Shape-Constraint Recurrent Flow for 6D Object Pose Estimation

Yang Hai [1], Rui Song [1], Jiaojiao Li [1], Yinlin Hu [2]
[1] Xidian University, [2] MagicLeap
yanghai1218@gmail.com, {rsong, jjli}@xidian.edu.cn, yhu@magicleap.com

Abstract

Most recent 6D object pose estimation methods rely on 2D optical flow networks to refine their results. However, the optical flow methods typically do not consider any 3D shape information of the targets during matching, making them suffer in 6D object pose estimation. In this work, we propose a shape-constraint recurrent flow network for 6D object pose estimation, which embeds the 3D shape information of the targets into the matching procedure. We first introduce a flow-to-pose component to learn an intermediate pose from the current flow estimation, then impose a shape constraint from the current pose on the lookup space of the 4D correlation volume for flow estimation, which reduces the matching space significantly and is much easier to learn. Finally, we optimize the flow and pose simultaneously in a recurrent manner until convergence. Our results on three challenging 6D object pose datasets show that our method outperforms the state of the art significantly in both accuracy and efficiency.

1. Introduction

6D object pose estimation, *i.e.*, estimating the 3D rotation and 3D translation of a target object with respect to the camera, is a fundamental problem in 3D computer vision and also a crucial component in many applications, including robotic manipulation [8] and augmented reality [35]. Most recent methods rely on pose refinement to obtain accurate pose results [17, 32, 52]. Typically, they first synthesize an image based on some rendering techniques [9, 39] according to the initial pose, then estimate dense 2D-to-2D correspondence between the rendered images and the input based on optical flow networks [47]. After lifting the 2D flow to 3D-to-2D correspondence based on the target's 3D shape information, they can obtain a new refined pose based on some Perspective-n-Points (PnP) solvers [4, 18, 28].

Although this paradigm works well in general, it suffers from several weaknesses. First, the flow network they use is built on top of two assumptions, *i.e.*, the brightness consistency between the two potential matches and the smoothness of matches within a local neighbor [1], which is general but does not have any clue about the target's shape in the context of 6D object pose estimation, making the potential matching space of every pixel unnecessarily large in the target image caused by the missing of shape constraint. Second, this missing of shape information during matching will result in flow results that do not respect the target shape, which will introduce significant matching noise to the following PnP solvers, as shown in Fig. 1. Third, this multi-stage paradigm trains a network that relies on a surrogate matching loss that does not directly reflect the final 6D pose estimation task [18], which is not end-to-end trainable and suboptimal.

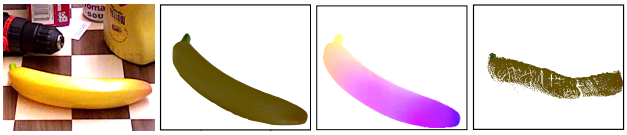
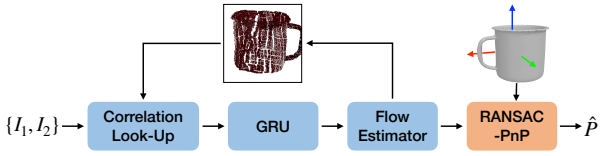
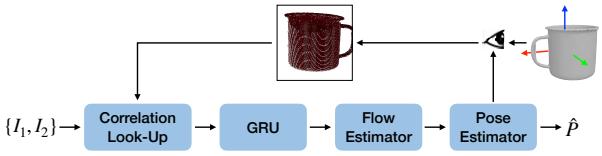


Figure 1. **Problem of 2D flow in 6D object pose estimation.** Given an initial pose, one can estimate the dense 2D-to-2D correspondence (optical flow) between the input and the synthetic image rendered from the initial pose, and then lift the 2D flow to 3D-to-2D correspondence [17] which can be used to compute a new pose by PnP solvers [4, 28]. However, the flow estimation does not take the target shape into account, as illustrated by the warped flow in the last subfigure, which will introduce significant matching noise to pose solvers and is thus suboptimal in the object pose context.

To address these problems, we propose a shape-constraint recurrent matching framework for 6D object pose refinement. It is built on top of the intuition that the dense 2D matching should comply with the 3D shape of the target, in addition to the brightness consistency and smoothness assumption in classic optical flow solutions [2, 36]. We first build a 4D correlation volume between every pixel of the source image and every pixel of the target image, similar to RAFT [47]. However, unlike indexing from the correlation volume according to the current flow, we propose only indexing the correlation volume based on the 2D reprojection



(a) The standard strategy



(b) Our strategy

Figure 2. **Different pose refinement paradigms. (a)** Most pose refinement methods rely on RAFT [47] to iteratively estimate dense 2D flow based on the correlation map constructed from the flow result of the last iteration. After the convergence of the flow network, the new refined pose can be obtained by PnP solvers via the lifting of 2D flow to 3D-to-2D correspondence. This strategy has a large matching space for every pixel in constructing the correlation maps, and optimizes a surrogate matching loss that does not directly reflect the final 6D pose estimation task. **(b)** By contrast, we propose simultaneously optimizing the pose and flow in an end-to-end recurrent framework with the guidance of the target's shape. We impose a shape constraint on constructing the correlation map for flow estimation, reducing the matching space significantly, as shown in the upper part of the figure. On the other hand, we propose to directly learn the input images' pose difference from the current flow between them, yielding a system end-to-end trainable.

of the target object according to an estimated pose, which reduces the matching space significantly for predicting the flow. On the other hand, for every iteration, we learn the 6D pose difference between the input images from the estimated 2D flow between them directly, which removes the necessity of explicit PnP solvers, making our system end-to-end trainable and also more efficient, as shown in Fig. 2(b).

