Accurate 6D Object Pose Estimation and Refinement in Cluttered Scenes

Yixiang Jin[a], John Anthony Rossiter[b] and Sandor M. Veres[c]
Department of Automatic Control Systems and Engineering, University of Sheffield, U.K.

Keywords: 6D Pose Estimation, 3D Robotic Vision, 3D Object Detection.

Abstract: Estimating the 6D pose of objects is an essential part of a robot's ability to perceive their environment. This paper proposes a method for detecting a known object and estimating its 6D pose from a single RGB image. Unlike most of the state-of-the-art methods that deploy PnP algorithms for estimating 6D pose, the method here can output the 6D pose in one step. In order to obtain estimation accuracy that is comparable to RGB-D based methods, an efficient refinement algorithm, called contour alignment (CA), is presented; this can increase the predicted 6D pose accuracy significantly. We evaluate the new method in two widely used benchmarks, LINEMOD for single object pose estimation and Occlusion-LINEMOD for multiple objects pose estimation. The experiments show that the proposed method surpasses other state-of-the-art prediction approaches.

1 INTRODUCTION

Accurate 6D pose estimation of objects is important in many real-world applications of computer vision, including augmented reality, robot manipulation and advanced autopilot operations on aerial and ground vehicles. Currently the majority of accurate 6D pose estimation methods rely on RGB-D information (Brachmann et al., 2014; Brachmann et al., 2016; Michel et al., 2017; Xiang et al., 2017; Wang et al., 2019). However, the depth sensor exposes several practical limitations such as high power consumption, limited working range, and sensitivity to the environmental effects. Such impediments mean that accurate 6D detection is not normally deployed on monocular cameras and mobile devices. The goal of this paper is to present a precise 6D detection method that works from a single RGB image and relies on the use of deep neural networks.

Traditionally, the 6D pose estimation issue is addressed by pairing feature points between 2D images and to obtain the corresponding 3D object models (Lowe, 2004) from the resulting cloud point. However, such approaches have failed to address textureless targets. By contrast, the template-matching method (Hinterstoisser et al., 2011; Hinterstoisser et al., 2012) is more robust than feature-matching, but it leads to low pose detection accuracy in environments full of occluded objects. Although dense feature learning approaches (Kendall and Cipolla, 2017; Krull et al., 2017) present good performance in occlusions, they fail to resolve the case of symmetric objects.

The emergence of deep learning techniques, especially CNN-based category detectors, have shown excellent outcomes for object detection (Krull et al., 2017; Ren et al., 2015) and object segmentation(He et al., 2017). Recently, there is an increasing number of works (Kehl et al., 2017; Tekin et al., 2018; Hu et al., 2019; Hodan et al., 2020), which employ deep learning for 6D pose estimation. Most of these approaches follow a similar paradigm: first they use a neural network to detect the eight 3D bounding box vertices associated with the target objects, then they perform an Perspective-n-Point (PnP) (Lepetit et al., 2009) algorithm calculating the orientation and translation. However, this paradigm suffers from a severe shortcoming in terms of low detection accuracy. The reason is that the key points are often not on the surface of the object, so there is some inaccuracy in the detection. As the PnP algorithm continues to accumulate these errors, an Iterative Closest Point (ICP) processing is executed in several steps to refine the pose.

The goal of this paper is to resolve the above limitations by training a deep neural network that can accurately predict 6D pose from an RGB image in a sin-

[a] https://orcid.org/0000-0001-6286-278X
[b] https://orcid.org/0000-0002-1336-0633
[c] https://orcid.org/0000-0003-0325-0710

Jin, Y., Rossiter, J. and Veres, S.
Accurate 6D Object Pose Estimation and Refinement in Cluttered Scenes.
DOI: 10.5220/0010654500003061
In *Proceedings of the 2nd International Conference on Robotics, Computer Vision and Intelligent Systems (ROBOVIS 2021)*, pages 31-39
ISBN: 978-989-758-537-1

gle step. Compared with previous works, our method can estimate the object pose directly without a PnP iterative process. In addition, we design a contour alignment (CA) refinement algorithm to replace the ICP processing that requires depth information. Due to the use of CA, our system only needs RGB information to run.

