RNNPose: 6-DoF Object Pose Estimation via Recurrent Correspondence Field Estimation and Pose Optimization

Yan Xu, Kwan-Yee Lin, Guofeng Zhang, Xiaogang Wang, and Hongsheng Li

***Abstract*—6-DoF object pose estimation from a monocular image is a challenging problem, where a post-refinement procedure is generally needed for high-precision estimation. In this paper, we propose a framework, dubbed RNNPose, based on a recurrent neural network (RNN) for object pose refinement, which is robust to erroneous initial poses and occlusions. During the recurrent iterations, object pose refinement is formulated as a non-linear least squares problem based on the estimated correspondence field (between a rendered image and the observed image). The problem is then solved by a differentiable Levenberg-Marquardt (LM) algorithm enabling end-to-end training. The correspondence field estimation and pose refinement are conducted alternately in each iteration to improve the object poses. Furthermore, to improve the robustness against occlusion, we introduce a consistency-check mechanism based on the learned descriptors of the 3D model and observed 2D images, which downweights the unreliable correspondences during pose optimization. We evaluate RNNPose on several public datasets, including LINEMOD, Occlusion-LINEMOD, YCB-Video and TLESS. We demonstrate state-of-the-art performance and strong robustness against severe clutter and occlusion in the scenes. Extensive experiments validate the effectiveness of our proposed method. Besides, the extended system based on RNNPose successfully generalizes to multi-instance scenarios and achieves top-tier performance on the TLESS dataset.**

***Index Terms*—Object pose estimation, recurrent neural network, uncertainty in optimization.**

I. Introduction

6-DoF object pose estimation is of crucial importance in various applications, including augmented reality, robotic manipulation, and autonomous driving. Influenced by varying illuminations and occlusions, appearances of the differently posed objects may vary significantly from different views, which poses great challenges for 6-DOF object pose estimation from a single color image.

The earlier-phase learning-based methods usually estimate the 6-DoF pose in a one-shot manner. Yu et al. [1] proposed to directly regress the object's center and distance as well as rotation components with a convolutional neural network. More recent works [2], [3], [4], [5], [6], [7], [8] proposed to first estimate 2D-3D correspondences between the observed image and the object model, and then solve for the object pose with PnP algorithm [9], [10]. The performance of these one-shot methods is generally constrained by the network capacity and prone to be affected by adverse conditions such as varying illumination and occlusion.

The recent top-performing methods [11], [12], [13], [14], [15] additionally include a pose refinement procedure which substantially improves the performance. Some of these frameworks [11], [12] rely on depth sensors and refine the poses with the ICP algorithm [16]. To avoid the expensive depth sensor, Li et al. [13] and Manhardt et al. [14] pioneered the RGB-based pose refinement. During refinement, these methods first render a reference color image according to the coarse pose estimate. This rendered image along with the observed image is then fed to a CNN to directly predict the residual pose for refining the coarse pose [13], [14], [15]. While these methods perform well in ideal scenarios based on massive training data, the pose regression becomes less stable in practice. More recently, Iwase et al. [17] formulated the object pose refinement as an optimization problem based on feature alignment, and reported significant performance improvements. In their work, the encoded features of a 3D model by a neural network are projected to the 2D image plane according to the pose parameters. Thereafter, the pose optimization is conducted by aligning the projected features with the observed target image features. As the pose optimization depends on the gradients from the pixel-level feature differences, the feature alignment based methods are only applicable to small inter-frame pose variations [18] and are not quite robust with erroneous initial poses. Moreover, Iwase et al. [17] still have a limited design for occlusion handling, which might limit the deployment scope.

In this work, we propose a recurrent object pose refinement framework, dubbed RNNPose. Unlike RePose [17] optimizing

Manuscript received 18 March 2023; revised 9 September 2023; accepted 17 January 2024. Date of publication 30 January 2024; date of current version 5 June 2024. This work was supported in part by National Key R&D Program of China Project under Grant 2022ZD0161100, in part by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd. under the Innovation and Technology Commission (ITC)'s InnoHK, by General Research Fund of Hong Kong RGC Project under Grant 14204021. Hongsheng Li is a PI of CPII under the InnoHK. Recommended for acceptance by P. TAN. *(Corresponding author: Hongsheng Li.)*

Yan Xu, Kwan-Yee Lin, and Xiaogang Wang are with the Multimedia Laboratory, The Chinese University of Hong Kong, Hong Kong, SAR, China (e-mail: yanxu@link.cuhk.edu.hk; junyilin@cuhk.edu.hk; xgwang@ee.cuhk.edu.hk).

Guofeng Zhang is with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China (e-mail: zhangguofeng@zju.edu.cn).

Hongsheng Li is with the Multimedia Laboratory, The Chinese University of Hong Kong, Hong Kong, SAR, China, also with the Centre for Perceptual and Interactive Intelligence, Hong Kong, SAR, China, and also with the Shanghai AI Laboratory, Shanghai 200031, China (e-mail: hsli@ee.cuhk.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2024.3360181, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2024.3360181

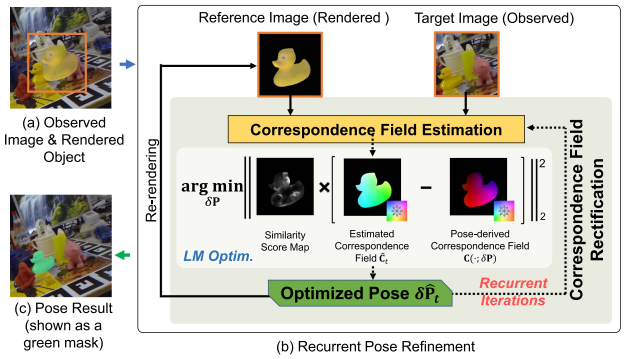


Fig. 1. Overall framework. (a) Before refinement, a reference image is rendered according to the object initial pose (shown in a fused view). (b) Our RNN-based framework recurrently refines the object pose based on the estimated correspondence field between the reference and target images. The pose is optimized to be consistent with the reliable correspondence estimations highlighted by the similarity score map (built from learned 3D-2D descriptors) via differentiable LM optimization. (c) The output refined pose.

poses based on feature alignment, we formulate the object pose optimization based on the correspondence field estimated by a carefully designed RNN. Given the extensive receptive field of RNN and the enhancements from several proposed modules, RNNPose is capable of handling significant initial pose errors and occlusions. The overall pipeline is illustrated in Fig. 1. Before refinement, a reference image of the object is rendered according to the initial pose estimation. Our refinement module refines the initial pose based on this rendered image and the observed image. To increase the tolerance to erroneous initial poses, our refinement is conducted within a recurrent framework, where the pose optimization is formulated as a non-linear least squares problem based on estimated correspondence fields. In each recurrent iteration, the dense correspondences between the rendered image and observed image are estimated, and the object pose is then optimized to be consistent with the correspondence field estimation. The architecture of our correspondence estimation is inspired by the recent optical flow estimation techniques [19], [20], which is integrated with our pose optimization recurrently. To suit our task where unpatterned objects and illumination variations are ubiquitous, we further include a correspondence field rectification step in each recurrent iteration based on the currently optimized pose. The inconsistent correspondences are rectified by enforcing rigid-transformation constraints. The rectified correspondence field is also used to initialize the next recurrent iteration to improve the robustness further.

For occlusion handling, we introduce a 3D-2D hybrid network trained with a contrastive loss, which generates distinctive point-wise descriptors for the 3D object model and observed 2D images. A similarity score is constructed for each estimated correspondence pair based on the learned descriptors, with which to downweight the unreliable correspondences during pose optimization. The pose optimization is conducted by a differentiable Levenberg-Marquardt (LM) algorithm (sharing the ideas of [21], [22]) for end-to-end training.

Our contributions are summarized as follows:

1) We propose an RNN-based 6-DoF pose refinement framework that is robust to large initial pose errors and occlusions. During recurrent iterations, the pose optimization is formulated as a non-linear least squares problem based on the estimated correspondence field. Meanwhile, the correspondence field is also being rectified and improved by the optimized pose for robustness.
2) To handle the occlusions, a 3D-2D hybrid network is introduced to learn point-wise descriptors which are used to downweight unreliable correspondence estimations during pose optimization.
3) We build a stand-alone system based on this pose refinement framework, which is able to handle multiple instances of the same class in a scene efficiently.
4) We achieve new state-of-the-art performances on LINEMOD, Occlusion LINEMOD, and YCB-Video datasets. Our code is public at https://github.com/DecaYale/RNNPose.

II. Related Work

One-Shot 6-DoF Object Pose Estimation: 6-DoF object pose estimation systems aim to estimate the 3-DoF orientations and 3-DoF locations of rigid objects.

Classic methods are mainly based on template matching techniques [23], [24], [25], which are prone to errors and cannot generalize well to different environments. The boom of deep learning has significantly improved object pose estimation. A series of methods have been proposed to holistically estimate object poses from monocular color images [1], [26], [27], [28] or with the aid from depth sensors [29], [30], [31], [32], [33], [34]. These methods took advantage of the CNNs' regression ability to learn mapping functions from the observed images to object poses. More recently, correspondence-based methods [2], [3], [4], [5], [6], [7], [8], [35], [36], [37], [38], [39], [40] become more popular. They employed CNNs to estimate the 3D correspondences of a set of observed 2D keypoints of the object in images. and then solve for poses with PnP [9], [10]. These methods may estimate the object's bounding box corners [35], [36], predict dense 2D-3D correspondence maps [3], or vote the predefined keypoint locations by all object pixels jointly [4]. Hodan et al. [5] proposed to handle symmetric objects by segmenting 3D models into patches and estimating the patch centers. ZebraPose [8] proposed a coarse-to-fine surface encoding scheme to improve the robustness of correspondence association. Some recent works [6], [7], [28] developed networks to solve the PnP problem in a differentiable manner. EPro-PnP [7] created a probabilistic Perspective-n-Points layer and model the loss function with KL divergence for end-to-end training to improve the estimation robustness. There is also a series of works focusing on model-free [41], [42], [43], [44] or self-supervised [45], [46], [47] 6-DoF object pose estimation. However, these methods are still in the exploration stage and their estimation accuracy still cannot satisfy the scenarios where high-precision estimation is demanded.