To demonstrate the effectiveness of our method, we evaluate our framework on the challenging 6D object pose benchmarks, including LINEMOD [14], Occluded-LINEMOD [26], and YCB-V [50]. Our framework significantly outperforms most other methods in both accuracy and efficiency.

2. Related Work

Object pose estimation, has shown significant improvement in both robustness and accuracy [37, 48, 50] after the utilizing of well-designed convolution neural networks [13, 51]. However, they still follow the traditional paradigm, which consists of the establishment of 3D-to-2D correspondence and PnP solvers. Most of the recent methods create the correspondence either by making the network predict 2D points of some predefined 3D points [19, 22, 37, 38] or predicting the corresponding 3D points for every 2D pixel locations [3, 10, 30, 43, 49, 53]. They then rely on either numerical PnP solvers [28] or learning the pose results [5, 10, 18, 49] from intermediate correspondences directly. Although these methods work well in general, they still have some accuracy problems in practice, caused mainly by the complexity of clutter scenes in typical 6D object pose scenarios. We propose to use pose refinement to obtain more accurate results in this work.

Object pose refinement, previously is dominated by methods relying on some additional depth image [30, 46, 48, 50], which is accurate but the depth image they rely on usually is inaccessible for many applications [22, 41]. Most recent refinement methods use a render-and-compare strategy without any access to the depth data and achieve comparable, or even better, performance [17, 24, 27, 29, 32, 34, 38, 52, 53]. These methods, however, formulate the pose refinement as a pure 2D-to-2D matching problem without any constraints, which is suboptimal in 6D object pose estimation. By contrast, we propose a shape-constraint recurrent matching framework guided by the 3D shape of the target in each iteration, which we will show outperforms most existing methods in both accuracy and efficiency.

Optical flow estimation, whose goal is to obtain the matching of every pixel from the source image to the target image, has been widely studied for a long time [1]. Classically, it is formulated as an energy optimization problem, which is usually based on the assumption of brightness consistency and local smoothness [1, 6, 20, 21, 40]. Recently, learning-based methods inspired by those traditional intuitions have shown great progress in estimating flow in large-displacement and occlusion scenarios [1, 11, 23, 25, 31, 44, 45, 47]. However, these general optical flow methods do not consider the task properties when used in the 6D object pose context, especially the fact that the flow should also comply with the 3D shape of the target in addition to the two standard assumptions discussed above, which should reduce the matching space significantly. We therefore propose embedding this shape prior knowledge into the flow estimation in the context of 6D object pose estimation. Our experiments demonstrate the advantages of our method in both optical flow and 6D object pose estimation.

3. Approach

Given an RGB input image, the 3D model of the target, and also the camera's intrinsic matrix, our goal is to estimate the 3D rotation and 3D translation of the target with respect to the camera. Our approach consists of two steps, including estimating the pose initialization based on some existing pose methods [22, 50], and refining it based on a recurrent framework. We focus on the second step of refine-

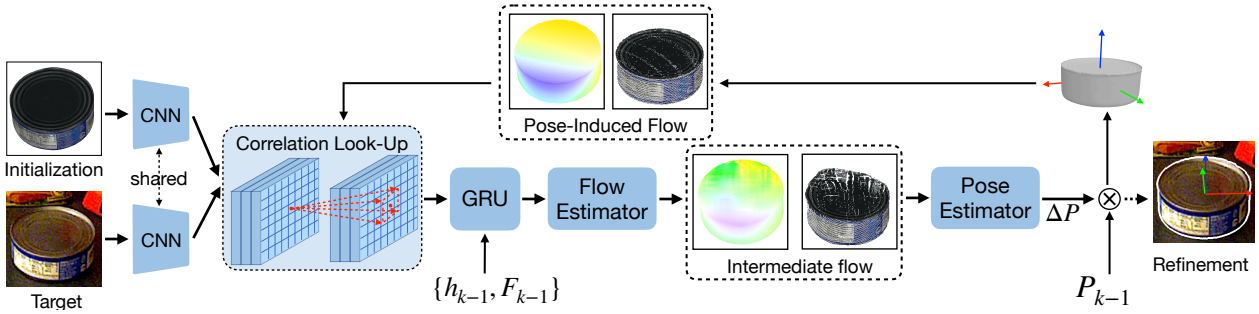


Figure 3. **Overview of our shape-constraint recurrent framework.** GRU takes the previously estimated flow F_{k-1} to update its hidden state h_{k-1}, and the updated hidden state h_k will be used to predict the intermediate residual flow ΔF_k, also as a part of the pose estimator's input to predict the relative pose ΔP. After applying ΔP to the previously estimated pose P_{k-1}, we obtain the currently estimated pose P_k and drive the currently estimated flow F_k, *i.e.*, the pose-induced flow. We show both the flow and the results of flow wrapping in the dashed boxes. To better visualization, we add the intermediate residual flow to F_{k-1} as the intermediate flow here. Note how the pose-induced flow preserves the shape, but the intermediate flow does not.