In this paper, we propose a two-stage convolution neural network inspired by Mask RCNN (He et al., 2017). This network takes a single RGB image as an input and can output the object class, 2D bounding box, the object mask and object rotation simultaneously. Following these, the lateral position of the object is calculated by a reverse projection algorithm. In order to obtain estimation accuracy comparable to RGB-D, we propose an efficient algorithm to align the object 2D projection and the object mask contour.

We evaluate our approach on the LINEMOD dataset(Hinterstoisser et al., 2012) (a single object 6D pose estimation dataset) and on the Occluded-LINEMOD dataset (Brachmann et al., 2014) (a multiple objects dataset). Additionally, we compare our result with some recent work. Furthermore, to completely evaluate our algorithm, we perform some tests on objects in the real world.

In summary, the main contributions of this paper are:

- We propose a novel 6D pose estimation method which can detect objects, segment instances and predict 6D pose simultaneously without any PnP process.
- We introduce Contour-Alignment, an efficient algorithm for pose refinement in an RGB image.

This paper consists of five sections. After describing the related prior work, the paper introduces the new methodology. This is then followed by presentation of a range of experiments and finally conclusions are drawn.

2 RELATED WORK

In this section, we review published 6D pose estimation methods, ranging from traditional feature and template matching approaches to state-of-art CNN-based methods.

Early object 6D pose estimation approaches mainly used feature matching (Lowe, 2004) and template matching (Hinterstoisser et al., 2011; Hinterstoisser et al., 2012). These works were primarily applicable to objects with rich texture However, many objects are texture free in the real world and industry. Consequently, these traditional approaches often fail in severely occluded and cluttered environments.

In recent years, there are an increasing number of 6D pose estimation works which involve the use of CNNs. CNN-based approaches can be classified in terms of input data into RGB based methods (Kehl et al., 2017; Tekin et al., 2018; Hu et al., 2019; Rad and Lepetit, 2017) and RGB-D based methods (Michel et al., 2017; Xiang et al., 2017; Wang et al., 2019). As for the RGB-D inputs, a common strategy is to establish correspondences between 3D scene points and 3D model points (Park et al., 2019), and then estimate the 6D pose of the object by solving a least-squares problem. Some authors (Brachmann et al., 2014) proposed a system to predict dense object coordinates that can compute object pose from dense correspondences, while others (Wang et al., 2019) embed and fused RGB pixel and point clouds at a per-pixel level as training data.

Methods for the RGB image pose detection can be divided into two groups. Methods in the first group detect 3D bounding box vertices for objects and then compute 6D pose by solving the PnP problem (Kehl et al., 2017; Tekin et al., 2018; Hu et al., 2020). This is currently the most popular computational paradigm. The second type of RGB-based pose estimation treats 6D pose estimation as a regression issue (Do et al., 2018). However, the performance of these approaches is not comparable to RGB-D based works owing to the lack of an effective pose refinement procedure using RGB images only. Additionally, to make estimation more precise, some researchers focus on refinement methods for pose correction after the initial calculation. For example, Deep-IM (Li et al., 2018) proposes an iterative matching network and Fabian et al. (Manhardt et al., 2018) introduce ”visual loss” to improve the initial pose.

In this paper we propose an end-to-end network, which can not only detect and segment but also estimate the 6D pose from an RGB image. We also introduce a novel refinement technique, called contour-alignment, which is applied as post-processing in the presented RBG based 6D pose estimation method.

3 METHODOLOGY

In this section we will introduce our novel 6D pose estimation algorithm and refinement approach. We first describe our network architecture, then we present our method to estimate object pose. After that, we detail our CA refinement algorithm before finally introducing the set up for training and inference.

