Shape Enhanced Keypoints Learning with Geometric Prior for 6D Object Pose Tracking

Mateusz Majcher Bogdan Kwolek✉
AGH University of Science and Technology, 30 Mickiewicza, 30-059 Kraków, Poland
{majcher,bkw}@agh.edu.pl

Abstract

Until now, there has not been much research in exploiting geometric reasoning on object shape and keypoints in object pose estimation. First, the current RGB image and quaternion representing rotation in the previous frame are fed to a multi-branch neural network responsible for regressing sparse object keypoints. The initial object pose is estimated using PnP, which is adjusted in a least-square optimization. The weights of boundary and keypoints components are determined in each iteration via geometric reasoning on the projected and segmented 3D object boundary, object shape extracted by a pretrained neural network and keypoints extracted by our network. Different from previous methods, our voting scheme is object boundary-based. We demonstrate experimentally that the accuracy of pose estimation is competitive in comparison to the accuracy of SOTA algorithms achieved on challenging YCB-Video dataset.

1. Introduction

Recent studies [13] show that human observers apply the same inferential rule from all viewpoints, leveraging the geometrically derived back-transform from retinal images to actual 3D scenes. Both in picture perception and 3D scene understanding the observers mentally apply projective geometry to retinal images. In computer vision, geometry describes the structure and shape of the world. One of the aims of geometric perception is to infer a relationship between the world and a robot [3]. Such a relationship is commonly modeled by 6DoF (degrees-of-freedom) pose, i.e. position and orientation. Estimation and tracking of object pose in real scenes is crucial for machine vision systems to enable robots' interaction with real environments and objects [8].

The sensor data from which the pose of an object is estimated can be either a single RGB image, a stereo image pair/depth map, or an image sequence. Estimation of 6D object pose from RGB images is not easy due to the intrinsic ambiguity caused by objects' visual appearance under different viewpoints and occlusion [8]. The object symmetry might induce visual ambiguities resulting in multiple visual appearances for the same pose, i.e. multiple pose estimates for the same visual appearance. The object pose estimation is an inherently 3D problem, where one of the main difficulties is recovering object scale from single RGB images. The essential difficulties stem from the fact that estimating 3D attributes of the objects from 2D measurements is an ill-posed problem.

Deep learning-based models have surpassed classical machine learning-based approaches in various computer vision tasks [2], including object pose estimation. The most straightforward way of recovering the object pose can be achieved by estimating the pose parameters directly [12,15,18,25,31]. Recent work demonstrates that pose estimation approaches, which combine deep learning with geometric optimization (P*n*P) are capable of achieving superior results [10, 17]. Such methods first learn a model to predict the 2D landmarks or fiducial points from the input image, then perform pose estimation by executing a P*n*P algorithm on the 2D-3D correspondences.

Object pose tracking approaches leverage information from the previous frame to enhance recovering object pose [8, 22]. While a large number of studies have been conducted in the field of 6DoF pose estimation, there is a comparatively small number of studies concerning 6D object pose tracking [8], in particular from RGB image sequences. In this work, we present an approach for object pose tracking, where a current RGB image and quaternion representing pose in the previous frame are fed to a multi-branch neural network, see Fig. 1, which delivers blobs representing object fiducial keypoints. Given a current pose guess, a 3D object boundary is segmented and then projected onto the 2D image plane. Object shape that is determined by a pretrained shape network is matched with the segmented object shape and then used to vote for confidences (weights) of the 3D keypoints. The final pose is determined via optimization-based pose refinement. The initial pose for the optimization algorithm is estimated on the basis of the object keypoints and a P*n*P algorithm. The

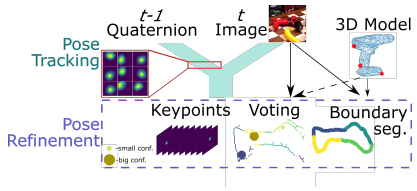

Figure 1. Shape enhanced keypoints learning with geometric prior.

value of the objective function is determined via matching the coarse 3D geometric model with keypoints estimated by our neural network and projected 3D shape with the object shape.