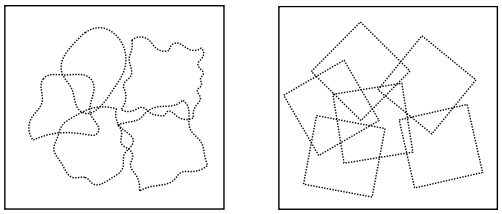
ment in this section. We first have an overview of our recurrent framework, and then discuss how we reduce the matching space by using a shape-constraint correlation space. Finally, we present how we learn the pose differences from optical flow to make our refinement framework end-to-end trainable.

3.1. Overview

Inspired by RAFT [47], our framework mainly consists of 5 components: feature extraction, correlation volume construction, Gated Recurrent Unit (GRU) [7], flow estimator, and pose estimator. We first synthesize an image based on rendering the target according to the initialization pose, and use a shared-weight CNN to extract features for both the rendered and input image. We then build a 4D correlation volume containing correlations of all pairs of feature vectors between the rendered and input image. We optimize the optical flow between the input images and estimate a pose difference based on the predicted optical low simultaneously. For every iteration, we use a pose-induced flow, which is computed based on the 3D shape of the target and the predicted pose difference from the last iteration, to index from the 4D correlation volume, facilitating the estimation of optical flow and pose difference for the next iteration. Fig. 3 provides an overview of our approach.

3.2. Shape-Constraint Correlation Space

After obtaining the 4D correlation volume $\mathbf{C} \in \mathbb{R}^{H\times W\times H\times W}$ based on the dot product between all pairs of feature vectors from image features from different pyramid levels [47], the next critical step is to index from the 4D correlation volume according to the current flow estimation to update the recurrent framework for the next iteration. The standard lookup operation $L_{\mathbf{C}}$ generates a feature map by indexing from the correlation volume. With a current estimate of optical flow $\mathbf{f}(x, y)$, it maps each pixel $\mathbf{x} = (u, v)$ in I_1 to its estimated correspondence in I_2:



(a) Standard (b) Shape-constraint

Figure 4. **Illustration of shape-constraint correlation space.** After constructing the 4D correlation volume between the two input images, we impose a constraint on the operation space of the recurrent indexing procedure, making it contains only all the 2D reprojections of the current 3D shape, which reduces the matching space significantly compared with the standard strategy.

$\mathbf{x}' = (u, v) + f(u, v)$. The new feature vector is obtained from bilinear sampling since the new location $\mathbf{x}'$ usually is not on a grid location. This standard lookup operation works well in general, but does not consider the fact that all the matches should comply with the shape of the current object, making its matching space unnecessarily large.

We propose to embed the 3D shape information in the lookup operation, making the newly mapped location depends only on the shape and pose changing of the target:

$$\mathbf{x}' = (u, v) + f(u, v; \mathbf{K}, \mathbf{S}, \mathbf{P}_1, \mathbf{P}_2), \tag{1}$$

where $\mathbf{K}$ is the camera intrinsic matrix, $\mathbf{S}$ is the 3D shape of the object, $\mathbf{P}_1$ and $\mathbf{P}_2$ are the initial pose and the currently estimated pose of the target object, respectively. We call the flow fields $\mathbf{f}(x, y; \mathbf{K}, \mathbf{S}, \mathbf{P}_1, \mathbf{P}_2)$ pose-induced flow, and it can be easily computed by some perspective geometry reasoning [17]. This pose-induced flow is the foundation of our shape-constraint correlation space, since it embeds the 3D shape information of the target object and constrains the size of the candidate space of all lookup operations, as illustrated in Fig. 4.

3.3. Learning Pose Difference From Flow

The pose-induced flow relies on both the initial and currently estimated pose of the target object. To achieve this, we propose to use a small deep regressor to learn the current pose. Instead of predicting the absolute current pose, we learn a residual pose $\Delta\mathbf{P}_k$ between the current pose and the estimated pose from the last iteration, and update the current pose $\mathbf{P}_k$ progressively:

$$\mathbf{P}_k = \mathbf{P}_{k-1} \otimes \Delta\mathbf{P}_k. \tag{2}$$

Instead of learning the residual pose $\Delta\mathbf{P}_k$ from the correlation maps directly, we use the same GRU and flow decoder as in RAFT [47] to obtain an intermediate flow first and then use a small network to regress the $\Delta\mathbf{P}_k$ based on the intermediate flow, as shown in Fig. 3.

To make the residual pose easy to learn, we represent the residual pose in the disentangled coordinate frame [29], and parameterize the residual rotation ΔR in a six-dimensional representation as in [54], which shows better continuous properties, and benefits the neural network's learning [27, 49].

For the residual translation ΔT, we encode it into $(\Delta x, \Delta y, \Delta z)$, such that the updated translation t_k is computed as following:

$$\begin{aligned} z_k &= z_{k-1}/e^{\Delta z} \\ x_k &= z_k(\Delta x/f_x + x_{k-1}/z_{k-1}) \\ y_k &= z_k(\Delta y/f_y + y_{k-1}/z_{k-1}) \end{aligned} \tag{3}$$

where Δx and Δy stand for how many pixels should move along the image x-axis and y-axis, and Δz denotes the relative scale change in the depth. As the depth estimation is relatively harder, in practice, we discard the gradients from x_k and y_k for Δz for a stable depth optimization. Fig. 5 shows the advantages of the proposed method over the baseline RAFT method.