3.1 Network Architecture

We propose an architecture inspired by Mask-RCNN and goes beyond Mask-RCNN in capability. Our network contains two stages: i) it starts with the ResNet101 (He et al., 2016) backbone that extracts features over the entire image and then ii) the Region of Interest (ROI) is extracted by a Region Proposal Network (RPN) that feeds its results to the head branches. In our system we have five parallel head branches as follows:

1. class regression branch
2. box regression branch
3. segmentation head branch
4. orientation head branch
5. corner head branch

The combined network with the Mask-RCNN can achieve classification, segmentation, and estimation of 6D pose of object instances simultaneously.

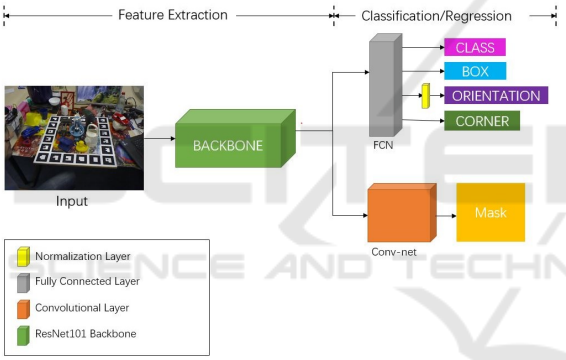


Figure 1: The neural network architecture. The shape of fully connected layers are two $4096 \times 1 \times 1$ layer and the size of convolutional layer is $7 \times 7 \times 512$.

Our architecture in Fig.1, uses quaternions to represent rotation, so there is a normalization layer in front of a rotation layer. We also use the fully-convolutional layer to predict a pixel-wise instance segmentation by up-sampling the feature map to 28×28.

3.2 Pose Estimation

The object pose usually includes a rotation matrix and a translation vector. The rotation matrix is estimated using quaternion regression from the neural network. As for the translation vector, instead of predicting it from neural networks directly, we have designed a fast and simple algorithm to calculate it. The reason why we deprecate regression of translation is that the neural network can't handle camera intrinsic matrix changes. It is impossible to train a network for each type of camera. So our network predicts object rotation and translations separately.

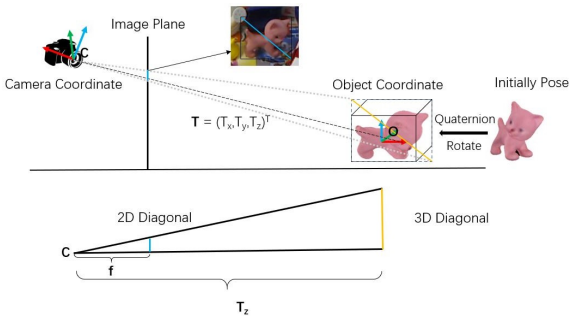


Figure 2: Illustrating the relationship between the object coordinate system and the camera coordinate system. The 3D translation is calculated by a projection principle.

As shown in Fig.2, the translation vector $T = [T_x, T_y, T_z]$ defines the coordinates of the object center in the camera coordinate system and the cat model is under the current orientation. The crucial step to estimate the 3D translation is calculating T_z. The camera projection is a 3D-to-2D perspective projection and we utilize the reverse projection principle to recover the depth T_z. As illustrated in Fig.2, a 2D diagonal (the blue line obtained from neural network) and a 3D diagonal (the yellow line calculated from 3D model) can be used to derive the T_z:

$$T_z = \frac{3Ddiagonal}{2Ddiagonal} * f \tag{1}$$

where f denote the focal lengths of the camera. We assume that the focal lengths in horizontal f_x and vertical f_y directions are equivalent. The same procedure can be easily adapted to obtain T_x and T_y:

$$\begin{bmatrix} T_x \\ T_y \end{bmatrix} = \begin{bmatrix} \frac{u - c_x}{f_x} * T_z \\ \frac{v - c_y}{f_y} * T_z \end{bmatrix} \tag{2}$$

where $[u, v]$ is the object center, which predicts from the neural network. $[c_x, c_y]$ expresses the principal point, which would be theoretically in the centre of the image.