Our contribution. The main shortcomings of the current two-stage methods are due to that networks predict landmarks independently and consistency is only imposed by the pose solver, which is not part of the landmark regressor. To mitigate this effect, many top performing two-stage approaches are either based on pixel-vise voting [20] or on generating an ensemble of predictions from each image pixel or patch [17], and then aggregating them to improve final predictions. However, regressing landmarks is done independently without coherence with the object shape. In order to address the shortcomings mentioned we propose an approach for geometric reasoning on keypoints and object shape. The initial object pose is determined by a P*n*P, whereas the final pose is determined by the Levenberg-Marquardt algorithm. The pose estimation model at this stage utilizes object's distinct topological information, i.e. sparse 3D keypoints that are projected onto 2D image plane, distance transform representing object shape, and shape-based voting for keypoint confidences. The object rotation determined at this stage is then fed in the next frame to one of the inputs of the neural network responsible for the regression of object keypoints.

2. Relevant Work

Top-performing methods on existing benchmarks, rather than directly regressing the object pose, are based on two-stage approaches [12, 16, 17, 20, 21, 24, 25, 31, 33], which first predict landmarks of the object (intermediate features) with established 2D-3D correspondences, and then utilize a P*n*P like algorithm to determine the pose. Direct regression of the keypoints is performed in [11, 21, 25], while [12, 17] employ heatmaps to represent the keypoints. Recent experimental results [10, 17] indicate that two-stage methods are generally superior in comparison to methods directly estimating object pose. This two-stage mechanism is promising because learning intermediate features permits more flexibility for potential improvements during both the feature discovering and pose refinement with the help of scene geometry. A method proposed in [16] first utilizes a CNN to regress stable 3D object properties and then combines these estimates with geometric constraints provided by a 2D object bounding box. In [17], image patches are utilized to learn a landmark predictor. It has been assumed that at least some patches are not occluded and thus they could deliver more accurate landmark heatmaps. The final landmarks are determined via an ensemble of heatmaps representing object patches. To better cope with occluded objects, [20] proposed a neural network for pixel-wise voting for the 2D keypoints location. The distribution of the keypoints is determined using a generalized Hough voting scheme [1].

There are other notable approaches tackling object pose estimation on the basis of intermediate features. Segmentation-driven method [11] first predicts landmark locations for each small patch of the input image and then aggregates all patch predictions to establish 2D-3D correspondences needed for solving the pose. In order to utilize the advantages of end-to-end methods, Chen et al. [5] developed a differentiable P*n*P. In [28], color cue with the edge cue have been utilized in order to diminish the domain gap between synthetic images for training and real ones for inference. Edge cues have previously been used to identify object boundaries for 3D object detection and pose estimation [19, 23]. Edge features or object boundary features are rarely used in recent CNN-based approaches to object pose recovery.

Keypoint-based representations are commonly used to encode 3D geometric structure in objects. While [21, 25] utilize bounding box corners as keypoints, more recent approaches [20] use designated surface keypoints. The reason is that such bounding box corners are virtual landmarks, which can be far away from the object's shape.

Tracking of object pose is achieved with the help of information from the previous frame [8]. A recently proposed PoseRBPF (Rao–Blackwellized particle filter) [6] permits estimating the 3-D translation of an object and its full distribution over the 3-D rotation. This distribution captures the uncertainty in the object's pose, and owing to tracking the algorithm can better disambiguate the pose of the object. This means that after temporal occlusion the algorithm can recover the pose by tracking it from previous frames. In a recently proposed DeepIM [14] an optical flow is exploited to adjust initial pose estimates through minimization of differences between the 2D re-projection of the 3D model and the object appearance. While learning a deep neural network gradually learns to match the pose of the object via re-rendering the target such that the two input images that are fed to the network become more and more similar. Because of using a pretrained FlowNet [7], which takes two images and generates optical flow between them the DeepIM has substantial computational requirements. Moreover, DeepIM requires a large number of real-world images for training it.

3. Method

Our network is a two-branch architecture that takes in time t:

- RGB image as a first input
- quaternion representing object rotation in time $t-1$ as the second input

and generates k-channelled heatmaps with object keypoints. Each channel is a gray mask with active pixels forming heatmaps, whose centers represent keypoint locations. Each keypoint is regressed on a channel assigned to it in order to establish 2D-3D correspondences. The keypoints are then fed to a PnP that determines the initial object pose. It is utilized as an initial pose guess for the Levenberg-Marquardt (LM), which iteratively refines the object pose, see Fig. 2.

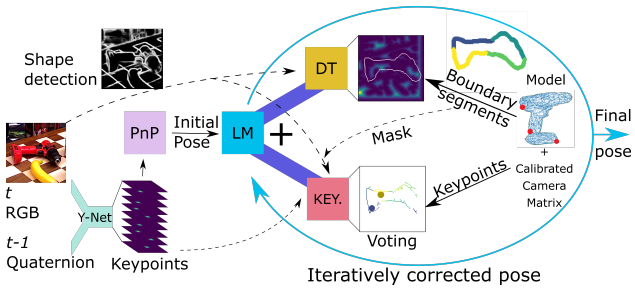

Figure 2. Visualization of data flow in the proposed approach to 6D object pose tracking on sequences of RGB images.