3.4. Implementation Details

Losses. We optimize the flow estimation and pose estimation simultaneously in our recurrent matching framework. Since our predicted pose and optical flow are both from recurrent steps, we supervise each element in the sequence with exponentially increasing weights:

$$\mathcal{L} = \sum_{k}^{N} \gamma^{N-k}(\mathcal{L}^k_{pose} + \alpha * \mathcal{L}^k_{flow}) \tag{4}$$

where N is the total recurrent numbers. We set $\gamma = 0.8$ and $\alpha = 0.1$ in our experiments to balance the flow estimation and pose estimation. For flow supervision, we apply the residual flow ΔF_k from the flow estimator to the shape-aware flow F_{k-1} driven from P_{k-1}, and minimize the standard L1 end point error with the ground truth flow F_{gt}:

$$\mathcal{L}^k_{flow} = ||F_{gt} - F_{k-1} - \Delta F_k||_1 \tag{5}$$

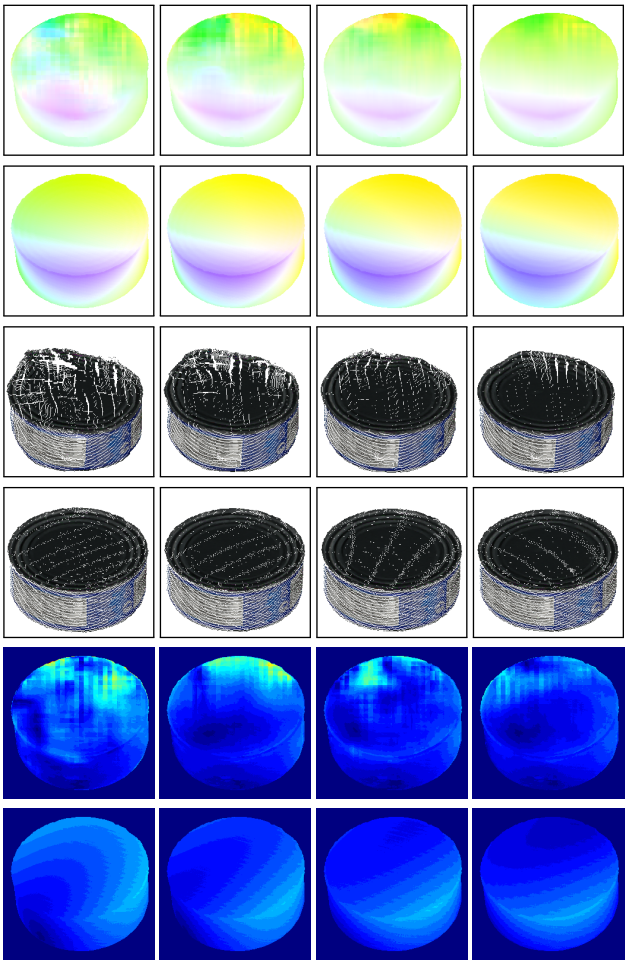


Figure 5. **Comparison of the obtained flow with different iterations.** We compare the baseline RAFT method and our method with different iteration numbers, from left to right. We show their flow, flow warp, and flow error from top to bottom, respectively. Although the baseline improves the results with more iterations, it still can not preserve the object's shape well by the end. By contrast, our method preserves the object's shape from the very beginning of the iteration.

where F_{gt} is computed from the established 2D-3D correspondences in the rendered image and the ground truth pose P_{gt} [17]. We only supervise the pixels within the rendered object mask, and discard the pixels without correspondence due to occlusion.

We use the widely used PointMatching Loss [27, 29, 52] as our pose loss. We randomly sample $n = 1000$ points from the object's mesh vertices and compute the distance of these points transformed by the ground truth pose $\mathbf{P}_{gt}$ and predicted pose $\mathbf{P}_k$. As suggested by [16, 47], we detach the gradient from $\mathbf{P}_{k-1}$ for both flow and pose supervision.

Input. To synthesize the initialization image, we use Py-

torch3D [39] to render the object according to the initial pose, with a fixed resolution of 256×256. Accordingly, we crop the region of interest from the input image based on the initial pose, and then resize it to the same resolution as the rendered image.

GRU. We recurrently optimize the estimated pose P_k towards the target pose. Similar to RAFT, we use GRU to model these recurrent sequences, which takes the concatenation of correlation feature F_c and flow F_{k-1} as the input x_k to update its hidden state h_{k-1} based on a gated activation:

$$z_k = \sigma(\text{Conv}([h_{k-1}, x_k])) \tag{6}$$

$$r_k = \sigma(\text{Conv}([h_{k-1}, x_k])) \tag{7}$$

$$\hat{h_k} = \tanh(\text{Conv}([r_k \odot h_{k-1}, x_k])) \tag{8}$$

$$h_k = (1 - z_k) \odot h_{k-1} + z_k \odot \hat{h_k}. \tag{9}$$

Here Conv denotes two 3×3 convolutions.