3.3 Pose Refinement

Though the estimated object poses are already precise, they can still be improved by a further refinement. For the RBG-D data, the detection usually follows by ICP processing. In this paper, we propose an edge-based refinement algorithm by aligning the object instance contours and 2D projection contours.

We call it a contours alignment (CA) algorithm. This method can be adapted to any CNN-based 6D pose estimation framework to improve accuracy.

Algorithm 1: Position Refinement.

Input: Initialise object pose P_0; Object mask M_0 predicted by neural network; 3D-2D projection function f; Object model;

Output: Refine object pose P_n

1: Calculate contour C_0 for object mask M_0.
2: Set C_0 as reference points.
3: Compute 2D projection $proj$ with current pose P_0, $proj = f(P_0)$
4: Extract contour C_1 from $proj$.
5: Apply a closest point pairs algorithm between C_0 and C_1 to obtain C_3.
6: Compute residual error: $df = C_3 - C_1$.
7: Calculate Jacobian matrix J of f, so $df = Jdx$.
8: Solve dx using pseudo inverse $\mathrm{d}x = (J^T J)^{-1} J^T \mathrm{d}y$, and update pose P_0.
9: Repeat steps 3-8 until reach threshold; **return** P_0.

In Algorithm 1, we extract contours by using the "find_contours" function from the skimage module(Van der Walt et al., 2014), that is an image processing module in python. The "find_contours" function uses the "matching squares" and linearly interpolated approach to obtaining the iso-valued contours of the input 2D array for a specific value. The closest points pairing process employs a kd-tree search from the "sklearn.neighbors" module (Pedregosa et al., 2011). The closest point pairs guarantee that two contour arrays have the same shape so that we can perform arrays subtraction.

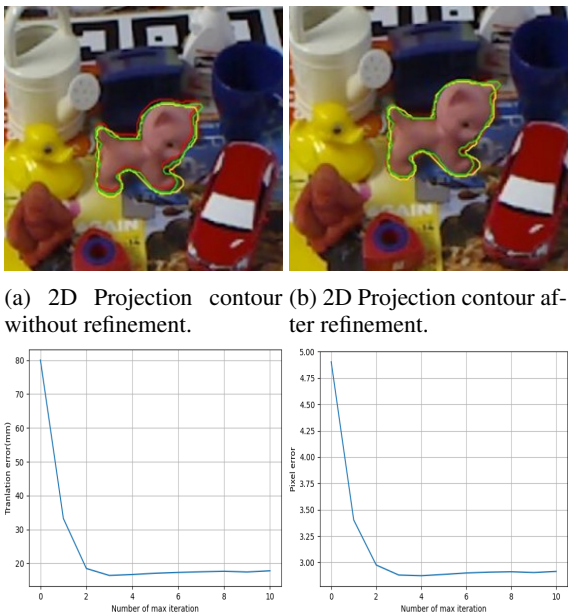
In order to achieve an appropriate balance between accuracy and efficiency, we only optimize the translation because the error in translation is more dominant than rotation. the Jacobian matrix J is:

$$J = [\frac{\partial f}{\partial T_x}, \frac{\partial f}{\partial T_y}, \frac{\partial f}{\partial T_y}] \tag{3}$$

we approximate the derivatives to obtain:

$$J \approx \begin{bmatrix} \frac{f(T+[\varepsilon,0,0]) - f(T)}{\varepsilon} \\ \frac{f(T+[0,\varepsilon,0]) - f(T)}{\varepsilon} \\ \frac{f(T+[0,0,\varepsilon]) - f(T)}{\varepsilon} \end{bmatrix}^T \tag{4}$$

where T denotes the translation vector and ε is a tiny number. In this paper we choose $\varepsilon = 0.0000001$ to guarantee the size of projection points is constant. Therefore, in Algorithm 1, the two contour arrays C_1 and C_3 can subtract.



(a) 2D Projection contour without refinement. (b) 2D Projection contour after refinement.

(c) Translation error under different iteration times. (d) Pixel error under different iteration times.