6-DoF Object Pose Refinement: The above direct object pose estimation methods usually become less stable when varying

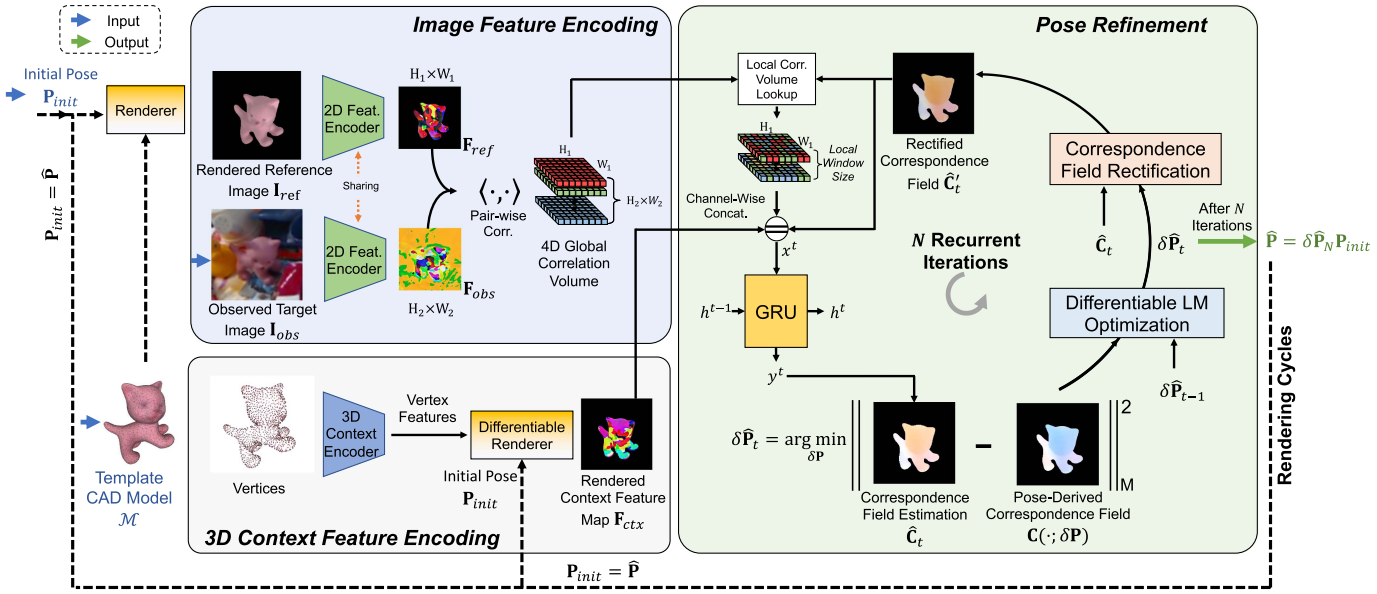


Fig. 2. Overview of the proposed method. For pose refinement, a reference image $\mathbf{I}_{ref}$ is rendered with the object CAD model and its initial pose $\mathbf{P}_{init}$. The *image feature encoding* module encodes the rendered image $\mathbf{I}_{ref}$ and the observed image $\mathbf{I}_{obs}$ to feature maps and build a 4D global correlation volume. In parallel, the *3D context feature encoding* module encodes the 3D model geometry and render the 4D features to a 2D context feature map $\mathbf{F}_{ctx}$ according to the initial pose estimation. During *pose refinement*, the correspondence field $\hat{\mathbf{C}}_t$ and the residual pose $\delta\hat{\mathbf{P}}_t$ are alternately estimated in a recurrent framework. After the LM pose optimization, correspondence field estimation $\hat{\mathbf{C}}_t$ is rectified as $\hat{\mathbf{C}}'_t$ by enforcing rigid-transformation constraints with the currently optimized pose $\delta\hat{\mathbf{P}}_t$ to further improve next-iteration estimations. After N recurrent iterations, the reference image $\mathbf{I}_{ref}$ is re-rendered with the current pose estimation.

illuminations and occlusions exist. Many methods [11], [12], [13], [14], [15], [17] hence conducted pose refinement based on the estimated coarse initial pose above, which achieved significant performance gains. Some of these methods [11], [12] relied on depth data from costly sensors and utilized ICP to align the known object model to the observed depth image. While [13], [14], [15] first rendered a 2D object image according to the initial pose and then compared the rendered image with the observed image via a CNN to estimate the residual pose. These RGB-based methods are especially attractive due to their economical nature. However, most of these methods need massive training data and are not quite robust in practical scenarios. Moreover, they need a cumbersome CNN for pose regression, which sacrifices efficiency. Iwase et al. [17] proposed to alleviate such dilemmas by reusing the image features extracted by CNN and attained real-time processing. Concretely, they employed the CNN as an image feature encoder, based on which to formulate a non-linear optimization problem to align the features from the inference and target images for pose refinement inspired by BA-Net [21]. Though efficient, their formulation is built upon overlapped object regions across the reference image and target image, which is thus less robust against erroneous pose initialization. Grabner et al. [48] proposed to refine the pose based on the correspondences, but their method is still limited to ideal scenarios.

Non-Linear Least Squares Optimization With Deep Learning: Non-linear least squares optimization algorithms, such as Gauss-Newton [49] and Levenberg-Marquardt [50], are widely used in computer vision [51], [52], [53], [54], given their efficient and effective nature. Recently, the differentiability of the optimization algorithm itself has been widely studied and several works [21], [22], [55], [56] have included the differentiable optimization algorithm during the network training for localization systems and visual SLAMs. These inspire our formulation for object pose refinement.

III. Method

Given an observed object image $\mathbf{I}_{obs}$, an initial object pose estimate $\mathbf{P}_{init}$ and the object's CAD model $\mathcal{M}$ as inputs, a 6-DoF pose refinement system aims to further improve the object pose estimation. In this paper, we propose a recurrent pose refinement method, dubbed RNNPose, which is robust to erroneous initial poses and occlusions. Our method is based on a rendering pipeline and may have several rendering cycles as illustrated by Fig. 2. At the beginning of the first rendering cycle, a reference image $\mathbf{I}_{ref}$ is rendered with the object's CAD model according to its initial pose $\mathbf{P}_{init}$ (estimated by any direct methods [4], [11]). Then, the rendered reference image, the observed target image, and the vertices of the CAD model are encoded as high-dimensional features which will be used to estimate the correspondences (between the rendered image and observed image) in the follow-up pose refinement module. The pose refinement module constitutes our major contribution, where we formulate an optimization problem based on the correspondence estimations. We integrate correspondence field estimation and pose refinement into a recurrent framework for robustness and efficiency. To handle occlusions, we generate point-wise distinctive descriptors for the 3D object model and observed images with a 3D-2D hybrid network, with which to downweight the unreliable correspondences during pose optimization. After every several recurrent iterations, the reference

image $\mathbf{I}_{ref}$ is re-rendered with the currently optimized pose to decrease the pose gap to the target for the next cycle.

In the ensuing subsections, we will detail the feature extraction (Section III-A), recurrent pose refinement (Section III-B), and the loss functions for training (Section III-C).

A. 2D-3D Feature Encoding and Rendering

The rendered reference image $\mathbf{I}_{ref}$ and observed target image $\mathbf{I}_{obs}$ first need to be encoded into high-dimensional feature maps $\mathbf{F}_{ref}$ and $\mathbf{F}_{obs}$ for the follow-up feature correlation volume construction [19], [20], [57], [58], [59]. The correlation volume encodes the appearance similarities between image pixels, which is essential for correspondence reasoning. In our work, we adopt several residual blocks [60] for image feature encoding, and the pair-wise correlations of the encoded features are calculated to create a global correlation volume. The global correlation volume will be frequently queried for correspondence field estimation in the follow-up pose refinement module.

Besides the pair-wise correlation volume, popular dense correspondence estimation methods also incorporate context features of the reference image for guidance. As shown in Fig. 2, to better encode the geometric contexts, unlike previous methods encoding the context features from 2D images, we directly encode the features from 3D object point clouds with a *3D context feature encoder* based on KPConv [61]. The point-wise geometric features are then rendered as a 2D context feature map $\mathbf{F}_{ctx}$ according to the initial object pose estimation. Here, we adopt a differentiable renderer [62] for feature rendering to enable geometric feature learning. We empirically found that encoding the context features from point clouds brings more robustness. Besides, the vertex features only need to be extracted once per object model and archived for inference after training, which is quite efficient.

B. Recurrent Correspondence Field Estimation and 6-DoF Pose Refinement

Based on the constructed correlation volume and encoded context features, we propose a 6-DoF object pose refinement system by integrating correspondence estimation and pose optimization as a recurrent framework. The correspondence field estimation and pose optimization rely on each other and improve recurrently for robust pose refinement. The basic pipeline is illustrated in the pose refinement module in Fig. 2.

1) Correspondence Field Estimation: For correspondence field estimation, we adopt a network architecture similar to RAFT [20] but make major modifications to suit our task, i.e., including the 3D context feature encoding (Section III-A) and correspondence rectification (Section III-B2). At the beginning of each recurrent iteration, for each spatial location of the reference image's feature map, we first look up and collect (from the global correlation volume) its correlation values with the candidate features of the target image. The candidate locations are within a square local window centered at the estimated correspondences from the previous iteration. The collected correlations are then reshaped as a local correlation volume (a 2D map) spatially aligned with the reference image. In the first iteration, we use an all-zeros correspondence field to bootstrap correlation candidate identification, while in the later iterations, the rectified correspondence field (to be elaborated in Section III-B2) is used.

After the correlation lookup, the collected local correlation volume, the rectified correspondence field, and the previously encoded context feature map $\mathbf{F}_{ctx}$ are concatenated as inputs to a GRU network to estimate the correspondence field $\hat{\mathbf{C}}_t$ for the current (t-th) recurrent iteration.

2) 6-DoF Pose Refinement. Basic Formulation: Given a reference image (with depth map) and a target image, the ground-truth correspondence field of the reference image can be derived based on the ground-truth residual pose $\delta\mathbf{P}^{gt}$ point-wisely:

$$\mathbf{C}(\mathbf{x}^i;\delta\mathbf{P}^{gt}) = \pi\left(\delta\mathbf{P}^{gt}\pi^{-1}\left(\mathbf{x}^i, z^i\right)\right), \tag{1}$$

where $\mathbf{C}(\mathbf{x}^i;\delta\mathbf{P}^{gt}) \in \mathbb{R}^2$ denotes the ground-truth correspondence field value of point $\mathbf{x}^i$, and z^i denotes the associated rendered depth value. Here, $\pi(\cdot)$ and $\pi^{-1}(\cdot;z^i)$ are the projection (3D-to-2D) and inverse projection (2D-to-3D) functions of a pinhole camera model.

To estimate the residual pose, we take the correspondence field $\hat{\mathbf{C}}_t$ estimated by the GRU as an approximation of its ground-truth, i.e., $\hat{\mathbf{C}}_t(\mathbf{x}^i) \approx \mathbf{C}(\mathbf{x}^i;\delta\mathbf{P}^{gt})$, and push the correspondence field derived by the pose argument $\delta\mathbf{P}$, i.e., $\mathbf{C}(\mathbf{x}^i;\delta\mathbf{P})$, close to the GRU's estimation by optimizing $\delta\mathbf{P}$. In this way, the residual pose parameter $\delta\mathbf{P}$ will approximate the ground-truth $\delta\mathbf{P}^{gt}$ after the optimization. The specific formulation is a non-linear least squares problem and the objective function is expressed as

$$E(\boldsymbol{\xi}) = \sum_{i=1}^{M}\left(\hat{\mathbf{C}}_t\left(\mathbf{x}^i\right) - \mathbf{C}\left(\mathbf{x}^i;\boldsymbol{\xi}\right)\right)^T\left(\hat{\mathbf{C}}_t\left(\mathbf{x}^i\right) - \mathbf{C}\left(\mathbf{x}^i;\boldsymbol{\xi}\right)\right), \tag{2}$$

where the residual pose argument $\delta\mathbf{P}$ is parameterized as its minimal representation $\boldsymbol{\xi} \in \mathfrak{se}(3)$ (of the associated Lie-algebra) during optimization. $\hat{\mathbf{C}}_t$ is the GRU-estimated correspondence field at the t-th recurrent iteration, and $\mathbf{C}(\mathbf{x}^i;\boldsymbol{\xi})$ denotes the correspondence of point $\mathbf{x}^i$ derived with the pose parameter argument $\boldsymbol{\xi}$, and M is the total number of object points in the rendered reference image.

Handling Unreliable Correspondences with Similarity Scores: The formulation of (2) is based on an impractical assumption that the correspondence field $\hat{\mathbf{C}}_t$ can be reliably estimated for all foreground regions, which is extremely difficult considering ubiquitous occlusions. We further propose to incorporate a consistency-check mechanism to downweight the unreliable values in $\hat{\mathbf{C}}_t$ during pose optimization. To model the reliability of estimated correspondence, one option is to adopt a forward-and-backward consistency check [63], [64]. However, the bidirectional consistency check doubles the computational cost, and the domain gap between the rendered images and the real images increases the learning difficulty.