The objective function of LM contains two components: object shape component and keypoint component. Given the actual pose guess, the 3D object shape is determined and then split into k segments. The 3D keypoints and 3D shape segments are then projected onto 2D image plane using the camera matrix. The object shape is calculated using a pretrained neural network, which operates on RGB images. The values of distance transform (DT) on projected boundary segments are used to determine the weights for the shape component. The labeled shape segments are also used to calculate the weights of the keypoints. The shape segments are thickened and then used to mask object shape pixels. Each non-zero shape pixel votes for the nearest keypoint. The weights are used to weight matching results between the projected 3D points and keypoint heatmaps. In this way, we can potentially determine the occluded parts of the object and minimize the effects of occlusions.

In contrast to [20] which predicts dense vectors pointing to 2D keypoints, our algorithm votes for keypoint confidences on the basis of the (segmented) object shape. Moreover, our algorithm utilizes also a distance transform. There are two advantages of our method: i) it enables locating invisible or occluded keypoint(s) from segments of the object shape, ii) it iteratively calculates votes for the keypoints and shape weights, and then uses them in the objective function.

Figure 3 details the calculation of the objective function and pose refinement with the help of geometry. Given the current pose guess, the 3D object model is transformed to determine 3D object boundary. Afterwards, the 3D object boundary is split into k segments, which are projected onto 2D image plane, where each shape segment has assigned k-th label, see upper row on Fig. 3. Next, the object shape is calculated using a pretrained neural network. The labeled shape segments are used to mask the object shape pixels. Each non-zero shape pixel votes for the nearest keypoint. This means that occluded boundary pixels, which usually have zero or small values are masked with segmented boundary of the 3D model in the current pose. The confidences of keypoints are calculated on the basis of such votes. Thus, occluded parts of the object shape contribute less to the confidence of the nearest object keypoints. The confidences are used to calculate the weights of the projected 3D keypoints. The values of the heatmaps at the locations of the projected 3D keypoints are multiplied by weights (confidences) of the keypoints, see bottom row on Fig. 3. Finally, the values of DT on projected boundary segments are used to determine the weights of k boundary fragments. As values of distance transform on occluded objects parts are usually high, the corresponding labeled segments of the shape can be identified easily. In the current approach, the object segments with distance transform higher than a predefined threshold are not included in the calculation of the shape component in the objective function. The remaining shape segments are weighted such segments with higher average values of distance transform assume smaller weights, see middle row on Fig. 3.

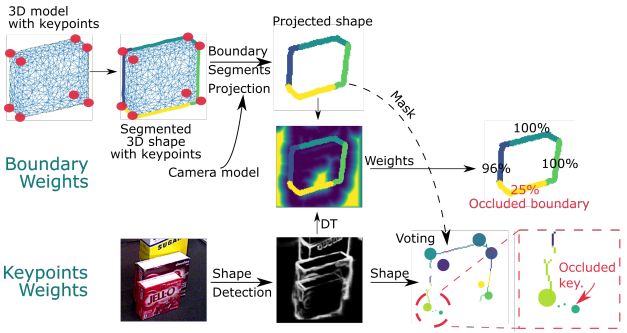

Figure 3. Calculation of objective function for pose refinement.

The objective function assumes the following form:

$$L = \min_{p}((1-\gamma)L_{key}(p) + \gamma L_{Bound.}(p)) \quad (1)$$

$$L_{key}(p) = \sum_{n=1}^{9} Voting_n^p * ||\mathbf{y}_{Pose}^n - \mathbf{y}_{Net}^n||_2^2 \quad (2)$$

$$L_{Bound.}(p) = \frac{1}{|E|} \sum_{\mathbf{e} \in E} Edge_e^p * D(\mathbf{e})^2 \quad (3)$$