Figure 3: Improvement of our refinement algorithm for 6D pose estimation. In (a) and (b), green lines show the ground truth contour, yellow lines present the predicted mask contour and red lines indicate 2D projection using the current pose.

In Fig. 3 one can observe that the projection contour extracted by refinement of pose (red line) coincides with the ground truth contour (green line). This shows that our algorithm can improve pose accuracy significantly. Furthermore, we can see that in the first refinement iteration, both the translation and pixel errors are reduced by nearly 60%, and that this tends to converge after the second refinement iteration. Therefore, our algorithm can refine object pose quickly and effectively.

3.4 Training and Inference

We have implemented our system in Python3 using the TensorFlow library(Abadi et al., 2016). The input to the neural network was an RGB image with size 640×480. Our training data consisted of three parts: i) first is the RGB image; ii) second is a binary mask image and iii) the third part is a label. Unlike other approaches using eight corner annotations or 6D pose annotations, we adopt a new annotation method based on a quaternion and two corner points as shown in the Fig.4, because such an annotation can fit our pose estimation algorithm better.

In training, we define a multi-task loss to jointly train the classification, bounding box regression, instance segmentation, quaternion regression and cor-

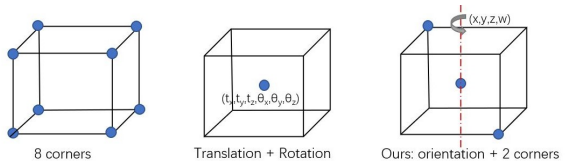


Figure 4: Comparing with different annotation method.

ner point regression. Formally, the total loss function is defined as follows:

$$L = \alpha_1 L_{cls} + \alpha_2 L_{box} + \alpha_3 L_{mask} + \alpha_4 L_{quat} + \alpha_5 L_{cor} \quad (5)$$

where α_1, α_2, α_3, α_4, α_5 are loss weights, which indicate the importance of each loss component. In our experiments, we set $\alpha_1 = \alpha_2 = \alpha_5 = 1$, $\alpha_3 = 10$ and $\alpha_4 = 2$. L_{cls} is softmax loss, L_{box} and L_{cor} are smooth L1 loss, L_{mask} is binary cross-entropy loss, and L_{quat} is a derivation of L2 Loss, defined as follows:

$$L_{quat} = \frac{\sum_{i=1}^{n} (\beta r_i - \beta \bar{r})^2}{n} \quad (6)$$

where r_i denotes the predicted quaternion and $\bar{r}$ is the ground true quaternion. The four parameters of the quaternion are all between 0 and 1, so we apply a magnification factor β ($\beta = 10$ in our experiments).

We train our network on a Tesla V100 GPU for 90 epochs. The first 20 epochs train network heads with a 0.002 learning rate. Then, using the same learning rate, we fine tune the layers from ResNet stage 4 in the next 10 epochs. After that, we train all layers for 30 epochs. In the following 10 epochs, the learning rate is decreased by 10 until we train all the layers. Lastly, we change the learning rate to 0.00002 to fine tune all the layers in the final 10 epochs.

At the inference phase, we select object instances which have their detection scores higher than 0.9. Our pose estimation algorithm and refinement method are then applied to the detected objects to obtain accurate 6D pose matrices.

4 EXPERIMENTS

We conduct our experiments on two standard data sets including a single object pose data set LINEMOD, and a multiple objects pose data set Occlusion-LINEMOD to evaluate our method for 6D pose estimation. We compare our work against some widely used state-of-the-art 6D pose estimation approaches. We also prove that our method can apply to real-world custom objects.