We therefore propose a descriptor-based consistency check to alleviate the dilemma. The basic idea is to represent the 3D object model $\mathcal{M}$ and the observed 2D target image $\mathbf{I}_{obs}$ as two sets of distinctive descriptors point-wisely via a 3D-2D hybrid network (with KPConvs [61] and a keypoint description net [65] as backbones). The corresponding descriptors of the object model and object images are enforced to be similar, while

the non-corresponding descriptors are enforced to be dissimilar (by training with a contrastive descriptor loss function being described in Section III-C). The learned 3D model descriptors are rendered as 2D feature maps, denoted as $\mathbf{D}_{\mathcal{M}}$, according to the object pose of the reference image for fast indexing. The encoded target image descriptor map is denoted as $\mathbf{D}_{\mathcal{I}}$.

With these high-dimensional distinctive descriptors, for each estimated correspondence pair $(\mathbf{x}^i, \hat{\mathbf{C}}_t(\mathbf{x}^i))$, we measure its reliability according to the similarity between their associated 3D and 2D descriptors $(\mathbf{d}^i_{\mathcal{M}}, \mathbf{d}^i_{\mathcal{I}})$. $\mathbf{d}^i_{\mathcal{M}}$ and $\mathbf{d}^i_{\mathcal{I}}$ here are collected from the above descriptor maps: $\mathbf{d}^i_{\mathcal{M}} = \mathbf{D}_{\mathcal{M}}(\mathbf{x}^i)$ and $\mathbf{d}^i_{\mathcal{I}} = \mathbf{D}_{\mathcal{I}}(\hat{\mathbf{C}}_t(\mathbf{x}^i))$, where bilinear interpolation may be applied for non-integer correspondence coordinates.

The reliability of this correspondence pair is modeled with a similarity score:

$$w^i = \exp\left(-\frac{\left|1 - \mathbf{d}^{i^T}_{\mathcal{M}}\mathbf{d}^i_{\mathcal{I}}\right|}{\sigma}\right), \tag{3}$$

where σ is a learnable parameter (initialized with 1) adjusting the sharpness. The similarity scores are used as the weights of the Mahalanobis distance measurements in (2), which effectively downweight unreliable correspondences during optimization. By introducing a diagonal weighting matrix $\mathbf{w}^i = \begin{pmatrix} w^i & 0 \\ 0 & w^i \end{pmatrix}$, the weighted version of (2) is written as

$$E(\boldsymbol{\xi}) = \sum_{i=1}^{M} \left(\hat{\mathbf{C}}_t\left(\mathbf{x}^i\right) - \mathbf{C}\left(\mathbf{x}^i; \boldsymbol{\xi}\right)\right)^T \mathbf{w}^i \left(\hat{\mathbf{C}}_t(\mathbf{x}^i) - \mathbf{C}\left(\mathbf{x}^i; \boldsymbol{\xi}\right)\right). \tag{4}$$

The pose optimization is thus formulated as

$$\hat{\boldsymbol{\xi}} = \arg\min_{\boldsymbol{\xi}} E(\boldsymbol{\xi}), \tag{5}$$

where the pose parameter $\boldsymbol{\xi} \in \mathfrak{se}(3)$ is optimized by minimizing the objective function defined by (4).

Differentiable Residual Pose Optimization: We solve the non-linear least squares problem (5) with Levenberg-Marquardt (LM) algorithm. For the optimization in the t-th recurrent iteration, the pose parameter is initialized with the estimated pose from the previous iteration i.e., $\boldsymbol{\xi}_0 = \log(\delta\hat{\mathbf{P}}_{t-1})$. Continuing from the parameter $\boldsymbol{\xi}_{p-1}$ of the previous LM iteration, the left-multiplied increment $\triangle\boldsymbol{\xi}_p$ is computed by

$$\triangle\boldsymbol{\xi}_p = (\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}(\boldsymbol{\xi}_{p-1}), \tag{6}$$

with which we update the parameter as $\boldsymbol{\xi}_p \leftarrow \triangle\boldsymbol{\xi}_p \circ \boldsymbol{\xi}_{p-1}$, to approach the optimal solution. Here, $\mathbf{J} = -\frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}}$ is the Jacobian matrix containing the derivative of the stacked residual vector $\mathbf{r} = (r_1, r_2, \ldots, r_{2\tilde{}M})^T$ (established from (4)) with regard to a left-multiplied increment. We unroll the parameter update procedure and make the LM optimization layer differentiable to enable end-to-end network training. The differentiable optimization procedure enhances the feature learning for correspondence field estimation, which is essential to high performance. After LM optimization, the residual pose of the t-th recurrent iteration is estimated as $\delta\hat{\mathbf{P}}_t = \exp(\hat{\boldsymbol{\xi}})$, where $\hat{\boldsymbol{\xi}}$ denotes the optimized parameter after several updates with (6).

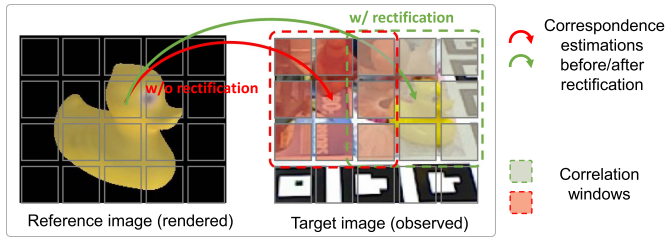


Fig. 3. With the rectified correspondences, the related local correlation windows are accordingly shifted to better locations, which improves the estimation in the next recurrent iteration.

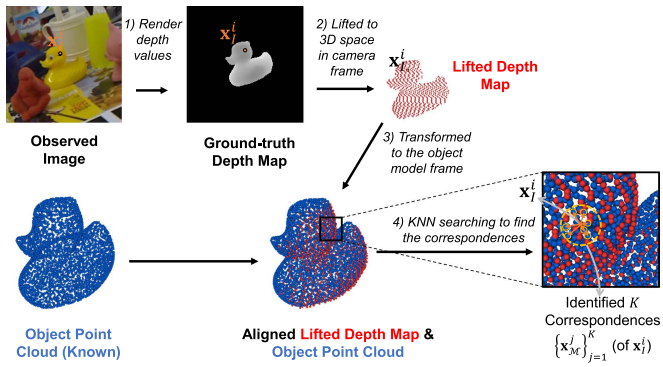


Fig. 4. Correspondence establishment between 2D image pixels and 3D model points for 3D-2D descriptor learning. To associate the correspondences from the observed 2D image to the 3D model for training, we first render a depth map and then lift the 2D pixels to 3D space. We then associate them with the object point cloud via k-NN search. The identified correspondences constitute the positive set $\{\tilde{}\}_+$ for contrastive learning in the descriptor loss L_d (8).

Correspondence Field Rectification: The erroneous initial poses usually produce large offsets between the rendered reference object and the observed object, which poses challenges for correspondence estimation. Moreover, unlike the standard scenarios of optical flow estimation [19], [20], [58], [59], unpatterned objects and varying illuminations are ubiquitous in our task, which further increases the difficulty. Considering the optimized pose by (5) is mainly supported by the reliable correspondence estimations with our weighting mechanism (3), we rectify the correspondence field as $\hat{\mathbf{C}}'_t(\mathbf{x}) = \pi\left(\delta\hat{\mathbf{P}}_t\pi^{-1}(\mathbf{x}; z)\right)$ based on the currently optimized pose $\delta\hat{\mathbf{P}}_t$. The rectification enforces the rigid-transformation constraints among the correspondence field, which improves the overall correspondence quality for the correlation volume lookup in the following recurrent iteration. A toy example is shown in Fig. 3 for better understanding.

Object Pose Estimation Update: After every N recurrent iterations, the residual pose is estimated as $\delta\hat{\mathbf{P}}_N$ by the RNN. We update the object pose estimation with the estimated residual pose $\delta\hat{\mathbf{P}}_N$ as $\hat{\mathbf{P}} \leftarrow \delta\hat{\mathbf{P}}_N\mathbf{P}_{init}$, and we re-render the reference image $\mathbf{I}_{ref}$ based on this updated pose to start the next N-recurrent-iteration refinement, as illustrated in Fig. 2. We refer to the N-recurrent-iteration refinement as a *rendering cycle*, and the initial pose $\mathbf{P}_{init}$ for the next cycle is set to $\hat{\mathbf{P}}$ accordingly. The performance and efficiency with different rendering cycles and recurrent iterations will be discussed in Section IV-B.

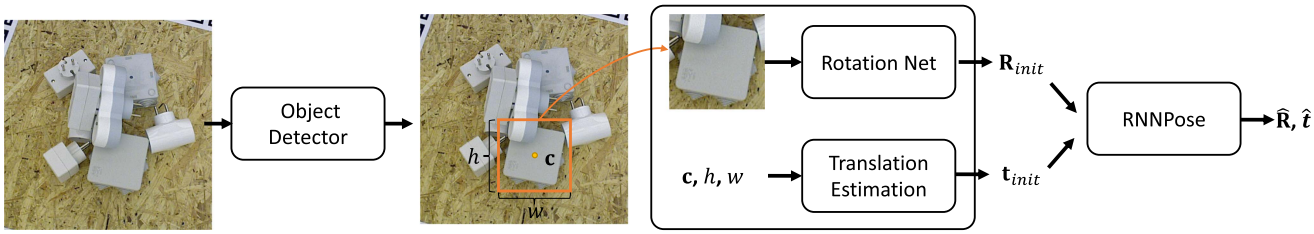


Fig. 5. Extending RNNPose to a 6-DoF pose estimation system. Given an input image, the object detector predicts the bounding boxes of all the object instances of interest. Then the regions covered by the bounding boxes are cropped out and sent to the rotation net for rotation estimation. For the translation part, we directly approximate it with (10) based on the bounding box's center and size with minimal computational overhead. The coarse estimates of rotation $\mathbf{R}_{init}$ and translation $\mathbf{t}_{init}$ are used to initialize RNNPose to obtain the refined object pose $\hat{\mathbf{R}}$, $\hat{\mathbf{t}}$.