Flow and pose estimator. We use the GRU's updated hidden feature h_k to predict a residual flow ΔF_k. For the pose estimator, we take the concatenation of intermediate residual flow and updated hidden feature as input and use three 3×3 convolutions plus two fully connected layers to predict the residual pose ΔR and ΔT, respectively.

4. Experiments

In this section, we first introduce the experiment settings and then compare our method to the state of the art, showing that our method outperforms most of them in both accuracy and efficiency. Finally, we conduct extensive experiments systematically demonstrate the effectiveness of our method.

4.1. Experiment Setup

Datasets. We evaluate our method on three challenging dataset: LINEMOD("LM") [14], Occluded-LINEMOD("LM-O") [26] and YCB-V [50].

LINEMOD dataset contains 13 sequences, each containing a single object annotated with the ground-truth pose. Occluded-LINEMOD dataset comprises 8 objects which is a subset of the LINEMOD objects. Its test set is one of the sequences in LINEMOD, which annotates 8 objects. There are no standard experiment settings on LM and LM-O. Some previous methods [10, 49, 52] use different training settings for LM and LM-O, and some methods train a separated model for every single object [24, 37]. However, for consistency and simplicity, we train a single model for all the objects on LM and LM-O. Specifically, for each sequence in LINEMOD, we use ≈15% of the RGB images for training, resulting in a total of 2.4k images. To simulate the occlusion in LM-O, we crop the objects in OpenGL-rendered images and paste them to the target object region.

YCB-V is a larger dataset that contains 130k real images for 21 objects, which are captured in cluttered scenes, and the objects appear texture-less feature, making it challenging. Since BOP [15] provides around 50k physically-based rendering(PBR) images [9] with accurate annotations and more severe occlusion for all the above datasets, we will also use some PBR images for training, similar to most recent methods [17, 32, 43].

Implementation details. For all the experiments, we train the model using AdamW [33] optimizer with a batch size of 16. We use an adaptive learning rate scheduler based on One-Cycle [42], starting from 4e-4. We use pre-trained components from RAFT trained with the Flyingchairs [11] and Flyingthings3D [36] datasets. For the newly added components that RAFT does not have, we randomly initialize them. We train the model by 100k iterations when using only real training images. When using the mixed real and PBR training images, we sample them to make their ratio 1:1 and double the iterations to 200k.

When training, we generate the initial poses in a testing-agnostic manner [29]. We add a Gaussian noise $\mathcal{N}(0, 15)$ for each axis's rotation angle. For translation, we add a Gaussian noise $\mathcal{N}(0, 20)$ for x-axis and y-axis, and a Gaussian noise $\mathcal{N}(0, 50)$ for z-axis, since the estimated depth is less accurate. We apply the standard color-space augmentations for the target image [17, 22], including random noise and color jitter. We update 8 times for both training and testing by default.

Evaluation metrics. We use the standard ADD(-S) metric to report the performance, which computes the mean distance between the mesh vertices transformed by the predicted pose and the ground-truth pose. We report most of the numbers in ADD-0.1d, which considers the pose right if the distance is less than 10% of the mesh diameter. In some ablation experiments, we also report ADD-0.05d with the threshold as 5% of the model diameter. We also report the overall performance under the BOP [15] benchmark, which computes three errors, including the Visible Surface Discrepancy (VSD), the Maximum Symmetry-aware Surface Distance (MSSD), and the Maximum Symmetry-aware Projection Distance (MSPD), and average them as the final metric. We refer readers to [15] for a detailed definition of them.

4.2. Comparison to the State of the Art

We now compare our method with the state-of-the-art ones on different datasets. The overall results are shown in Table. 1. On LINEMOD, most pose refinement methods use PoseCNN [50] as the initial pose, except RePose [24] using PVNet [37]. We outperform all the competitor methods by a large margin. For Occluded-LINEMOD, the initial pose we use is slightly worse than PVNet used by other methods, while it gets 35.8% on ADD-0.1d metric. Nevertheless, we still outperform all other methods. On YCB-V, all the pose refinement methods use PoseCNN as the initialization. To

Dataset	PoseCNN	PVNet	SO-Pose	DeepIM	RePose	RNNPose	PFA	**Ours**
LM	63.3	86.3	96.0	88.6	96.1	97.4	95.8	**99.3**
LM-O	24.9	40.8	62.3	55.5	51.6	60.7	65.3	**73.1**
YCB-V	21.3	-	56.8	53.6	62.1	66.4	62.8	**73.2**

Table 1. **Comparison against the state of the art in ADD-0.1d.** Our method outperforms the baseline PFA (RAFT-based) and all other competitor methods by a large margin.

Method	Avg.	MSPD	VSD	MSSD
YCB-V (Real+PBR)				
Ours	**0.826**	**0.860**	0.778	0.840
RAFT	0.795	0.844	0.743	0.797
CIR	0.824	0.852	**0.783**	0.835
Cosypose	0.821	0.850	0.772	**0.842**
SurfEmb	0.781	-	-	-
YCB-V (PBR)				
Ours	**0.651**	0.769	**0.556**	**0.626**
RAFT	0.615	0.739	0.521	0.585
SurfEmb	0.647	**0.773**	0.548	0.620
Cosypose	0.574	0.653	0.516	0.554
LM-O (PBR)				
Ours	**0.682**	0.842	**0.532**	**0.674**
RAFT	0.674	0.818	0.531	0.673
CIR	0.655	0.831	0.501	0.633
SurfEmb	0.647	**0.851**	0.497	0.640
Cosypose	0.633	0.812	0.480	0.606

Table 2. **Comparison in BOP metrics.** We compare our method with the state-of-the-art pose refinement methods SurfEmb [12], CIR [32], and Cosypose [27]. All the method trains a single model for a dataset.