4.1 Evaluation Metrics

Our work has been evaluated under the average distance (ADD) metric (Hinterstoisser et al., 2012). The average distance calculates the mean of pairwise distances between 2D projections of the 3D models, calculated utilizing the estimated pose and ground truth pose:

$$ADD = \frac{1}{m} \sum_{x \in M} \min_{M} \|(\boldsymbol{R}\boldsymbol{x} + \boldsymbol{T}) - (\bar{\boldsymbol{R}}\boldsymbol{x} + \bar{\boldsymbol{T}})\| \quad (7)$$

where $\boldsymbol{R}$, $\boldsymbol{T}$, $\bar{\boldsymbol{R}}$, and $\bar{\boldsymbol{T}}$ are ground true rotation, ground true translation, estimated rotation and estimated translation, respectively. M denotes the vertex set of the 3D model, and m means the number of 3D points. evaluation is based on the widely used metric **ADD-0.1d** and **REP-5px**, where the estimated pose is considered to be correct if the average distance is below 10% of the object's diameter or smaller than a 5 pixels threshold.

4.2 Single Object Pose Estimation

We first test our method on the LINEMOD data set, which contains 15 objects with poor texture in a cluttered environment. In common with other papers in the literature, we evaluate methods on 13 of these objects. We adopt similar settings with (Tekin et al., 2018) to randomly select 30% of the images as training data and the rest of images as test data. Only RGB images are however used in the training and testing phase.

Table 1: Comparison of our method with state-of-the-art work on LINEMOD data set in terms of ADD-0.1 metric. We present percentages of correctly estimated pose and highlight the best result among those by **bold** numbers.

Object \ Method	Zhao	Yolo-6D	SSD-6D	Our
Ape	35.1	21.62	0	**42.29**
Benchvise	23.9	**81.8**	0.18	77.64
Cam	33.2	36.57	0.41	**66.78**
Can	21.0	68.80	1.35	**74.09**
Cat	30.6	41.82	0.51	**57.89**
Driller	28.6	63.51	2.58	**70.45**
Duck	27.9	27.23	0	**37.81**
Eggbox	38.9	**69.58**	8.9	64.5
Glue	31.2	**80.02**	0	44.51
Holepuncher	13.4	42.63	0.30	**62.40**
Iron	37.8	74.97	8.86	**78.01**
Lamp	34.5	71.11	8.20	**84.5**
Phone	19.9	47.74	0.18	**65.27**
Average	28.9	55.95	2.42	**63.59**

We compare our method with the state-of-the-art approaches Yolo-6D(Tekin et al., 2018), Zhan(Zhao et al., 2020) and SSD-6D (Kehl et al., 2017), which

Table 2: Comparison of our method with state-of-the-art work on Occluded LINEMOD dataset in terms of ADD-0.1 metric and REP-5px metric. We present percentages of correctly estimated pose and highlight the best result among those by **bold** numbers.'-' denote the results not in the original paper.

Method / Object	ADD-0.1				REP-5px		
	PoseCNN	Heatmaps	Seg-drive	Our	iPose	Yolo-6D	Our
Ape	9.6	16.5	12.1	**18.87**	24.2	7.0	**54.69**
Can	45.2	42.5	39.9	**50.52**	30.2	11.2	**44.82**
Cat	0.9	2.8	8.2	**15.38**	12.3	3.6	**53.73**
Driller	41.4	**47.1**	45.2	34.0	-	1.4	**17.49**
Duck	19.6	11.0	17.2	**27.00**	12.1	5.1	**51.91**
Eggbox	22.0	**24.7**	22	20.62	-	-	**41.37**
Glue	38.5	**39.5**	38.5	26.43	25.9	6.5	**43.72**
Holepuncher	22.1	21.9	**36.0**	32.0	20.6	8.3	**31.78**
Average	24.9	25.8	27.0	**28.1**	20.8	6.2	**42.43**

run under a similar setting. In TABLE 1, the competing methods are presented results. On average, our method outperforms all the considered competitors by a margin of at least 7% or more. We also find that our algorithm is more effective for small-size objects. For example, with the camera model whose diameter is 17.24 cm, the estimated pose accuracy increases by nearly 30%. Even when compared with some RBG-D based methods such as SSD-6D, for which the average accuracy reaches 76.3%, our method is still competitive. A possible reason that our method gives a less accurate results than Yolo-6D for glue is related to the shape of glue. The side of glue object is so narrow that hard to extra accurate side counter.