TABLE I
(A) ABLATION STUDY ON LINEMOD DATASET

	w/o correspondence loss			w/o L_{ma}			w/o $\hat{\mathbf{C}}'_t$ rect.			w/o 3D context $\mathbf{F}_{ctx}$			w/ 2D context			Full(Ours)		
Object	0.02d	0.05d	0.1d	0.02d	0.05d	0.1d	0.02d	0.05d	0.1d	0.02d	0.05d	0.1d	0.02d	0.05d	0.1d	0.02d	0.05d	0.1d
Ape	1.29	17.60	61.23	8.65	36.03	70.35	4.40	35.76	74.51	14.86	50.48	80.10	12.19	52.22	82.33	**18.76**	**57.14**	**88.19**
Benchvise	31.60	87.08	99.32	58.14	94.30	99.71	**79.56**	98.72	**100.0**	72.26	**99.13**	**100.0**	75.26	98.37	99.81	75.17	98.25	**100.0**
Camera	19.37	70.89	94.90	45.13	82.31	95.95	**56.72**	90.09	97.91	53.63	90.69	**98.73**	56.90	**91.68**	97.78	55.39	89.12	98.04
Can	8.95	77.88	96.83	32.65	86.71	98.76	47.13	94.37	99.31	53.25	95.28	**99.80**	**53.21**	**95.62**	99.72	**54.53**	94.69	99.31
Cat	4.59	28.39	71.64	25.24	62.60	92.81	31.74	75.76	97.98	32.34	74.55	96.71	**36.81**	**79.15**	**98.55**	36.43	74.85	96.41
Driller	40.25	84.04	92.57	49.88	88.50	98.22	59.81	**96.43**	**99.70**	60.46	95.34	**99.70**	60.69	95.54	99.41	**62.44**	95.44	**99.70**
Duck	5.62	22.44	69.08	16.66	47.46	79.69	19.18	55.68	87.01	16.71	57.37	85.92	25.19	**63.62**	88.01	**25.82**	61.13	**89.30**
Eggbox	43.45	89.81	**99.65**	46.40	87.12	98.12	52.64	83.45	97.65	50.05	81.03	95.59	54.51	86.38	96.36	**59.06**	**93.80**	99.53
Glue	44.08	93.57	69.83	10.67	52.84	92.29	51.83	93.95	**99.87**	55.12	94.40	99.52	54.14	**95.71**	**99.87**	**60.14**	95.56	99.71
Holep.	6.26	15.89	51.95	31.55	65.55	95.04	32.81	70.22	96.53	24.26	66.51	93.91	20.61	56.03	91.04	**35.68**	**75.26**	**97.43**
Iron	42.33	96.09	99.08	52.14	95.48	99.69	62.46	97.32	99.59	63.74	97.45	**100.00**	63.07	**98.24**	**100.0**	**68.03**	98.16	**100.0**
Lamp	33.87	88.57	98.98	30.18	81.90	99.17	45.95	94.75	99.81	46.35	93.76	**99.81**	60.71	94.43	99.00	**61.32**	**94.91**	**99.81**
Phone	2.33	18.53	55.59	31.08	72.81	95.36	36.55	82.74	98.13	39.66	82.06	97.26	**42.68**	83.85	**98.39**	42.30	**83.95**	**98.39**
Average	21.85	60.83	85.27	33.72	73.35	93.47	44.68	82.25	96.00	44.82	82.93	95.93	47.44	83.91	96.17	**50.39**	**85.56**	**97.37**

For more detailed comparison, the evaluations with different thresholds of add(-s) metric are conducted. The thresholds are set to 2%, 5% and 10% of the model diameter, denoted as 0.02 d, 0.05 d, 0.1 d respectively.

C. Loss Functions of RNNPose

Model Alignment Loss: To supervise the residual pose estimations $\{\delta\hat{\mathbf{P}}_t|t=1\dots N\}$ generated in each rendering cycle (including N recurrent iterations), we apply these residual poses as the left-multiplied increments to the initial pose $\mathbf{P}_{init}$, having the corresponding object pose estimations $\{\hat{\mathbf{P}}_t|t=1\dots N\}$, where $\hat{\mathbf{P}}_t=\delta\hat{\mathbf{P}}_t\mathbf{P}_{init}$. Thereafter, we adopt a 3D model alignment loss to supervise these pose estimations for each rendering cycle:

$$L_{ma}=\sum_{t=1}^{N}||\hat{\mathbf{P}}_t\mathbf{X}_{\text{model}}-\mathbf{P}^{gt}\mathbf{X}_{\text{model}}||_1, \tag{7}$$

where $\hat{\mathbf{P}}_t$ is the object pose estimation mentioned above and $\mathbf{P}^{gt}$ denotes the ground-truth pose. Here, $\mathbf{X}_{\text{model}}\in\mathbb{R}^{4\times M}$ contains homogeneous coordinates of the M model points. This loss function encourages the pose estimation to be close to the ground-truth so that the transformed model points can be well aligned.

Correspondence Loss: We adopt $L1$ loss [20] for correspondence field supervision, where the ground-truth correspondence fields are derived with (1) based on ground-truth poses.

Descriptor Loss: We use circle loss $\mathcal{L}_{cir}$ [66] as the contrastive loss to supervise the point-wise descriptor learning of the 3D object model and the target images for similarity score calculation (3). Concretely, we view the target image $\mathbf{I}_{obs}$ as two parts, i.e., the foreground region (object region) denoted as $fg(\mathbf{I}_{obs})$ and the background region denoted as $bg(\mathbf{I}_{obs})$. For each foreground descriptor $\boldsymbol{d}^i_{\mathcal{I}}\in fg(\mathbf{I}_{obs})$, we first find a set of its corresponding 3D descriptors $\{\boldsymbol{d}^j_{\mathcal{M}}\}_+$ of object model via KNN searching see Fig. 4 for details. Then, $\boldsymbol{d}^i_{\mathcal{I}}\in fg(\mathbf{I}_{obs})$ is enforced to be similar to $\{\boldsymbol{d}^j_{\mathcal{M}}\}_+$ and dissimilar to the remaining non-corresponding descriptors $\{\boldsymbol{d}^k_{\mathcal{M}}\}_-$ with circle loss $\mathcal{L}_{cir}$ [66], which is expressed as $\mathcal{L}_{cir}(\boldsymbol{d}^i_{\mathcal{I}},\{\boldsymbol{d}^j_{\mathcal{M}}\}_+,\{\boldsymbol{d}^k_{\mathcal{M}}\}_-)$. Moreover, for background descriptors $\boldsymbol{d}^i_{\mathcal{I}}\in bg(\mathbf{I}_{obs})$, we constrain them to be similar to each other in the background, while to be dissimilar to the foreground

TABLE II
EFFICACY VERIFICATION OF THE PROPOSED SIMILARITY SCORES WITH THE OCCLUSION-LINEMOD DATASET

	w/o similarity score			w/ similarity score (Ours)		
Object	0.02d	0.05d	0.1d	0.02d	0.05d	0.1d
Ape	**0.17**	8.97	**38.63**	0.09	**9.74**	37.18
Can	7.29	53.69	85.50	**7.79**	**56.01**	**88.07**
Cat	**1.60**	**11.71**	27.97	**1.60**	11.63	**29.15**
Driller	13.76	52.47	78.42	**14.58**	**59.80**	**88.14**
Duck	0.18	**11.31**	47.77	**0.26**	11.13	**49.17**
Eggbox	2.98	25.96	61.28	**4.94**	**38.47**	**66.98**
Glue	6.98	35.22	**65.01**	**10.52**	40.97	63.79
Holep.	0.08	18.33	59.83	**0.42**	21.42	**62.76**
Average	4.13	27.21	58.05	**5.02**	**31.15**	**60.65**

Different thresholds of add(-s) metric are used for evaluation, i.e., 2%, 5% and 10% of the model diameter, denoted as 0.02 d, 0.05 d, 0.1 d respectively.

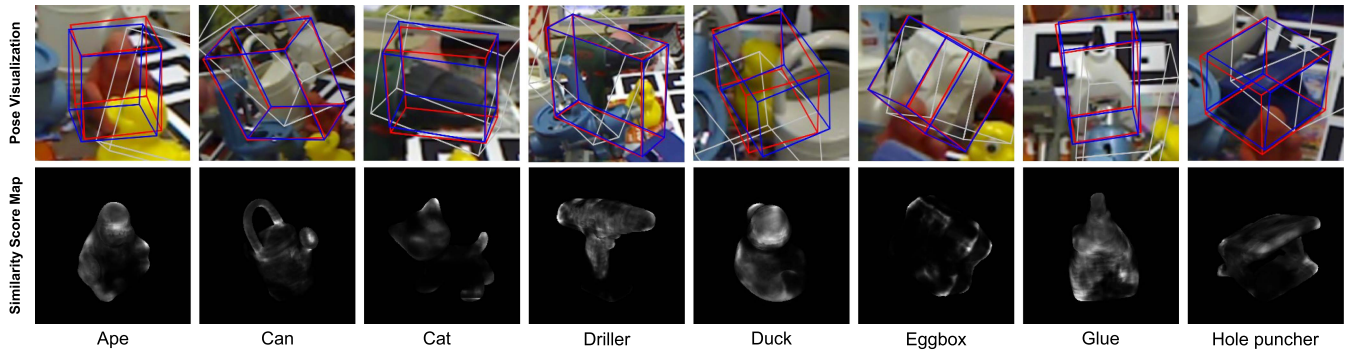


Fig. 6. Visualization of our pose estimations (first row) on Occlusion LINEMOD dataset and the similarity score maps (second row) for downweighting unreliable correspondences during pose optimization. For pose visualization, the white boxes represent the erroneous initial poses, the red boxes are estimated by our algorithm and the ground-truth boxes are in blue. Here, the initial poses for pose refinement are originally from PVNet [4] but added with significant disturbances for robustness testing.

descriptor set $fg(\mathbf{I}_{obs})$ with loss $\mathcal{L}_{cir}(\boldsymbol{d}_{\mathcal{I}}^{i}, bg(\mathbf{I}_{obs}), fg(\mathbf{I}_{obs}))$. Traversing all target image descriptors $\mathbf{d}_{\mathcal{I}}^{i}$, the descriptor loss is calculated as

$$L_d = \sum_{\boldsymbol{d}_{\mathcal{I}}^{i} \in fg(\mathbf{I}_{obs})} \mathcal{L}_{cir}(\boldsymbol{d}_{\mathcal{I}}^{i}, \{\boldsymbol{d}_{\mathcal{M}}^{j}\}_{+}, \{\boldsymbol{d}_{\mathcal{M}}^{k}\}_{-}) + \sum_{\boldsymbol{d}_{\mathcal{I}}^{i} \in bg(\boldsymbol{I}_{obs})} \mathcal{L}_{cir}(\boldsymbol{d}_{\mathcal{I}}^{i}, bg(\mathbf{I}_{obs}), fg(\mathbf{I}_{obs})) \quad (8)$$

to supervise descriptor learning. With contrastive learning, the corresponding 2D-3D descriptors would be similar while the noncorresponding ones would be dissimilar, which provides the foundation for unreliable correspondence handling with similarity scores (3).

D. 6-DoF Object Pose Estimation System Based on RNNPose

RNNPose relies on the direct estimation methods [1], [4] for pose initialization. These conventional methods usually adopt specially-designed networks for accurate pose estimation, which are usually cumbersome and computationally intensive. Moreover, most of these direct methods cannot handle multiple instances of the same class in a scene.

To extend RNNPose into a more efficient object pose estimation system, we 1) incorporate an object detection module to address the scene containing multiple same-category objects and 2) tailor a more lightweight initial pose estimation module to initialize RNNPose.

Fig. 5 illustrates our overall system. We design our object detector with MaskRCNN [67] to detect the instances of the target object. Each detected instance is cropped out and sent to the rotation estimation network to estimate object rotation $\mathbf{R}_{init}$ in the quaternion form. To handle symmetric objects, we follow [34] to define a set of rotations S that lead to the same appearance for each symmetric object, and define a symmetric-aware loss function to supervise the rotation learning:

$$L_{init} = \min_{\mathbf{R}_{sym} \in S} ||\mathbf{R}_{init}\mathbf{X}_{\text{model}} - \mathbf{R}_{gt}\mathbf{R}_{sym}\mathbf{X}_{\text{model}}||_1, \quad (9)$$

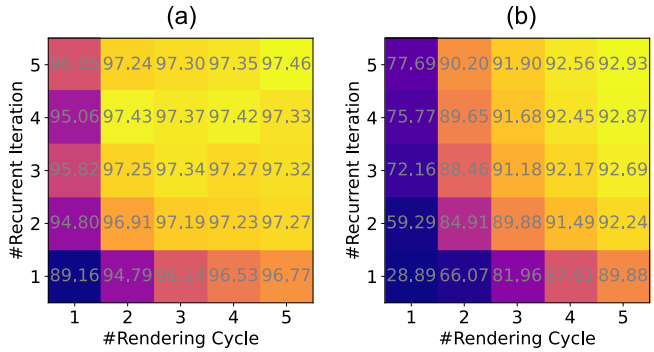


Fig. 7. ADD(-S) accuracies on LINEMOD when different recurrent iterations and rendering cycles are conducted. (a) Results are based on the initial poses from PoseCNN [11]. (b) Initial poses are the disturbed PoseCNN poses (with Gaussian noise $\sigma_t = 15$ cm, $\sigma_r = 10°$).