Method	CIR	Cosypose	SurfEmb	PFA	Ours
Timing	11k	20	1k	37	**17**

Table 3. **Efficiency comparison.** We run all the methods on the same workstation, and report the running time in milliseconds in processing an image containing only one object instance. Our method is the most efficient one among all the competitors.

make our method comparable to some methods having results only on the BOP benchmark [15], we also report the numbers of our method in BOP metrics, as shown in Table 2. Following the standard BOP setting, we only report the results obtained with PBR training images for LM-O. For a fair comparison, we use the same initialization pose as Cosypose [27]. Our method significantly outperforms the baseline RAFT method and also all other competitor methods.

Method	LM	LM-O	YCB-V
WDR-Pose	48.8	35.8	27.4
w/ RAFT	92.4	65.3	55.3
w/ **Ours**	**95.8**	**71.1**	**65.5**
PoseCNN	62.6	16.2	24.9
w/ RAFT	96.6	57.2	61.9
w/ **Ours**	**99.3**	**66.4**	**70.3**

Table 4. **Refinement comparison with different initializations.** We compare our method with the baseline method PFA [17] in refining two different versions of pose initialization, including WDR-Pose [22] and PoseCNN [50], respectively. Our method is consistently better than PFA across all three datasets. Here we report numbers in ADD-0.1d.

Running time analysis. We evaluate our method on a workstation with an NVIDIA RTX-3090 GPU and an Intel-Xeon CPU with 12 2.1GHz cores. Our method takes about 17ms for a single object. We find that the baseline PFA method takes about 36ms, which is basically based on the RAFT method. However, it needs an additional RANSAC-PnP to obtain the final pose, which requires about 14ms.

We further evaluate most recent refinement methods on the same workstation. For a fair comparison, we only take the running time of pose refinement into account, and ignore the running time of object detection and pose initialization, since different methods use very different detection or initialization strategies. Table 3 summarizes the results, our method is much more efficient than most refinement methods.

4.3. Ablation Study

Different pose initialization. In principle, our pose refinement method is general and can be used with most pose methods providing reasonable pose initializations. To verify the generalization ability of our method to different initial poses, we evaluate our refinement method with different initial poses from WDR [22] and PoseCNN [50], respectively. These two methods are common initialization settings used in other methods [17, 29, 52]. WDR is a typical keypoint-based pose estimation method, while PoseCNN is

P	L	E	YCB-V 0.05d	YCB-V 0.1d
-	-	-	33.5	61.9
✓	-	-	33.2	63.2
✓	-	✓	34.0	63.6
✓	✓	-	46.1	67.6
✓	✓	✓	**50.4**	**70.3**

Table 5. **Evaluation of different components on YCB-V.** "P" represents the use of the proposed pose estimator based on intermediate flow, "L" represents the proposed lookup operation guided by the pose results of the pose estimator, and "E" represents the utilization of the end-to-end training to obtain the final pose. For the numbers without "E", we use RANSAC-EPnP [28] to obtain the pose results. The first row is the baseline RAFT [47]. Our method outperforms the baseline by a large margin.

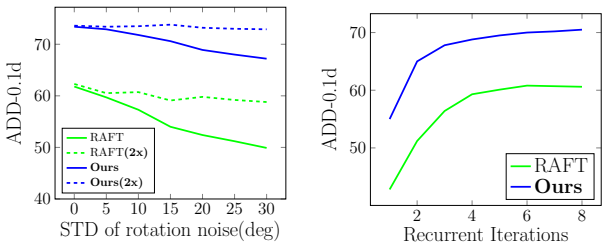
a typical method directly regressing the pose, making them exhibit different accuracy and properties. For the WDR, we obtain the initialization result trained only on the PBR images, similar as in PFA [17]. For PoseCNN, we use their offline generated results, which was trained on real images. Table 4 reports the results. Our method significantly improves the accuracy of the initial pose and outperforms the baseline RAFT method consistently.

Additionally, we compare the performance of our method and RAFT when adding different noise levels to the pose initialization, as shown in Fig. 6(a). Our method is much more robust than RAFT in scenarios with heavy initialization noise.

Convergence analysis. We compare our method with RAFT with different recurrent iterations during inference. Our method outperforms RAFT after only 2 iterations, as shown Fig. 6(b). Furthermore, we study their convergence properties during training, as shown in Fig. 7. After only 20k training steps, our method achieves similar performance as RAFT trained with 100k steps

Evaluation of different components. We study the effect of the proposed components, including the pose estimator based on intermediate residual flow, and the shape-constraint lookup operation guided by the results of the pose estimator, as shown in Table 5. The first row is basically the baseline method, which relies on RAFT to obtain correspondences first and then computes the final pose by RANSAC-EPnP [28]. We can see that the shape-constraint lookup operation is the most critical component for the boost of performance. Additionally, obtaining the final pose from the pose estimator directly rather than computing it from PnP Solvers increases the performance further.