where $\mathbf{y}^n_{Pose}$ is the projected keypoint of the 3D model in the actual pose p, rescaled to a reference plane, $\mathbf{y}^n_{Net}$ is 2D position of the corresponding n-th keypoint that is detected by our network and then rescaled to the reference plane, whereas E is a set of pixels representing the boundary of the object in the actual pose and rescaled to the reference plane, $\mathbf{e}$ is boundary segment, $D(\mathbf{e})$ is a function which returns the value of the distance transform for pixels in $\mathbf{e}$, and γ is a factor that was determined experimentally. $Voting^p_n$ is a weight of n-th keypoint, which is calculated on the basis of the voting, whereas $Edge^p_e$ stands for the weight of boundary segment e. $\mathbf{y}^n_{Pose} = \mathbf{K}(\mathbf{q}[0\ \mathbf{y}^n_{Model}]\mathbf{q}^{-1} + \mathbf{z})$, where $\mathbf{K}$ is matrix of camera's intrinsic parameters, $\mathbf{y}^n_{Model}$ is a keypoint from the 3D model, $\mathbf{q}$ is quaternion, and $\mathbf{z}$ is the translation. $Edge^p_e$ is calculated analogously. The initial pose was used to resize the object to the reference frame.

Figure 4 depicts architecture of our network. It operates on RGB images of size 128×128. The location and dimensions of object sub-window are determined on the basis of gravity center and height/width of 3D object boundary in the previous frame. The object sub-window is cropped using the camera model and the distance between the object and the camera in the previous frame.

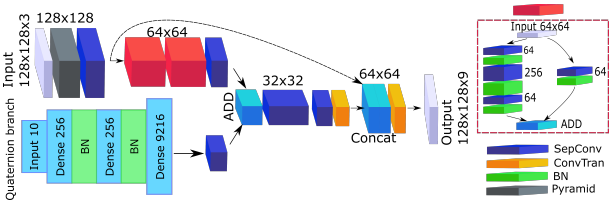

Figure 4. Architecture of neural network for object pose tracking.

4. Experiments

4.1. Dataset

The YCB video dataset [29] is one of the largest and most challenging datasets for benchmarking 6D object pose estimation. This dataset is also suited well for object pose tracking, c.f. [6, 14]. It contains 21 everyday life objects. It consists of 92 video sequences with a total of 133 827 frames. For each object, a 3D model is provided in the point cloud format. According to recommended train/test split, 80 video sequences are for training, whereas 2 949 keyframes chosen from the remaining 12 sequences are for testing. Additional 80 000 frames of synthetic data are provided in the training set.

4.2. Evaluation Metrics

The error of 6D object pose estimation was determined on the basis of Average Distance Metric (ADD) [9]. Given the object model $\mathcal{M}$ and its points $\boldsymbol{x}$, the ADD metric expresses the average distance of all model points' estimated pose $\hat{\mathbf{P}}\boldsymbol{x}$ to their ground truth pose $\mathbf{P}\boldsymbol{x}$:

$$e_{ADD}(\hat{\mathbf{P}}, \mathbf{P}, \mathcal{M}) = \underset{\boldsymbol{x}\in\mathcal{M}}{\text{avg}} ||\mathbf{P}\boldsymbol{x} - \hat{\mathbf{P}}\boldsymbol{x}||_2 \quad (4)$$

The e_{ADD} is evaluated with respect to the model diameter $d_{\mathcal{M}}$. The estimated pose is assumed to be correct if $e_{ADD} \leq \theta d_{\mathcal{M}}$, where θ is usually set to 0.1. The ADD-S metric [9], in which the distance to the closest ground truth point is measured:

$$e_{ADD-S}(\hat{\mathbf{P}}, \mathbf{P}, \mathcal{M}) = \underset{\boldsymbol{x_1}\in\mathcal{M}}{\text{avg}} \underset{\boldsymbol{x_2}\in\mathcal{M}}{\text{min}} ||\mathbf{P}\boldsymbol{x_1} - \hat{\mathbf{P}}\boldsymbol{x_2}||_2 \quad (5)$$

can handle symmetric objects. The AUC metric [29] is defined by the area under the accuracy-threshold curve when using the ADD/ADD-S metric. The curve is built by varying the threshold, to a maximum threshold of 10 cm. Routinely, the AUC for ADD/ADD-S is calculated with a distance threshold. For YCB-Video the 6D pose estimates are approved if ADD/ADD-S are smaller than 0.1 m threshold.

4.3. Experimental Results

Table 1 presents reprojection errors in terms of 5 px scores [4] that were obtained on the YCB-Video dataset for keypoints estimated by our network. The 5 px errors express how close the 2D projected vertices are to the ground-truth. As we can observe, the use of geometric prior in the form of object rotation from the previous frame permits achieving far better results in comparison to results achieved by an ordinary network for keypoints regression. For all YCB objects except two objects the 5 px error scores achieved by our network are far better in comparison to error scores achieved by algorithms discussed in [17] and [11].