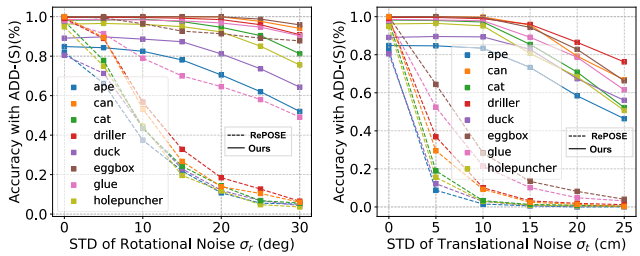


Fig. 8. Robustness comparison with RePOSE by degrading the initial poses (from PVNet [4]) with Gaussian noise on LINEMOD dataset.

TABLE III
RUNTIME ANALYSIS OF INDIVIDUAL MODULES

Module	Runtime (ms)
Ref. Image Rendering	8.88
3D Context Encoding (run once per sequence)	35.20
3D Feat. Rendering (context&descriptor)	5.39
Image Feat. Encoding	6.39
2D-3D Hybrid Net (2D part)	2.99
CF Estimation	6.21
Pose Optim.	6.23
CF Rectification	1.48

TABLE IV
COMPARISON OF ESTIMATION ACCURACY WITH THE COMPETITIVE DIRECT METHODS (POSECNN [11], PVNET [4], HYBRIDPOSE [73] AND EPRO-PNP [7]) AND THE REFINEMENT METHODS (DPOD [15], DEEPIM [13] AND REPOSE [17]) ON LINEMOD DATASET IN TERMS OF THE ADD(-S) METRIC

Method	PoseCNN	PVNet	HybridPose	EPro-PnP	DeepIM	DPOD	RePOSE		Ours	
Init. pose	-	-	-	-	PoseCNN	self-designed	PoseCNN	PVNet	PoseCNN	PVNet
Ape	25.62	43.62	63.1	-	76.95	87.73	47.4	79.5	**88.19**	85.62
Benchvise	77.11	99.90	99.9	-	97.48	98.45	88.5	**100.0**	**100.0**	**100.0**
Camera	47.25	86.86	90.4	-	93.53	96.07	67.0	**99.2**	98.04	98.43
Can	69.98	95.47	98.5	-	96.46	99.71	88.0	**99.8**	99.31	99.51
Cat	56.09	79.34	89.4	-	82.14	94.71	80.6	**97.9**	96.41	96.41
Driller	64.92	96.43	98.5	-	94.95	98.80	78.5	99.0	**99.70**	99.50
Duck	41.78	52.58	65.0	-	77.65	86.29	66.1	80.3	89.30	**89.67**
Eggbox	98.50	99.15	**100.0**	-	97.09	99.91	98.6	**100.0**	99.53	**100.0**
Glue	94.98	95.66	98.8	-	99.42	96.82	95.6	98.3	**99.71**	97.30
Holep.	52.24	81.92	89.7	-	52.81	86.87	62.7	96.9	**97.43**	97.15
Iron	70.17	98.88	**100.0**	-	98.26	**100.0**	80.3	**100.0**	**100.0**	**100.0**
Lamp	70.73	99.33	99.5	-	97.50	96.84	87.8	99.8	99.81	**100.0**
Phone	53.07	92.41	94.9	-	87.72	94.69	74.3	**98.9**	98.39	98.68
Average	63.26	86.27	91.3	95.80	88.61	95.15	78.1	96.1	**97.37**	97.10

TABLE V
ACCURACY COMPARISON WITH THE STATE-OF-THE-ART ON OCCLUSION LINEMOD DATASET IN TERMS OF THE ADD(-S) METRIC

Object	PoseCNN [11]	PVNet [4]	HybridPose [73]	GDR-Net [37]	SO-Pose [38]	DPOD [15]	RePOSE [17]	Ours
Ape	9.60	15.8	20.9	39.3	**46.3**	-	31.1	37.18
Can	45.2	63.3	75.3	79.2	81.1	-	80.0	**88.07**
Cat	0.93	16.7	24.9	23.5	18.7	-	25.6	**29.15**
Driller	41.4	65.7	70.2	71.3	71.3	-	73.1	**88.14**
Duck	19.6	25.2	27.9	44.4	43.9	-	43.0	**49.17**
Eggbox	22.0	50.2	52.4	58.2	46.6	-	51.7	**66.98**
Glue	38.5	49.6	53.8	49.3	63.3	-	54.3	**63.79**
Holep.	22.1	39.7	54.2	58.7	**62.9**	-	53.6	62.76
Average	24.9	40.8	47.5	53.0	54.3	47.3	51.6	**60.65**

where $\mathbf{X}_{\text{model}} \in \mathbb{R}^{3\times M}$ are coordinates of the model points. This loss function measures the L1 error between the transformed object models by the estimated rotation $\mathbf{R}_{init}$ and the ground-truth rotation $\mathbf{R}_{gt}$ with the best symmetry choice $\mathbf{R}_{sym}$.

Previous direct estimation methods [1], [4] depend on cumbersome networks to estimate the translation components. However, we find that RNNPose just needs a very coarse estimation for initialization. To decrease the computational burden, we estimate the initial translation according to the detected bounding box and the known object sizes. Specifically, given bounding-box center $\mathbf{c} \in \mathbb{R}^2$, bounding box diameter $s = \sqrt{h^2 + w^2}$ (h and w are the height and width of the bounding box), object diameter d, we approximate the translation vector as

$$\mathbf{t}_{init} = \frac{df}{s}\mathbf{K}^{-1}\begin{bmatrix}\mathbf{c}\\ 1\end{bmatrix}, \tag{10}$$

where f denotes the focal length obtained from the diagonal elements of the camera intrinsic matrix $\mathbf{K}$. Although this estimation is not in a strict form, it is found enough for RNNPose to perform well.

The pose $\mathbf{P}_{init}$ used to initialize the RNNPose is thereby obtained as $\mathbf{P}_{init} = \begin{bmatrix}\mathbf{R}_{init} & \mathbf{t}_{init}\\ \mathbf{0}^T & 1\end{bmatrix}$.

TABLE VI
ABLATION STUDY ON T-LESS WITH OUR POSE INITIALIZATION MODULE

	Average Recall (%)
w/ feature alignment	22.84
w/o $\hat{\mathbf{C}}'_t$ rect.	49.69
w/o 3D context $\mathbf{F}_{ctx}$	50.15
w/ 2D context	51.23
w/o similarity score	49.91
Ours	**53.86**

The evaluation is based on VSD metric [12].

IV. EXPERIMENTS

A. Experimental Setup

Implementation Details: We train all of our networks end-to-end using the Adam [68] optimizer with an initial learning rate of 10^{-4} and adjust it with the cosine annealing strategy. The weights of model alignment loss L_{ma} and descriptor loss L_d are set to 1's, while the weight of correspondence loss is set to 0.5. During training the RNNPose, we conduct 3 rendering cycles, each of which performs 4 recurrent refinement iterations. The crop size of the inputs is set to 240×240 and the feature maps used to construct the 4D correlation volume is of sizes 30×30, i.e., $H_1 = W_1 = H_2 = W_2 = 30$ ($1/8$ input image sizes) in

TABLE VII
PERFORMANCE COMPARISON WITH STATE-OF-THE-ART METHODS ON YCB-VIDEO DATASET WITH AUC ADD-(S) METRIC

Object	PoseCNN [11]	PVNet [4]	GDR-Net [37]	SO-Pose [38]	DeepIM [13]	RePose [17]	Ours		
Init. Pose	-	-	-	-	PoseCNN	PoseCNN	PoseCNN	GDR-Net	SO-Pose
002_master_chef_can	50.9	**81.6**	71.1	69.5	71.2	-	64.0	66.3	63.4
003_cracker_box	51.7	80.5	63.5	88.8	83.6	-	92.1	86.1	**94.0**
004_sugar_box	68.6	84.9	93.2	**95.0**	94.1	-	93.0	93.2	93.0
005_tomato_soup_can	66.0	78.2	88.9	94.6	86.1	-	94.7	**96.1**	96.0
006_mustard_bottle	79.9	88.3	**93.8**	92.7	91.5	-	92.9	93.5	88.5
007_tuna_fish_can	70.4	62.2	85.1	**94.3**	87.7	-	89.0	93.7	93.7
008_pudding_box	62.9	85.2	**86.5**	44.7	82.7	-	40.1	32.4	11.8
009_gelatin_box	75.2	88.7	88.5	92.6	91.9	-	**97.4**	97.2	97.3
010_potted_meat_can	59.6	65.1	72.9	80.3	76.2	-	81.3	80.5	**81.4**
011_banana	72.3	51.8	85.2	85.8	81.2	-	**93.2**	94.1	**93.2**
019_pitcher_base	52.5	91.2	94.3	98.5	90.1	-	98.8	**99.0**	98.8
021_bleach_cleanser	50.5	74.8	80.5	83.9	81.2	-	84.2	81.9	**87.7**
024_bowl*	69.7	**89.0**	84.0	84.0	81.4	-	86.6	84.4	87.7
025_mug	57.7	81.5	87.6	94.2	81.4	-	95.0	95.2	**95.3**
035_power_drill	55.1	83.4	78.7	**90.1**	85.5	-	88.1	88.4	87.9
036_wood_block*	65.8	71.5	77.3	**82.4**	81.9	-	78.3	81.0	75.7
037_scissors	35.8	54.8	43.7	49.9	60.9	-	73.1	53.5	**74.3**
040_large_marker	58.0	35.8	76.2	76.0	75.6	-	83.7	86.4	**86.9**
051_large_clamp*	49.9	66.3	69.3	90.1	74.3	-	76.1	90.0	**95.7**
052_extra_large_clamp*	47.0	53.9	73.6	94.6	73.3	-	90.5	74.7	**98.3**
061_foam_brick*	87.8	80.6	90.4	97.3	81.9	-	93.9	95.7	**99.7**
Average	61.3	73.8	80.2	84.7	81.9	80.8	85.0	84.0	**85.7**

The symbol * denotes symetric objects.