Evaluation with different data setting. We compare our method with the baseline RAFT in different training data settings, including only PBR images, PBR images with



(a) Robustness (b) Convergence

Figure 6. **Ablation study on YCB-V. (a)** we compare the performance of our method and RAFT with different noise levels of pose initialization. For completeness, we also evaluate a version with a second time of refinement based on the results from the first one, as denoted by "2x". Our method is much more robust than RAFT when facing heavy initialization noise. **(b)** we evaluate the methods with different recurrent iterations during inference, and our method outperforms RAFT after only 2 iterations.

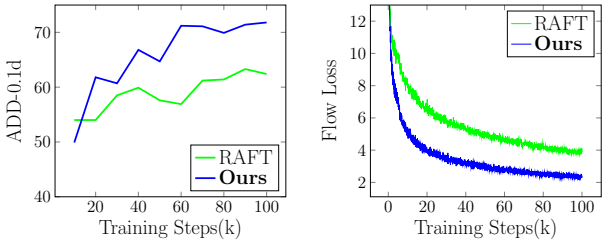


Figure 7. **Comparison during training on YCB-V.** We compare our method with the baseline RAFT in the validation accuracy and the flow loss w.r.t the number of training steps. After only 20k training steps, our method achieves similar performance as RAFT trained with 100k steps.

Dataset	LM 0.05d	LM 0.1d	LMO 0.05d	LMO 0.1d	YCB-V 0.05d	YCB-V 0.1d
PBR	65.5	95.0	28.6	57.0	8.8	28.9
	72.5	**96.8**	**35.6**	**63.7**	**10.9**	**36.5**
PBR+20	77.3	97.7	37.2	64.5	24.0	49.6
	81.5	**98.5**	**46.4**	**71.1**	**33.2**	**61.9**
Real	89.5	99.2	39.6	63.4	33.5	61.9
	92.9	**99.3**	**44.6**	**71.1**	**50.4**	**70.5**
Mixed	77.3	97.7	41.2	66.8	38.0	61.6
	90.9	**99.3**	**50.1**	**73.1**	**51.2**	**73.2**

Table 6. **Refinement comparison with different training data.** We compare our method with the baseline RAFT [47] in four different data settings. In each setting, the **first row** is from the baseline RAFT and the **second row** is from ours. Our method outperforms RAFT in all settings.

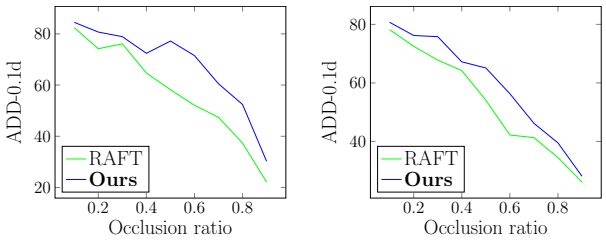
only 20 real images for every object ("PBR+20"), only real images, and also mixed PBR images and all the real images. Table 6 summarizes the results. For LINEMOD

(a) Target (b) Initialization (c) RAFT (d) Ours

Figure 8. **Qualitative results on YCB-V.** We compare our method with the baseline RAFT [47], and show the warped image from the predicted flow and the final pose results. Our method is much better than RAFT on this dataset.

and Occluded-LINEMOD, the performance of the model trained on PBR images with only 20 real images almost catches up with the one trained on real images. However, they still have a large gap on YCB-V. We believe this is caused by the different number of real images in different datasets. LINEMOD and Occluded-Linemod only have 2.4k real images for training, but YCB-V has 130k real ones, indicating that large-scale target domain data is still very beneficial to the target domain performance. Nevertheless, our method consistently outperforms the baseline RAFT in all different settings.

Evaluation against occlusion. We further compare our method with the baseline RAFT in occlusion scenarios. As shown in Fig. 9, our method consistently outperforms the baseline under different occlusion levels, both in LM-O and YCB-V. While we don't explicitly handle the occlusion, we attribute this to the utilization of 3D shape prior. With 3D shape prior, the pose estimator can infer the global pose from only the residual flow on the object's visible parts.



(a) YCB-V (b) LM-O

Figure 9. **Comparison w.r.t different occlusion ratios.**

5. Conclusion

We have introduced a shape-constraint recurrent flow framework for 6D object pose estimation, which is built on top of the intuition that the dense flow network should comply with the 3D shape of the target. We have first analyzed the weaknesses of common 2D flow networks in 6D object pose estimation and introduced a new recurrent matching framework embedding the 3D shape information of the tar-

get. We have demonstrated the effectiveness of our method on three challenging 6D object pose datasets. It produces much better pose results than the general matching methods, and achieves state-of-the-art pose results in both accuracy and efficiency.