Table 1. Reprojection errors, 5px scores [%] achieved by our network and recent algorithms on the YCB-Video dataset.

Object	[17]	[11]	Our Net w/o quat	Our Net w/ quat
002_master_chef_can	29.7	21.0	79.4	**82.4**
003_cracker_box	64.7	12	79.1	**86.9**
004_sugar_box	72.2	56.3	73.0	**77.1**
005_tomato_soup_can	39.8	46.2	54.9	**73.8**
006_mustard_bottle	**87.7**	70.3	75.7	82.3
007_tuna_fish_can	38.9	39.3	67.0	**72.2**
008_pudding_box	78.0	17.3	68.2	**84.8**
009_gelatin_box	**94.8**	83.6	0.1	87.6
010_potted_meat_can	41.2	60.7	87.7	**90.8**
011_banana	10.3	22.4	59.5	**63.0**
019_pitcher_base	5.43	33.5	74.2	**78.8**
021_bleach_cleanser	23.2	43.3	60.9	**66.8**
024_bowl*	26.1	13.3	38.2	**62.7**
025_mug	29.2	38.1	51.1	**66.7**
035_power_drill	69.5	43.3	64.1	**73.0**
036_wood_block*	2.06	2.5	34.9	**51.2**
037_scissors	12.1	8.8	36.4	**45.7**
040_large_marker	1.85	13.6	72.3	**83.0**
051_large_clamp*	24.2	7.6	45.9	**51.6**
052_extra_large_clamp*	1.32	0.6	55.8	**73.1**
061_foam_brick*	75.0	13.5	65.6	**78.2**
Avg.	39.4	30.8	59.2	**72.9**

Table 2 presents ADD and ADD-S that have been achieved on the YCB-Video dataset by P*n*P using keypoints determined by our network and our algorithm for 6D object pose tracking. In the P*n*P version of the algorithm, the keypoints have been determined by our network using rotation estimates from the previous frame. As we can see, the results achieved by our algorithm are much better.

Table 2. ADD scores [%] achieved by our algorithm. ADD is calculated for non-symmetric objects, whereas ADD-S is determined for symmetric objects, and * stands for symmetric objects.

Object	PnP	Our
002_master_chef_can	82.6	**95.6**
003_cracker_box	77.1	**88.8**
004_sugar_box	77.6	**92.6**
005_tomato_soup_can	45.0	**52.7**
006_mustard_bottle	84.5	**94.0**
007_tuna_fish_can	62.7	**69.2**
008_pudding_box	73.7	**89.1**
009_gelatin_box	90.6	**95.6**
010_potted_meat_can	45.5	**57.5**
011_banana	56.9	**61.5**
019_pitcher_base	91.6	**98.6**
021_bleach_cleanser	78.2	**85.5**
024_bowl*	69.0	**78.3**
025_mug	63.3	**74.5**
035_power_drill	66.7	**82.3**
036_wood_block*	80.5	**85.5**
037_scissors	46.3	**62.0**
040_large_marker	43.1	**51.6**
051_large_clamp*	69.5	**89.8**
052_extra_large_clamp*	66.0	**74.0**
061_foam_brick*	54.3	**59.3**
Avg.	67.8	**78.0**

Figure 6 presents example qualitative results, which have been achieved by our algorithm on the YCB-Video dataset. These sample images depict that our algorithm can cope with scene clutter, severe occlusions, reflection, different illumination, and various movements of objects. As shown, the large errors can occur for symmetrical objects, see also bowl object (2nd row, 6th from left).

Figure 5 contains plots of ADD scores vs frame number for the power drill. These sample plots illustrate the ADD scores that have been achieved using voting in object pose refinement (upper plot), using boundary in object pose refinement (middle plot), and with both voting and boundary used in pose refinement (bottom plot). As we can observe, our proposed pose refinement method achieved the average ADD score equal to 73%, whereas the average ADD scores for voting-based and boundary-based algorithms have been equal to 70% and 71%, respectively.