TABLE VIII
PERFORMANCE EVALUATION OF OUR 6-DOF OBJECT POSE ESTIMATION SYSTEM ON TLESS DATASET [71], WHICH IS ALSO COMPARED WITH OTHER BASELINE METHODS

Object	Brachmann *et al.* [75]	Kehl *et al.* [74]	AutoPose [12]	Pix2Pose [3]	ZebraPose [8]	Our Init.	AutoPose +RNNPose	Pix2Pose +RNNPose	Our Init. +RNNPose
1	8	7	14.30	21.97	32.57	1.48	35.72	**35.90**	21.94
2	10	10	16.89	13.01	34.32	0.46	22.81	**24.44**	16.76
3	21	18	14.42	20.98	41.23	0.13	39.86	**46.07**	37.30
4	4	24	7.73	6.99	33.95	2.03	19.26	21.62	**23.82**
5	46	23	37.45	42.35	46.34	0.79	69.39	70.00	**71.20**
6	19	10	28.10	23.44	47.72	1.02	64.25	62.66	**72.59**
7	52	0	17.20	25.57	39.96	1.41	47.11	**58.06**	55.02
8	22	2	8.73	40.08	43.14	2.01	43.85	**67.92**	55.18
9	12	11	12.47	57.83	47.17	0.81	76.82	**83.31**	83.20
10	7	17	27.09	69.16	47.40	0.35	68.16	**85.89**	82.35
11	3	5	14.10	14.80	45.51	0.00	51.40	53.65	**55.34**
12	3	1	18.76	37.32	47.12	0.00	66.55	72.20	**76.62**
13	0	0	12.99	28.67	36.59	0.36	46.84	43.34	**48.55**
14	0	9	10.10	7.44	36.55	0.00	17.67	17.67	19.31
15	0	12	29.82	43.67	42.18	0.68	67.00	68.32	**70.07**
16	5	56	31.11	44.73	46.72	0.00	65.42	**71.50**	60.37
17	3	52	40.35	64.76	47.44	0.00	79.23	**83.41**	83.28
18	54	22	30.30	43.34	46.74	0.00	65.62	76.59	**85.22**
19	38	35	3.22	0.50	41.73	0.00	42.12	33.30	**52.76**
21	39	26	4.96	0.00	**41.48**	0.00	28.85	27.48	39.49
22	19	27	4.81	0.00	37.14	0.00	**42.66**	28.97	29.44
23	61	71	11.49	4.33	46.06	0.40	51.24	51.64	**56.16**
24	1	36	24.21	16.99	45.83	5.99	**54.56**	49.72	45.83
25	16	28	26.37	0.70	46.35	0.00	**59.34**	34.67	48.44
26	27	51	29.17	0.40	47.24	1.01	48.71	42.36	**50.25**
27	17	34	11.29	53.31	42.41	0.00	64.19	82.40	**82.72**
28	13	54	23.29	40.96	44.53	0.52	43.13	**70.21**	46.61
29	6	**86**	24.06	33.70	48.98	0.00	40.56	46.42	48.47
30	5	69	37.12	51.24	44.95	0.00	51.04	**82.04**	76.31
Average	17.84	24.60	19.17	26.95	42.81	0.65	50.00	53.46	**53.86**

The evaluation is based on VSD metric [12]. The results of Brachmann et al. [70] and Kehl et al. [74] are obtained from [12].

Fig. 2. The memory consumption of this volume is around $H_1 \times W_1 \times H_2 \times W_2 \times 4$ Bytes $= 3.24$ MB, which is very limited.

For our rotation network training, we create a set of equivalent rotations S for the symmetric objects in (9). For the objects with infinite equivalent rotations, e.g., cylindrical objects, we discretize the rotation and create 64 rotation matrices that cover the continuous rotation space as the set S in practice. All the models are trained agnostic to the initial pose predictor with disturbed ground-truth poses for training following [13]. For evaluation, the numbers of rendering cycles and refinement iterations are the same as those during training for most experiments if without extra declaration, though more iterations could produce better results.

Datasets: We evaluate our method on four datasets, including LINEMOD [69], Occlusion LINEMOD [70], YCB-Video [11] and TLESS [71].

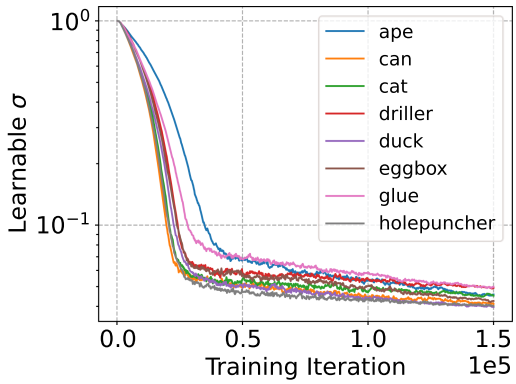


Fig. 9. Variation of the learnable parameter σ in similarity score (3) during training.

LINEMOD is a standard benchmark for 6D object pose estimation. This dataset contains cluttered texture-less objects captured with challenging illuminance variations.

Occlusion LINEMOD is a subset of LINEMOD dataset but with additional annotations for occluded objects, which is suitable for verifying the robustness to severe occlusions.

YCB-Video dataset contains the images of the YCB object set [72] where strong occlusions, clutters are exhibited. It includes more than 110 k real images captured for 21 objects with or without textures.

TLESS [71] is a benchmark that contains 30 industry-relevant textureless objects for 6D object pose estimation. TLESS contains 39 K training images and 10 K test images. Compared with LINEMOD and YCB-Video, TLESS requires an object detection step to identify the target objects before pose estimation. We follow similar conventions for data processing and synthetic training data generation adopted by the previous works [4], [17]. For the evaluation on YCB-Video and TLESS, we adopt the training data setups of [37], [73].

Evaluation Metrics: We evaluate our method with the metrics ADD-(S) [69] AUC of ADD(-S) [11], and VSD [12].

The ADD-(S) metric is based on the average point-wise distance between the transformed models, i.e., one transformed with the pose estimation and the other transformed with the ground-truth. With standard ADD-(S) metric, if the average distance is less than 10% of the model diameter, the pose estimation is regarded as correct. In some of our experiments, we also test the performances with this threshold set to 2% or 5% of the diameter for stricter testing. For symmetric objects, the average distance is computed based on closest point distances [69].

When evaluating on the YCB-Video dataset, we also compute the AUC (Area Under Curve) of ADD(-S) by varying the distance threshold from 0 cm to 10 cm following [11].

For evaluation on TLESS, we adopt VSD mertic following the previous methods [3], [12]. VSD metric measures the distance between the estimated and ground truth object depth surfaces that are visible, which is ambiguity invariant. As in the previous methods, we report the recall of correct object pose estimations at $err_{vsd} < 0.3$ with tolerance $\tau = 20$ mm and $>10\%$ object visibility.

B. Ablation Study

We conduct a thorough ablation study on LINEMOD and Occlusion LINEMOD datasets to evaluate the effectiveness of the components in our framework.

Correspondence Field Supervision: We first remove the correspondence loss to study the influence of correspondence field quality on pose estimation. The results 'w/o correspondence loss' in Table I correspond to this ablation study. The performance degrades significantly compared with our full model. Since our pose optimization is based on correspondence field estimation, the solid supervision on correspondence field estimation is essential to the overall system.

Effectiveness of the Pose Supervision and End-to-end Learning: We further remove the supervision to the pose estimation by setting the weight of the model alignment loss L_{ma} to 0. This is equivalent to adopting a typical non-differentiable LM optimizer because no gradient is backpropagated through the LM layer during training. It can be found that the object pose can still be reasonably estimated (denoted as 'w/o L_{ma}' in Table I), but with humble performance, especially with stricter evaluation criteria, i.e., by setting a smaller threshold 0.01d or 0.05d. The performance degradation reflects the importance of end-to-end pose learning. The differentiable LM layer enables the pose supervision to affect the feature learning for more robust correspondence field estimation, which is essential to our formulation.

Correspondence Field Rectification: Another key procedure in our recurrent pose refinement is the correspondence field rectification. To validate the effectiveness, we ablate this step and directly use the correspondence estimation $\hat{\mathbf{C}}_t$ from the GRU as the initialization for the next iteration (denoted as 'w/o $\hat{\mathbf{C}}'_t$ rect.' in Table I). We find that the performance drops significantly compared with our full framework, especially on more strict metrics, i.e., 0.01d and 0.05d. This phenomenon demonstrates that the corrected correspondence field with the rigid-transformation constraints from the optimized pose can facilitate the refinement in the following iterations.

3D Context Encoder: To verify the effectiveness of our 3D context encoder, we test the system without the context encoder (denoted as 'w/o 3D context $\mathbf{F}_{ctx}$') or with a commonly used 2D context encoder (denoted as 'w/ 2D context'). The performances of these two versions both degrade compared to that with a 3D context encoder. The degradation not only reveals the importance of context information as indicated by previous works [19], [20], but also proves that our 3D context encoder is a more effective choice than the 2D counterpart in our task. We reckon that the more robust performance may be attributed to the finer granularity of dense 3D point cloud features (compared with the low-resolution 2D image features). The finer-granularity features could provide more detailed geometric contexts.

Pose Optimization Based on Correspondence Field versus Feature Alignment: We formulate the object pose optimization based on the correspondence field estimated by the RNN (4). To compare our formulation to that based on feature alignment

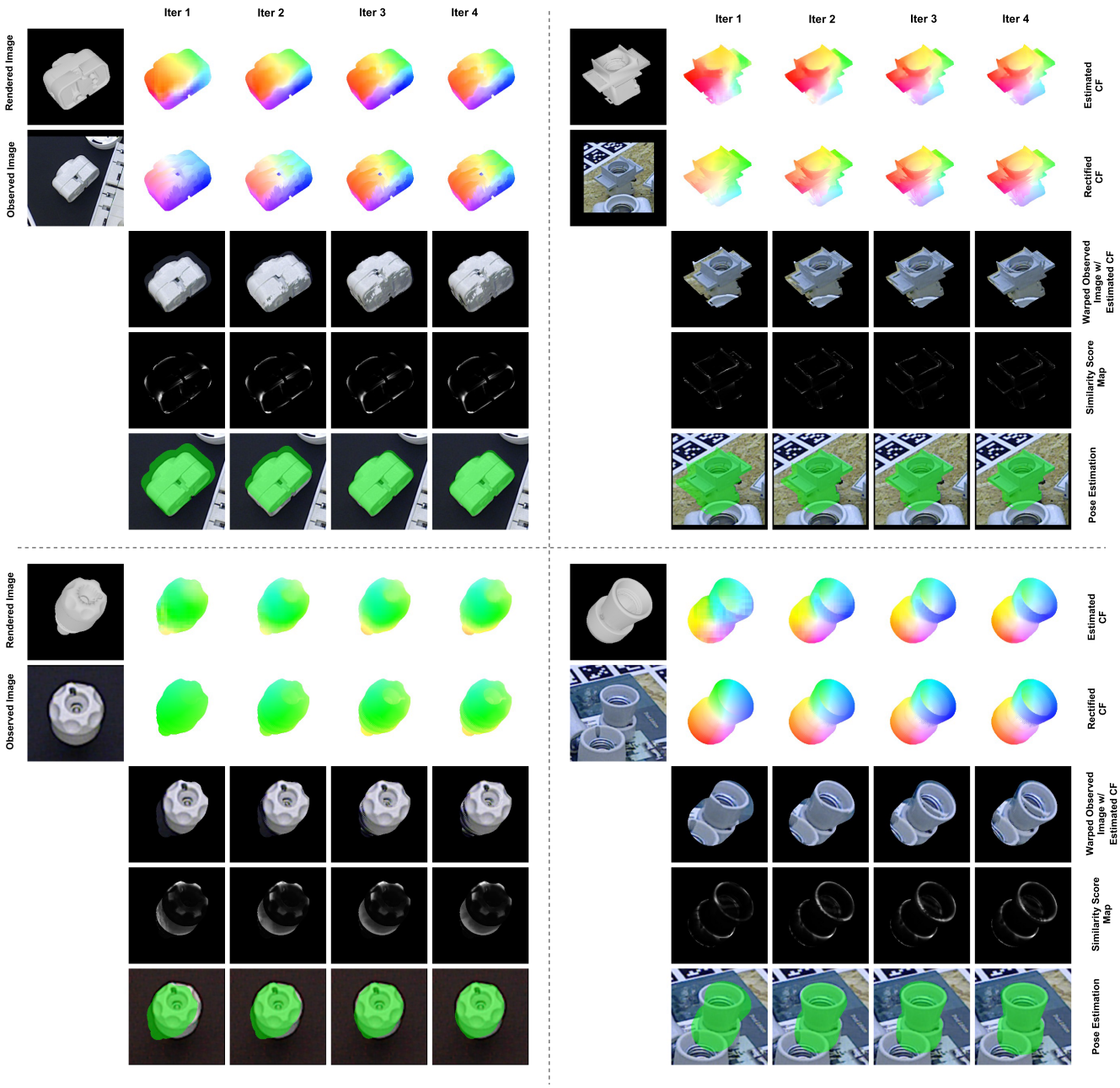


Fig. 10. Illustration of pose refinement process within one rendering cycle. We show four different cases in separate quadrants. We can find that the quality of estimated correspondence fields (CFs), rectified CFs as well as object poses are improved steadily as the refinement continues (iter1-iter4). After warping the observed image according to the estimated CFs, it can be observed that the textures of the observed images generally can be correctly mapped to the rendered objects, which demonstrates the correctness of the CF estimations.

adopted by RePose [17], we tried to fit RePose's pose optimization into our framework. The evaluation result 'w/ feature alignment' in Table VI corresponds to this variant. It can be found that the average recall is appreciably inferior to our method, i.e., 22.84% versus 53.86%.