References

[1] M.J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *International Conference on Computer Vision*, 1993. 1, 2

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, 2012. 1

[3] Ming Cai and Ian Reid. Reconstruct locally, localize globally: A model free method for object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2020. 2

[4] Bo Chen, Alvaro Parra, Jiewei Cao, Nan Li, and Tat-Jun Chin. End-to-end learnable geometric vision by backpropagating pnp optimization. In *Conference on Computer Vision and Pattern Recognition*, 2020. 1

[5] Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2

[6] Qifeng Chen and Vladlen Koltun. Full flow: Optical flow estimation by global optimization over regular grids. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[7] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014. 3

[8] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *The international journal of robotics research*, 30(10):1284–1306, 2011. 1

[9] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *arXiv preprint arXiv:1911.01911*, 2019. 1, 5

[10] Yan Di, Fabian Manhardt, Gu Wang, Xiangyang Ji, Nassir Navab, and Federico Tombari. So-pose: Exploiting self-occlusion for direct 6d pose estimation. In *International Conference on Computer Vision*, 2021. 2, 5

[11] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision*, 2015. 2, 5

[12] Rasmus Laurvig Haugaard and Anders Glent Buch. Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In *Conference on Computer Vision and Pattern Recognition*, 2022. 6

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[14] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*, 2012. 2, 5

[15] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas. BOP challenge 2020 on 6D object localization. *European Conference on Computer Vision Workshops*, 2020. 5, 6

[16] Markus Hofinger, Samuel Rota Bulò, Lorenzo Porzi, Arno Knapitsch, Thomas Pock, and Peter Kontschieder. Improving optical flow on a pyramid level. In *European Conference on Computer Vision*, 2020. 4

[17] Yinlin Hu, Pascal Fua, and Mathieu Salzmann. Perspective flow aggregation for data-limited 6d object pose estimation. In *European Conference on Computer Vision*, 2022. 1, 2, 3, 4, 5, 6, 7

[18] Yinlin Hu, Pascal Fua, Wei Wang, and Mathieu Salzmann. Single-stage 6d object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2

[19] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2019. 2

[20] Yinlin Hu, Yunsong Li, and Rui Song. Robust interpolation of correspondences for large displacement optical flow. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[21] Yinlin Hu, Rui Song, and Yunsong Li. Efficient coarse-to-fine patchmatch for large displacement optical flow. In *Conference on Computer Vision and Pattern Recognition*, 2016. 2

[22] Yinlin Hu, Sebastien Speierer, Wenzel Jakob, Pascal Fua, and Mathieu Salzmann. Wide-depth-range 6d object pose estimation in space. In *Conference on Computer Vision and Pattern Recognition*, 2021. 2, 5, 6

[23] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[24] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M. Kitani. Repose: Fast 6d object pose refinement via deep texture rendering. In *International Conference on Computer Vision*, 2021. 2, 5

[25] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *International Conference on Computer Vision*, 2021. 2

[26] Alexander Krull, Eric Brachmann, Frank Michel, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Learning analysis-by-synthesis for 6d pose estimation in rgb-d images. In *International Conference on Computer Vision*, 2015. 2, 5

[27] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, 2020. 2, 4, 6

[28] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009. 1, 2, 7

[29] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *European Conference on Computer Vision*, 2018. 2, 4, 5, 6

[30] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *International Conference on Computer Vision*, 2019. 2

[31] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching. In *International Conference on 3D Vision*, 2021. 2

[32] Lahav Lipson, Zachary Teed, Ankit Goyal, and Jia Deng. Coupled iterative refinement for 6d multi-object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 5, 6

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018. 5

[34] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *European Conference on Computer Vision*, 2018. 2

[35] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2015. 1

[36] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Conference on Computer Vision and Pattern Recognition*, 2016. 1, 5

[37] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2019. 2, 5

[38] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *International Conference on Computer Vision*, 2017. 2

[39] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020. 1, 5

[40] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1164–1172, 2015. 2

[41] Petter Risholm, Peter Ørnulf Ivarsen, Karl Henrik Haugholt, and Ahmed Mohammed. Underwater marker-based pose-estimation with associated uncertainty. In *International Conference on Computer Vision Workshops*, 2021. 2

[42] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, page 1100612. International Society for Optics and Photonics, 2019. 5

[43] Yongzhi Su, Mahdi Saleh, Torben Fetzer, Jason Rambach, Nassir Navab, Benjamin Busam, Didier Stricker, and Federico Tombari. Zebrapose: Coarse to fine surface encoding for 6dof object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2, 5

[44] Xiuchao Sui, Shaohua Li, Xue Geng, Yan Wu, Xinxing Xu, Yong Liu, Rick Siow Mong Goh, and Hongyuan Zhu. Craft: Cross-attentional flow transformers for robust optical flow. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2

[45] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2

[46] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *European Conference on Computer Vision*, 2018. 2

[47] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, 2020. 1, 2, 3, 4, 7, 8

[48] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Conference on Computer Vision and Pattern Recognition*, 2019. 2

[49] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Conference on Computer Vision and Pattern Recognition*, 2021. 2, 4, 5

[50] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Robotics: Science and Systems Conference*, 2018. 2, 5, 6

[51] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[52] Xu Yan, Lin Junyi, Zhang Guofeng, Wang Xiaogang, and Li Hongsheng. Rnnpose: Recurrent 6-dof object pose refinement with robust correspondence field estimation and pose optimization. In *Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 4, 5, 6

[53] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *International Conference on Computer Vision*, 2019. 2

[54] Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. On the continuity of rotation representations in neural networks. In *Conference on Computer Vision and Pattern Recognition*, 2019. 4