4.4. Comparison with SOTA

Table 3 compares AUC ADD scores achieved by our algorithm with AUC ADD scores achieved by recent algorithms. As we can observe, our algorithm achieves competitive results on challenging YCB-Video dataset. It achieved

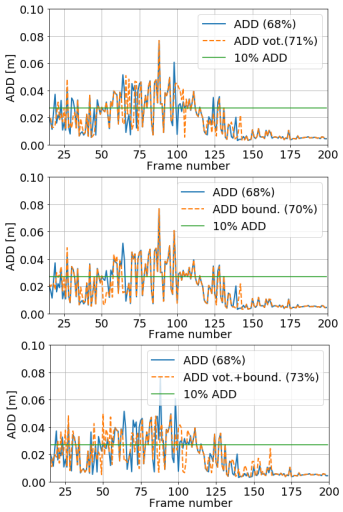

Figure 5. ADD scores vs time for power drill from YCB-Video dataset: using voting in object pose refinement (upper plot), using boundary in object pose refinement (middle plot), and with voting and boundary used in pose refinement (bottom plot).

better results in comparison to PoseRBPF [6] and similar average AUC ADD score with DeepIM [14], which similarly to our algorithm have been developed for tracking 6D object pose in sequences of RGB maps. For seven objects the AUC ADD scores achieved by our algorithm were better, whereas for one object the AUC ADD score achieved by DeepIM was better than the score achieved by our algorithm. For seven objects our algorithm achieved second best results. The recently proposed GDR-Net [27] achieved better AUC ADD scores for twelve objects. As we can notice, the average AUC ADD score for all objects is smaller in comparison to the average AUC ADD score achieved by our algorithm. For several objects, including pudding box and scissors the discussed results were considerably worse in comparison to results achieved by our algorithm. Most of the recent methods estimate the object pose on single RGB images without taking advantages of the temporal consistency among video frames. Temporal consistency is a very important cue, and we demonstrated that better results can be achieved using it in object tracking. Our pose refinement strategy permitted us to achieve competitive results.

4.5. Implementation Details

The system has been implemented in Python using Keras. The neural networks have trained in 500 epochs, batch size set to four, learning rate equal to 0.001, MSE loss function, using RMSprop. For $eps = 0.0001$ the average number of evaluations of the objective function by the LM is about eleven. The maximum number of iterations was set to twenty. The object boundary has been extracted using a pretrained HED network [30].

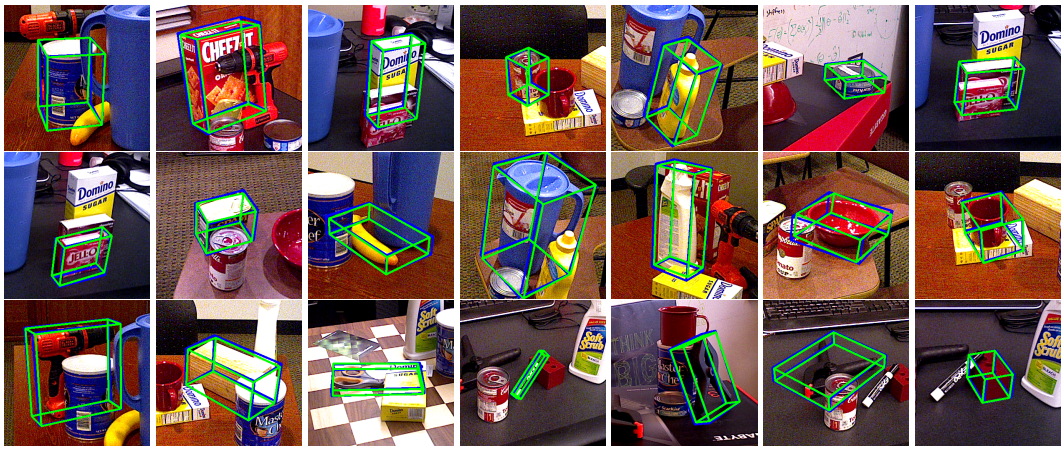

Figure 6. Qualitative results on YCB-Video. Example images demonstrate how our algorithm handles occlusions, different arrangement, lighting in ordinary shots for the following objects: 002_master_chef_can, 003_cracker_box, 004_sugar_box, 005_tomato_soup_can, 006_mustard_bottle, 007_tuna_fish_can, 008_pudding_box, 009_gelatin_box, 010_potted_meat_can, 011_banana, 019_pitcher_base, 021_bleach_cleanser, 024_bowl, 025_mug, 035_power_drill, 036_wood_block, 037_scissors, 040_large_marker, 051_large_clamp, 052_extra_large_clamp, and 061_foam_brick. The green bounding boxes show the ground truth poses, while the blue ones correspond to the estimated poses. For better visualization we cropped regions of interest.

Table 3. AUC ADD scores [%] (max. th. 10 cm) achieved by algorithms on the YCB-Video dataset. AUC ADD is calculated for non-symmetric objects, whereas AUC ADD-S [29] is determined for symmetric objects. "-" denotes unavailable results, '*' stands for symmetric objects, best results are marked in bold, second best results are underlined.