RePose's pose optimization is conducted by aligning the projected 3D features with the observed image features. As the pose optimization depends on the gradients from the pixel-level feature differences, if the initial pose is very erroneous, the overlap between the projected features and the observed image features could be limited and thus the optimization would be affected. In contrast, the correspondence field estimation is less influenced by the pose differences between two images (thanks to RNN's large receptive field), which leads to a more robust performance via our formulation (4).

Similarity Scores for Occlusion Handling: In Table II, we evaluate the effectiveness of similar scores in occlusion handling on the Occlusion LINEMOD dataset. The version 'w/ similarity score' performs better for severely occluded objects. By including similarity scores during pose optimization, flawed correspondence estimations in the occluded unreliable regions are effectively downweighted. Fig. 9 plots the variations of

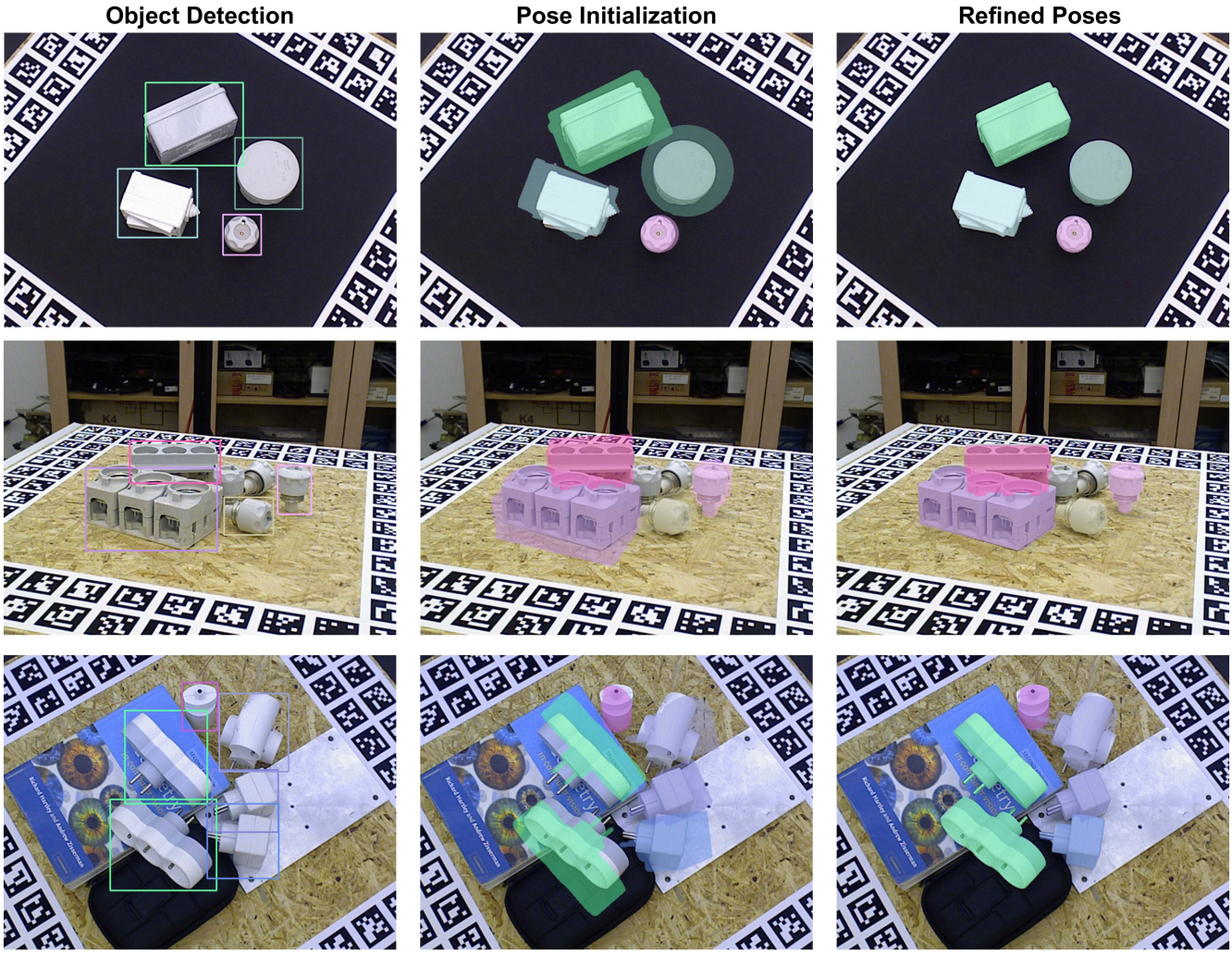


Fig. 11. Visualization of the outputs from each stage of our 6-DoF object pose estimation system. The 1-3 columns respectively exhibit the detection results, our estimated initial poses and the final pose refinement result by RNNPose. Better viewed in color.

parameter σ values (that control sharpness of the similarity score function) during training. As the training proceeds, the value of σ gradually decreases resulting in a shaper bell-shaped score function. This reflects that the descriptors become more distinctive as the training proceeds and the similarity scores become more effective in downweighting the unreliable correspondences. Some similarity score map examples are exhibited in Fig. 6 for better understanding.

Recurrent Iterations versus Rendering Cycles: The number of refinement iterations affects the system performance, especially when erroneous initial pose estimations exist. We analyze the performances with different recurrent iterations and rendering cycles in Fig. 7. From Fig. 7(a), it can be found that, by solely increasing the recurrent iterations while rendering the reference object image only once, we have achieved a high accuracy of 96.05% which is comparable to RePOSE [17]. If conducting refinement with more recurrent iterations and rendering cycles, steady improvements are reported, which reflect good convergence of our method. To further validate the robustness to erroneous initial poses, we add Gaussian noise to the initial poses. Specifically, we randomly disturb translation components and rotation Euler angles with Gaussian noise. For the rotation, we add angular noise with standard deviation (STD, denoted as σ_r) of 10° in all three axes. For the translational disturbance, we apply noise with a STD of 15 cm along the z axis (the axis perpendicular to the image plane) and STDs of 3 cm in x and y directions ($\frac{1}{5}\times$) considering current methods usually have larger variances on depth estimations. From Fig. 7(b), we find that the necessity of recurrent refinement becomes more noticeable.

Though more rendering cycles bring performance gains as well, the extra costs are significant, since most of the input features need re-encoding. Based on the runtime analysis (Table III), increasing the recurrent iterations is more economical for better performance as only the CF (correspondence field) estimation, pose optimization and CF rectification modules are activated for a recurrent iteration.

C. Comparison With State-of-the-Art Methods

We compare with the cutting-edge methods on LINEMOD, Occlusion LINEMOD, and YCB-Video.

For the LINEMOD dataset, we compare with the recent pose refinement methods RePOSE [17], DPOD [15] and DeepIM [13] as well as some direct estimation baselines [4], [7], [11], [73]. Table IV contains the comparison results and we achieve a state-of-the-art performance. Interestingly, we achieve slightly

better average performance when using PoseCNN [11] as the initial pose generator rather than the PVNet [4], although the pose accuracy of PVNet is much better as exhibited in Table IV. This phenomenon reveals the good tolerance of our system to erroneous initial poses. To test our robustness to even larger initial pose errors, we add random Gaussian pose noises to the initial rotation and translation components separately for accuracy evaluation similar to those in Section IV-B. Fig. 8 plots the accuracy variations w.r.t.the disturbance magnitudes. Our method exhibits strong robustness and works reasonably even with extremely noisy initial poses.

We also conduct comparisons on Occlusion LINEMOD. As shown in Table V, we outperform the cutting-edge method [17] by a significant margin ($51.6 \nearrow 60.65$), which manifests the system robustness to occlusions. We visualize some of our pose estimates from severely occluded images in the first row of Fig. 6, where the initial poses from PVNet are disturbed with Gaussian noise like before ($\sigma_t = 15$ cm, $\sigma_r = 10^\circ$) to pose more challenges. It is shown that our system is capable of handling large initial pose errors even in highly occluded scenarios.

We further evaluate RNNPose on the YCB-Video dataset. We use different methods as the pose initializer, including PoseCNN [1], GDR-Net [37], and SO-Pose [38], to test the generality of RNNPose. RNNPose still performs well on this large-scale dataset. The experimental results are shown in Table VII. We consistently improve the initial poses from other off-the-shelf methods and achieve a higher average accuracy. Besides, when adopting the same pose initialization methods, i.e., PoseCNN, our method outperforms the previous refinement methods DeepIM and RePose by significant margins.

D. Evaluation of Our 6-DoF Object Pose Estimation System Based on RNNPose

To test the applicability to multi-instance scenarios, we further evaluate our proposed 6-DoF object pose estimation system (Section III-D) on a more challenging dataset, i.e., TLESS dataset. Table VIII exhibits our results with the comparisons with previous counterparts. Our object detection module is based on MaskRCNN [67].

To verify the effectiveness of our pose initialization module, we separately test the performances when substituting the pose initialization module with AutoPose, Pix2Pose, and our initial pose estimation. Compared with the initial poses provided by AutoPose and Pix2Pose, our refinement technique significantly improves their performance, i.e., 'Ours+AutoPose' improves from 19.17% to 50.00%, and 'Ours+Pix2Pose' improves from 26.95% to 53.46%. Notice that, although our lightweight pose initialization module (denoted as 'Our Init.') has much inferior accuracy (merely 0.65%) compared with AutoPose and Pix2Pose, by working with the robust RNNPose, the overall performance (with an accuracy of 53.86%) even slightly surpasses the counterparts 'Ours+AutoPose' and 'Ours+Pix2Pose'. Fig. 10 shows more detailed refinement procedures within a rendering cycle. We also illustrate the intermediate outputs across different recurrent refinement iterations for better understanding. It can be observed that, even with significant initial errors, the initial poses can be steadily improved as the refinement iteration increases.

TABLE IX
RUNTIME COMPARISON OF INITIAL POSE ESTIMATION METHODS

Method	Runtime (ms)
AutoPose [12]	102.1
PixelPose [3]	99.3
PVNet [4]	112.5
ZebraPose [7]	283.1
Ours	7.9

To give a better view of the overall working pipeline, Fig. 11 additionally visualizes the outputs of different stages of our system. It is shown that our system can robustly work in cluttered scenes containing multiple instances (even of the same class). Moreover, as shown in Table IX, our pose initialization module is significantly more efficient compared with other off-the-shelf counterparts.

