yu, N. Navab, S. Ilic, and B. Busam, "Matchu: Matching unseen objects for 6d pose estimation from rgb-d images," *arXiv preprint arXiv:2403.01517*, 2024.

[31] H. Zhao, S. Wei, D. Shi, W. Tan, Z. Li, Y. Ren, X. Wei, Y. Yang, and S. Pu, "Learning symmetry-aware geometry correspondences for 6d object pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14 045–14 054.

[32] 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.

[33] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," *arXiv preprint arXiv:2312.08344*, 2023.

[34] S. Moon, H. Son, D. Hur, and S. Kim, "Genflow: Generalizable recurrent flow for 6d pose refinement of novel objects," *arXiv preprint arXiv:2403.11510*, 2024.

[35] I. Shugurov, F. Li, B. Busam, and S. Ilic, "Osop: A multi-stage one shot object pose estimation framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6835–6844.

[36] P. Ausserlechner, D. Haberger, S. Thalhammer, J.-B. Weibel, and M. Vincze, "Zs6d: Zero-shot 6d object pose estimation using vision transformers," *arXiv preprint arXiv:2309.11986*, 2023.

[37] V. N. Nguyen, T. Groueix, M. Salzmann, and V. Lepetit, "Gigapose: Fast and robust novel object pose estimation via one correspondence," *arXiv preprint arXiv:2311.14155*, 2023.

[38] T. Wang, G. Hu, and H. Wang, "Object pose estimation via the aggregation of diffusion features," *arXiv preprint arXiv:2403.18791*, 2024.

[39] E. P. Örnek, Y. Labbé, B. Tekin, L. Ma, C. Keskin, C. Forster, and T. Hodan, "Foundpose: Unseen object pose estimation with foundation features," *arXiv preprint arXiv:2311.18809*, 2023.

[40] V. N. Nguyen, Y. Hu, Y. Xiao, M. Salzmann, and V. Lepetit, "Templates for 3d object pose estimation revisited: Generalization to new objects and robustness to occlusions," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 6771–6780.

[41] Á. González, "Measurement of areas on a sphere using fibonacci and latitude–longitude lattices," *Mathematical Geosciences*, vol. 42, pp. 49–64, 2010.

[42] B. Wen and K. E. Bekris, "Bundletrack: 6d pose tracking for novel objects without instance or category-level 3d models," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[43] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE transactions on image processing*, vol. 6, no. 9, pp. 1305–1315, 1997.

[44] Q. Dai, J. Zhang, Q. Li, T. Wu, H. Dong, Z. Liu, P. Tan, and H. Wang, "Domain randomization-enhanced depth simulation and restoration for perceiving and grasping specular and transparent objects," in *European Conference on Computer Vision*. Springer, 2022, pp. 374–391.

[45] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *Robotics: Science and Systems XIV*, Aug 2018. [Online]. Available: http://dx.doi.org/10.15607/rss.2018.xiv.019

[46] L. Chen, Y. Qin, X. Zhou, and H. Su, "Easyhec: Accurate and automatic hand-eye calibration via differentiable rendering and space exploration," *IEEE Robotics and Automation Letters (RA-L)*, 2023.

[47] R. Tsai and R. Lenz, "A new technique for fully autonomous and efficient 3d robotics hand/eye calibration," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 345–358, 1989.

[48] J. Yang, M. Gao, Z. Li, S. Gao, F. Wang, and F. Zheng, "Track anything: Segment anything meets videos," 2023.

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