	Pose recovery on single RGB images					Pose tracking		
Object	PoseCNN [29]	DOPE [26]	[17]	[32]	GDR-Net [27]	PoseRBPF [6]	DeepIM [14]	Our
002_master_chef_can	50.9	-	81.6	49.9	65.2	63.3	<u>89.0</u>	**90.5**
003_cracker_box	51.7	55.9	83.6	80.5	<u>88.8</u>	77.8	88.5	**89.1**
004_sugar_box	68.6	75.7	82.0	85.5	**95.0**	79.6	<u>94.3</u>	90.0
005_tomato_soup_can	66.0	76.1	79.7	68.5	**91.9**	73.0	<u>89.1</u>	80.0
006_mustard_bottle	79.9	81.9	91.4	87.0	**92.8**	84.7	<u>92.0</u>	<u>92.0</u>
007_tuna_fish_can	70.4	-	49.2	79.3	**94.2**	64.2	<u>92.0</u>	89.2
008_pudding_box	62.9	-	**90.1**	81.8	44.7	64.5	80.1	<u>84.1</u>
009_gelatin_box	75.2	-	<u>93.6</u>	89.4	92.5	83.0	92.0	**94.6**
010_potted_meat_can	59.6	39.4	79.0	59.6	<u>80.2</u>	51.8	78.0	**85.0**
011_banana	72.3	-	51.9	36.5	**85.8**	18.4	<u>81.0</u>	77.2
019_pitcher_base	52.5	-	69.4	78.1	**98.5**	63.7	90.4	<u>92.3</u>
021_bleach_cleanser	50.5	-	76.1	56.7	**84.3**	60.5	81.7	<u>82.5</u>
024_bowl*	69.7	-	76.9	23.5	85.7	85.6	**90.6**	<u>89.2</u>
025_mug	57.7	-	53.7	54.0	**94.0**	77.9	<u>92.0</u>	85.9
035_power_drill	55.1	-	82.7	82.8	**90.1**	71.8	<u>85.4</u>	82.1
036_wood_block*	65.8	-	55.0	29.6	<u>82.5</u>	31.4	75.4	**84.1**
037_scissors	35.8	-	65.9	46.0	49.5	38.7	<u>70.3</u>	**70.5**
040_large_marker	58.0	-	56.4	9.8	76.1	67.1	<u>80.4</u>	**81.8**
051_large_clamp*	49.9	-	67.5	47.4	**89.3**	59.3	84.1	<u>86.9</u>
052_extra_large_clamp*	47.0	-	53.9	47.0	**93.5**	44.3	90.3	<u>90.8</u>
061_foam_brick*	87.8	-	89.0	87.8	**96.9**	92.6	<u>95.5</u>	93.9
Avg.	61.3	65.8	72.8	61.0	<u>84.4</u>	64.4	**86.3**	**86.3**

5. Conclusions

In this paper, we presented a novel algorithm for 6D object pose estimation on RGB images. We demonstrated experimentally that our network that takes the current RGB image on the first input and quaternion representing object rotation in the previous frame in the second input permits achieving better results in comparison to the network operating on RGB images only. We presented a pose refinement algorithm that on the basis of geometry between keypoints and objects shape permits improvement of accuracy in 6D object pose estimation.

Acknowledgment.

This work was supported by Polish National Science Center (NCN) under a research grant 2017/27/B/ST6/01743.

References

[1] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Rec.*, 13(2):111–122, 1981. 2

[2] Yoshua Bengio, Yann Lecun, and Geoffrey Hinton. Deep learning for AI. *Commun. ACM*, 64(7):5865, 2021. 1

[3] Julien Bidot, Lars Karlsson, Fabien Lagriffoul, and Alessandro Saffiotti. Geometric backtracking for combined task and motion planning in robotic systems. *Artificial Intelligence*, 247:229–265, 2017. Special Issue on AI and Robotics. 1

[4] E. Brachmann, F. Michel, A. Krull, M. Yang, S. Gumhold, and C. Rother. Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In *CVPR*, pages 3364–3372, 2016. 4

[5] B. Chen, A. Parra, J. Cao, N. Li, and T. Chin. End-to-end learnable geometric vision by backpropagating PnP optimization. In *CVPR*, pages 8097–8106, 2020. 2