V. CONCLUSION

We have presented a recurrent framework for 6-DOF object pose refinement. A non-linear least squares problem is formulated for pose optimization based on the estimated correspondence field between the rendered image and the observed image. Descriptor-based consistency checking is included to downweight unreliable correspondences for occlusion handling. Our method performs robustly against erroneous pose initializations and severe occlusions, which achieves state-of-the-art performances on public datasets.

We, humans, can easily identify the correspondence of the objects in two different frames or even in different modalities but usually have difficulties directly estimating their pose difference accurately. Our framework endeavors to exploit the network's learning capability for 2D-2D and 2D-3D correspondence identification, while leaving the pose optimization to mathematics to improve the robustness. Although this paper mainly focuses on rigid object pose estimation, we hope our findings can also inspire researchers in the field of localization, odometry, human-pose modeling *etc*., where 6-DoF pose is also a focus.

Despite the robust performance, it is still undeniable that our method is still limited to the known objects similar to many other works [13], [15], [17]. In the future, we plan to extend our method to handle unknown objects for better generality.

REFERENCES

[1] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2018, *arXiv: 1711.00199*.
[2] Z. Li, G. Wang, and X. Ji, "CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 7677–7686.
[3] K. Park, T. Patten, and M. Vincze, "Pix2Pose: Pixel-wise coordinate regression of objects for 6D pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 7667–7676.

[4] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4556–4565.

[5] T. Hodan, D. Barath, and J. Matas, "EPOS: Estimating 6D pose of objects with symmetries," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11700–11709.

[6] B. Chen, A. Parra, J. Cao, N. Li, and T.-J. Chin, "End-to-end learnable geometric vision by backpropagating PnP optimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8097–8106.

[7] H. Chen, P. Wang, F. Wang, W. Tian, L. Xiong, and H. Li, "EPro-PnP: Generalized end-to-end probabilistic Perspective-N-Points for monocular object pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 2771–2780.

[8] Y. Su et al., "ZebraPose: Coarse to fine surface encoding for 6DoF object pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6728–6738.

[9] L. Quan and Z. Lan, "Linear N-point camera pose determination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 8, pp. 774–780, Aug. 1999.

[10] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *Int. J. Comput. Vis.*, vol. 81, pp. 155–166, 2009.

[11] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Proc. Int. Conf. Robot. Sci. Syst.*, 2018, pp. 1–10.

[12] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3D orientation learning for 6D object detection from RGB images," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 712–729.

[13] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "DeepIM: Deep iterative matching for 6D pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 695–711.

[14] F. Manhardt, W. Kehl, N. Navab, and F. Tombari, "Deep model-based 6D pose refinement in RGB," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 833–849.

[15] S. Zakharov, I. Shugurov, and S. Ilic, "DPOD: 6D pose object detector and refiner," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1941–1950.

[16] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.

[17] S. Iwase, X. Liu, R. Khirodkar, R. Yokota, and K. M. Kitani, "RePOSE: Fast 6D object pose refinement via deep texture rendering," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 3283–3292.

[18] G. Younes, D. Asmar, and J. Zelek, "The advantages of a joint direct and indirect VSLAM in AR," in *Proc. CVPR Workshop Comput. Vis. Augmented Virtual Reality*, Long Beach, CA, USA, 2019, pp. 1–4.

[19] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8934–8943.

[20] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 402–419.

[21] C. Tang and P. Tan, "BA-Net: Dense bundle adjustment networks," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–18.

[22] Z. Teed and J. Deng, "DeepV2D: Video to depth with differentiable structure from motion," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–20.

[23] C. Gu and X. Ren, "Discriminative mixture-of-templates for viewpoint classification," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 408–421.

[24] S. Hinterstoisser et al., "Gradient response maps for real-time detection of textureless objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 876–888, May 2012.

[25] M. Zhu et al., "Single image 3D object detection and pose estimation for grasping," in *Proc. Int. Conf. Robot. Automat.*, 2014, pp. 3936–3943.

[26] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1530–1538.

[27] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5738–5746.

[28] Y. Hu, P. Fua, W. Wang, and M. Salzmann, "Single-stage 6D object pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2927–2936.

[29] C. Wang et al., "DenseFusion: 6D object pose estimation by iterative dense fusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3338–3347.

[30] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11629–11638.

[31] K. Wada, E. Sucar, S. James, D. Lenton, and A. J. Davison, "MoreFusion: Multi-object reasoning for 6D pose estimation from volumetric fusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14528–14537.

[32] D. Chen, J. Li, Z. Wang, and K. Xu, "Learning canonical shape space for category-level 6D object pose and size estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11970–11979.

[33] A. Remus, S. D'Avella, F. Di Felice, P. Tripicchio, and C. A. Avizzano, "i2c-Net: Using instance-level neural networks for monocular category-level 6D pose estimation," *IEEE Trans. Robot. Autom.*, vol. 8, no. 3, pp. 1515–1522, Mar. 2023.

[34] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6D object pose and size estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2637–2646.

[35] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3848–3856.

[36] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 292–301.

[37] G. Wang, F. Manhardt, F. Tombari, and X. Ji, "GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16606–16616.

[38] Y. Di, F. Manhardt, G. Wang, X. Ji, N. Navab, and F. Tombari, "SO-Pose: Exploiting self-occlusion for direct 6D pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 12376–12385.

[39] H. Lin, S. Peng, Z. Zhou, and X. Zhou, "Learning to estimate object poses without real image annotations," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 1159–1165.

[40] P. Liu, Q. Zhang, and J. Cheng, "BDR6D: Bidirectional deep residual fusion network for 6D pose estimation," *IEEE Trans. Autom. Sci. Eng.*, early access, Mar. 01, 2023, doi: 10.1109/TASE.2023.3248843.

[41] Y. Liu et al., "Gen6D: Generalizable model-free 6-DoF object pose estimation from RGB images," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 298–315.

[42] J. Sun et al., "OnePose: One-shot object pose estimation without CAD models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6815–6824.

[43] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner, "Zero-shot category-level object pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 516–532.

[44] X. He, J. Sun, Y. Wang, D. Huang, H. Bao, and X. Zhou, "OnePose++: Keypoint-free one-shot object pose estimation without CAD models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 35103–35115.

[45] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox, "Self-supervised 6D object pose estimation for robot manipulation," in *Proc. Int. Conf. Robot. Automat.*, 2020, pp. 3665–3671.

[46] G. Wang, F. Manhardt, X. Liu, X. Ji, and F. Tombari, "Occlusion-aware self-supervised monocular 6D object pose estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Dec. 17, 2021, doi: 10.1109/TPAMI.2021.3136301.

[47] L. Yen-Chen, P. Florence, J. T. Barron, T.-Y. Lin, A. Rodriguez, and P. Isola, "NERF-supervision: Learning dense object descriptors from neural radiance fields," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 6496–6503.

[48] A. Grabner et al., "Geometric correspondence fields: Learned differentiable rendering for 3D pose refinement in the wild," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 102–119.

[49] M. Meloun, and J. Militkỳ, *Statistical Data Analysis: A Practical Guide*. Sawston, U.K.: Woodhead Publishing Limited, 2011.

[50] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Proc. Biennial Conf. Numer. Anal.*, Springer, 1978, pp. 105–116.

[51] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[52] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "BARF: Bundle-adjusting neural radiance fields," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 5721–5731.

[53] Y. Xu, J. Lin, J. Shi, G. Zhang, X. Wang, and H. Li, "Robust self-supervised LiDAR odometry via representative structure discovery and 3D inherent error modeling," *IEEE Trans. Robot. Autom.*, vol. 7, no. 2, pp. 1651–1658, Apr. 2022.

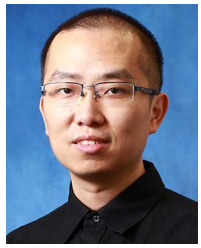
[54] Z. Huang et al., "VS-Net: Voting with segmentation for visual localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6097–6107.
[55] L. Von Stumberg, P. Wenzel, Q. Khan, and D. Cremers, "GN-Net: The Gauss-Newton loss for multi-weather relocalization," *IEEE Trans. Robot. Autom.*, vol. 5, no. 2, pp. 890–897, Apr. 2020.
[56] P.-E. Sarlin et al., "Back to the feature: Learning robust camera localization from pixels to pose," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3246–3256.
[57] A. Dosovitskiy et al., "FlowNet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2758–2766.
[58] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1647–1655.
[59] T.-W. Hui, X. Tang, and C. C. Loy, "LiteFlowNet: A lightweight convolutional neural network for optical flow estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8981–8989.
[60] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[61] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 6410–6419.
[62] N. Ravi et al., "Accelerating 3D deep learning with PyTorch3D," 2020, *arXiv: 2007.08501*.
[63] P. Liu, M. Lyu, I. King, and J. Xu, "SelFlow: Self-supervised learning of optical flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4566–4575.
[64] S. Meister, J. Hur, and S. Roth, "UnFlow: Unsupervised learning of optical flow with a bidirectional census loss," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 7251–7259.
[65] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 337–33712.
[66] Y. Sun et al., "Circle loss: A unified perspective of pair similarity optimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6397–6406.
[67] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.
[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[69] S. Hinterstoisser et al., "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes," in *Proc. Asian Conf. Comput. Vis.*, 2012, pp. 548–562.
[70] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6D object pose estimation using 3D object coordinates," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 536–551.
[71] T. Hodan, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2017, pp. 880–888.
[72] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *Proc. Int. Conf. Robot. Automat.*, 2015, pp. 510–517.
[73] C. Song, J. Song, and Q. Huang, "HybridPose: 6D object pose estimation under hybrid representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 428–437.
[74] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, "Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 205–220.
[75] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, and C. Rother, "Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3364–3372.

Yan Xu received the BS degree in information science and engineering from Southeast University, in 2015, and the MS degree in computer science and technology from Zhejiang University, in 2018. He is currently working toward the PhD degree with the Department of Electronic Engineering, Chinese University of Hong Kong. His research interests include 3D computer vision, robotics, and deep learning.

Kwan-Yee Lin received the PhD degree from the Department of Information Science, School of Mathematical Sciences, Peking University, China. She is currently a post-doc researcher with the Department of Electronic Engineering, Chinese University of Hong Kong. Her research interests lie at the intersection of computer vision, computer graphics, and machine learning.

Guofeng Zhang received the BS and PhD degrees in computer science from Zhejiang University, in 2003 and 2009, respectively. He is now a professor with the State Key Lab of CAD&CG, Zhejiang University. He received the National Excellent Doctoral Dissertation Award, the Excellent Doctoral Dissertation Award of China Computer Federation and the best paper award of ISMAR 2020. His research interests include structure-from-motion, SLAM, 3D reconstruction, augmented reality, video segmentation, and editing.

Xiaogang Wang received the BS degree from the University of Science and Technology of China, in 2001, the MS degree from the Chinese University of Hong Kong, in 2003, and the PhD degree from the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, in 2009. He is currently a professor with the Department of Electronic Engineering, Chinese University of Hong Kong. His research interests include computer vision and machine learning.



Hongsheng Li received the BS degree in automation from the East China University of Science and Technology, Shanghai, China, in 2006, and the MS and PhD degrees in computer science from Lehigh University, Bethlehem, Pennsylvania, in 2010 and 2012, respectively. He is currently an associate professor with the Department of Electronic Engineering, Chinese University of Hong Kong. His research interests include computer vision, medical image analysis, and machine learning.