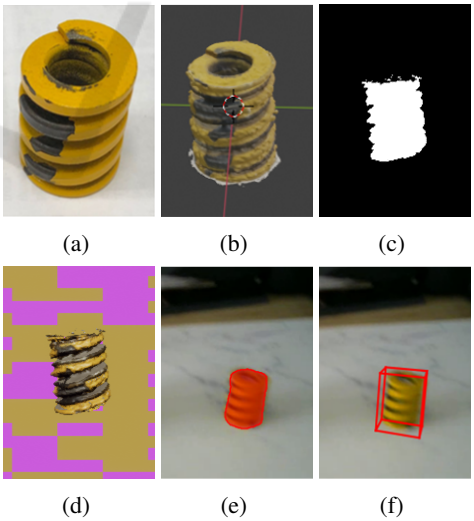
4.3 Multiple Object Instance Pose Estimation

The Occlusion-LINEMOD is a multi-objective estimation benchmark which contains 8 objects and 1214 images. As its name shows, a few objects in the images are heavily occluded, which makes estimation extremely difficult.

To create training data, we follow the same data selection setting as in the previous evaluation. Due to that every image contains several instances, we modify our training strategy: the training epoch increases from 90 to 160. The first 20 epochs train network heads with 0.004 learning rate. Then, using the same learning rate, we fine tune layers from ResNet stage 4 and up during the next 10 epochs. After that, we train all layers for 70 epochs. This initial learning rate value can make training convergence quickly. In the next 20 epochs, the learning rate is decreased by 10 in all layers. Finally, learning rate is set to 0.00004 in order to fine tune all layers in the final 20 epochs. Through twice learning rate tuning, we can obtain a minimize loss. This setting achieves excellent performance in our experiments. In addition, the segmentation loss weight α_3 changes to 40 in order to overcome excessive occlusion in the image.

As can be seen from the TABLE 2, our work outperforms other methods, such as PoseCNN(Xiang et al., 2017), Heatmaps(Oberweger et al., 2018), Seg-drive(Hu et al., 2019), iPose(Jafari et al., 2018), Yolo-6D(Tekin et al., 2018), in both ADD-0.1d metric and REP-5px metric. In Fig.7, we can notice that the estimated pose is still accurate with partial occlusion. But if the visibility of the object is too low, the estimation will fail.

4.4 Application to Real-world Object



(a) (b) (c) (d) (e) (f)

Figure 5: The application in real world object: (a) Real object. (b) Object model. (c) Synthesis mask. (d) Synthesis RGB image. (e) Detected mask. (f) Estimated pose.

The object models in the standard data set are precise, and the annotations are accurate. However, in the real

Figure 6: **Qualitative results on LINEMOD.** First row : the original images. Second row: the predicted object class, 2D bounding box and segmentation. Third row: 6D pose represented by 3D bounding boxes which green is the ground truth and the red is estimated.

Figure 7: **Qualitative results on Occluded LINEMOD.** First row : the original images. Second row: the predicted object class, 2D bounding box and instance segmentation(different color means different class). Third row: 6D pose represented by 2D projection contour which green is the ground truth and the other color is estimated. Forth row: Area screenshot, the first three columns is success cases and the last three columns is fail cases.

world, it is hard to obtain a perfect object model and annotate poses on authentic images. We consider synthetic images to train so that this method can apply our method on a broader range of objects.

In our experiment, the object model shown in the Fig.5(b) is obtained by structure from motion (SFM) (Wu et al., 2011) method, which can reconstruct an object model using the object images capturied from different angles. Then, utilising the NVIDIA Deep learning Dataset Synthesizer (NDDS) tool (To et al., 2018) generates synthetic training data. Finally, we feed the training data into the neural network. In this way, the pipeline of 6D object pose estimation can be more generic.