[6] Xinke Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. PoseRBPF: A Rao-Blackwellized particle filter for 6D object pose estimation. In *Proc. of Robotics: Science and Systems XV*, 2019. 2, 4, 5, 6

[7] Alexey Dosovitskiy, Philipp Fischery, 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 *ICCV*, pages 2758–2766. IEEE Comp. Society, 2015. 2

[8] Zhaoxin Fan, Yazhi Zhu, Yulin He, Qi Sun, Hongyan Liu, and Jun He. Deep learning on monocular object pose detection and tracking: A comprehensive overview. *CoRR*, abs/2105.14291, 2021. 1, 2

[9] 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 *Proc. of the 11th Asian Conf. on Computer Vision - Vol. I*, pages 548–562. Springer, 2013. 4

[10] Yinlin Hu, Pascal Fua, Wei Wang, and Mathieu Salzmann. Single-stage 6D object pose estimation. In *CVPR*, pages 2927–2936, 2020. 1, 2

[11] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6D object pose estimation. In *CVPR*, pages 3385–3394, 2019. 2, 4

[12] 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 *ICCV*, pages 1530–1538, 2017. 1, 2

[13] Erin Koch, Famya Baig, and Qasim Zaidi. Picture perception reveals mental geometry of 3D scene inferences. *Proc. of the National Academy of Sciences*, 115(30):7807–7812, 2018. 1

[14] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. DeepIM: Deep iterative matching for 6D pose estimation. *Int. J. Comput. Vis.*, 128:657–678, 2020. 2, 4, 5, 6

[15] Zhigang Li, Gu Wang, and Xiangyang Ji. CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation. In *ICCV*, pages 7677–7686, 2019. 1

[16] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3D bounding box estimation using deep learning and geometry. In *CVPR*, pages 5632–5640, 2017. 2

[17] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In *ECCV*, pages 125–141, 2018. 1, 2, 4, 6

[18] Kiru Park, Timothy Patten, and Markus Vincze. Pix2Pose: Pixel-wise coordinate regression of objects for 6D pose estimation. In *ICCV*, pages 7667–7676, 2019. 1

[19] Nadia Payet and Sinisa Todorovic. From contours to 3D object detection and pose estimation. In *ICCV*, pages 983–990, 2011. 2

[20] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao. PVNet: Pixel-wise voting network for 6DoF pose estimation. In *CVPR*, pages 4556–4565, 2019. 2, 3

[21] 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 *ICCV*, pages 3848–3856, 2017. 2

[22] Caner Sahin, Guillermo Garcia-Hernando, Juil Sock, and Tae-Kyun Kim. A review on object pose recovery: From 3D bounding box detectors to full 6D pose estimators. *Image and Vision Computing*, 96:103898, 2020. 1

[23] Byung-Kuk Seo, Hanhoon Park, Jong-Il Park, Stefan Hinterstoisser, and Slobodan Ilic. Optimal local searching for fast and robust textureless 3D object tracking in highly cluttered backgrounds. *IEEE Trans. on Visualization and Computer Graphics*, 20(1):99–110, 2014. 2

[24] Chen Song, Jiaru Song, and Qixing Huang. HybridPose: 6D object pose estimation under hybrid representations. In *CVPR*, pages 428–437, 2020. 2

[25] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6D object pose prediction. In *CVPR*, pages 292–301. IEEE Comp. Society, 2018. 1, 2

[26] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *Proc. 2nd Conf. on Robot Learn.*, volume 87, pages 306–316, 2018. 6

[27] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation. In *CVPR*, June 2021. 5, 6

[28] Yilin Wen, Hao Pan, Lei Yang, and Wenping Wang. Edge enhanced implicit orientation learning with geometric prior for 6D pose estimation. *IEEE Robotics and Automation Letters*, 5(3):4931–4938, 2020. 2

[29] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. *Proc. of Robotics: Science and Systems (RSS)*, 2018. 4, 6

[30] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, pages 1395–1403, 2015. 6

[31] S. Zakharov, I. Shugurov, and S. Ilic. DPOD: 6D pose object detector and refiner. In *ICCV*, pages 1941–1950, 2019. 1, 2

[32] Moritz Zappel, Simon Bultmann, and Sven Behnke. 6D object pose estimation using keypoints and part affinity fields. *ArXiv, CoRR*, abs/2107.02057, 2021. 6

[33] S. Zhang, W. Zhao, Z. Guan, X. Peng, and J. Peng. Keypoint-graph-driven learning framework for object pose estimation. In *CVPR*, pages 1065–1073. IEEE Comp. Society, 2021. 2