5 CONCLUSIONS

We have introduced a new method to detect an object class, segment instance and estimate object 6D pose simultaneously from a single RGB image. Our method can predict object orientation and calculate

translation without a PnP process. What's more, we propose a novel pose refinement algorithm Contour-Align by aligning the mask contour and the 2D projection contour for the single RGB image. This refinement technique can be applied to most of the post-processing of RBG based 6D estimation. Furthermore, the evaluation shows our work surpasses current state-of-the-art methods. Therefore, our work is encouraging because it indicates that it is feasible to accurately predict the 6D pose object pose in a cluttered environment using RGB data only. An interesting future work is to improve the estimation accuracy when the CAD model is unavailable.

ACKNOWLEDGEMENTS

This work was supported by EPSRC Grant No.EP/R026084/1, Robotics and Artificial Intelligence for Nuclear (RAIN), UK.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283.

Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., and Rother, C. (2014). Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer.

Brachmann, E., Michel, F., Krull, A., Ying Yang, M., Gumhold, S., et al. (2016). Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3364–3372.

Do, T.-T., Cai, M., Pham, T., and Reid, I. (2018). Deep-6dpose: Recovering 6d object pose from a single rgb image. *arXiv preprint arXiv:1802.10367*.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., and Lepetit, V. (2011). Gradient response maps for real-time detection of textureless objects. *IEEE transactions on pattern analysis and machine intelligence*, 34(5):876–888.

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., and Navab, N. (2012). Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer.

Hodan, T., Barath, D., and Matas, J. (2020). Epos: estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11703–11712.

Hu, Y., Fua, P., Wang, W., and Salzmann, M. (2020). Single-stage 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2930–2939.

Hu, Y., Hugonot, J., Fua, P., and Salzmann, M. (2019). Segmentation-driven 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3385–3394.

Jafari, O. H., Mustikovela, S. K., Pertsch, K., Brachmann, E., and Rother, C. (2018). ipose: instance-aware 6d pose estimation of partly occluded objects. In *Asian Conference on Computer Vision*, pages 477–492. Springer.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., and Navab, N. (2017). Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1521–1529.

Kendall, A. and Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5974–5983.

Krull, A., Brachmann, E., Nowozin, S., Michel, F., Shotton, J., and Rother, C. (2017). Poseagent: Budget-constrained 6d object pose estimation via reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6702–6710.

Lepetit, V., Moreno-Noguer, F., and Fua, P. (2009). Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155.

Li, Y., Wang, G., Ji, X., Xiang, Y., and Fox, D. (2018). Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Manhardt, F., Kehl, W., Navab, N., and Tombari, F. (2018). Deep model-based 6d pose refinement in rgb. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 800–815.

Michel, F., Kirillov, A., Brachmann, E., Krull, A., Gumhold, S., Savchynskyy, B., and Rother, C. (2017). Global hypothesis generation for 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 462–471.

Oberweger, M., Rad, M., and Lepetit, V. (2018). Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In *Proceedings of the European*

Conference on Computer Vision (ECCV), pages 119–134.

Park, K., Patten, T., and Vincze, M. (2019). Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7668–7677.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Rad, M. and Lepetit, V. (2017). Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3836.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Tekin, B., Sinha, S. N., and Fua, P. (2018). Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 292–301.

To, T., Tremblay, J., McKay, D., Yamaguchi, Y., Leung, K., Balanon, A., Cheng, J., and Birchfield, S. (2018). Ndds: Nvidia deep learning dataset synthesizer.

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T. (2014). scikit-image: image processing in python. *PeerJ*, 2:e453.

Wang, C., Xu, D., Zhu, Y., Martín-Martín, R., Lu, C., Fei-Fei, L., and Savarese, S. (2019). Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3343–3352.

Wu, C. et al. (2011). Visualsfm: A visual structure from motion system, 2011. *URL http://www. cs. washington. edu/homes/ccwu/vsfm*, 14:2.

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

Zhao, W., Zhang, S., Guan, Z., Luo, H., Tang, L., Peng, J., and Fan, J. (2020). 6d object pose estimation via viewpoint relation reasoning. *Neurocomputing*.