

RMTrack: 6D Object Pose Tracking by Continuous Image Render Match

ENYUAN CAO

Institute of Automation, Chinese Academy of Sciences; University of Chinese Academy of Sciences, School of Artificial Intelligence

XIAOYANG ZHU

Institute of Automation, Chinese Academy of Sciences

HAITAO YU

Institute of Automation, Chinese Academy of Sciences

YONGSHI JIANG

Institute of Automation, Chinese Academy of Sciences

Estimating the 6D pose of a known object has very important applications in augmented reality and robot operations. This problem is challenging because of the clutter of the scene, the diversity of objects, and the complexity of lighting and textures. In this work, we propose a deep learning architecture for 6D object pose estimation, and a neural network that can predict object movement. By learning to predict the relative pose between the observation of current frame and the rendered image of previous prediction, the pose of the object can be tracked robustly for a long time. We have also introduced an efficient way of representing object motion, which can reduce the influence of the field of view and object scale so that this method has a strong cross-dataset generalization. We have conducted a lot of experiments on the LINEMOD dataset, the OccludedLINEMOD dataset, and the YCB dataset to show that this method can provide accurate pose estimation using only color images as input while being highly robust to occlusion.

CCS CONCEPTS • Computing methodologies • Artificial intelligence • Computer vision • Computer vision problems • Tracking

Additional Keywords and Phrases: 6D pose tracking, Deep learning

ACM Reference Format:

First Author's Name, Initials, and Last Name, Second Author's Name, Initials, and Last Name, and Third Author's Name, Initials, and Last Name. 2018. The Title of the Paper: ACM Conference Proceedings Manuscript Submission Template: This is the subtitle of the paper, this document both explains and embodies the submission format for authors using Word. In Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.

1 INTRODUCTION

Recognizing objects in 3D space and estimating their pose is important in many applications. For example, in robot operation tasks, accurate pose estimation provides vital information for grasping and motion planning. In augmented reality applications, object pose estimation can realize the interaction between humans and virtual objects. This article focuses on tracking the 6-degree-of-freedom (6D or 6DoF) pose of an object from continuous RGB images, that is, rotation and translation in 3D space. This problem is challenging because of

the complexity of the application scenario, and, objects have different 3D shapes, and their appearance on the image is affected by lighting conditions, objects occlusion, and etc [28].

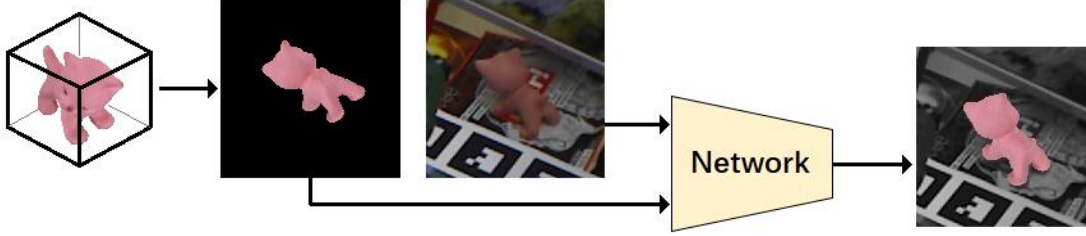


Figure 1: We propose a deep learning architecture for 6d pose tracking, and a novel deep neural network that can learn to predict the relative pose between the current observation and the previously predicted synthetic model rendering to track the object.

Traditionally, the problem of 6D object pose estimation is solved by matching the feature points [14] between the 3D model and the image or establishing the corresponding relationship between the object model and the image contour. Unfortunately, they rely on hand-made features that are not robust to image changes and background clutter [26]. The end-to-end neural network is trained based on the deep learning method, which takes a single image as input and outputs its corresponding posture. However, most methods do not effectively use the information of the image sequence.

In this work, we propose a data-driven optimization strategy and a novel deep neural network that can learn to predict the relative pose between the current observation and the previously predicted synthetic model rendering to track objects robustly for a long time, as shown in Figure 1. In addition, we also propose an efficient way of representing object motion, which can reduce the influence of the field of view and object scale, so that this method has strong cross-dataset generalization.

We evaluate our approach on LINEMOD [10], Occlusion LINEMOD [3] and YCB-Video datasets [28]. These are widely used benchmark datasets for 6D pose estimation. Experimental results show that, within the RGB only methods, our method achieves the most advanced performance on the Occlusion Linemod dataset, and achieves comparable performance on other datasets.

This paper is organized as follows. After discussing related work, we introduce RMTrack for 6D object pose tracking, followed by experimental results and a conclusion.

2 RELATED WORK

In recent years, research on tracking and detection has mainly focused on methods using RGB-D data. Although these methods perform better than methods based only on monocular RGB image data, these methods are difficult to apply in sunlight, and there are also limitations on the distance of the camera due to the limitation of depth sensors. Therefore, in this article, we will not include these works for comparison.

Methods for estimating the position of objects can be roughly divided into feature point-based methods, region-based methods and deep learning-based methods. The method based on feature points first extracts local features from the points of interest or each pixel in the image and then matches the features on the 3D model to establish a 2D to 3D correspondence, from which the 6DoF pose can be restored [14][20][17][17]. Feature-based methods are fast and accurate and can handle occlusion between objects. Most commercial

software on Augmented Reality use this method (such as ARKit, ARCore, Vuforia, etc.). However, this method requires the object to have sufficient texture to calculate local features [23]. This is because the traditional feature point extraction algorithm is at the pixel level and the degree of abstraction is low, so it is difficult to effectively track objects with weak surface texture, which makes such commercial software have limitations in application scenarios.

In order to deal with untextured objects, region-based methods are proposed [25]. Generally, the region-based method distinguishes the foreground area and the background area corresponding to the object. For modeling the composition of each pixel, differences in image statistics (such as colour) are used. Based on the 3D geometry of the object, the goal is to find the pose that best explains the two areas[9][10][4]. This algorithm estimates the pose only based on the contour of the object to be tracked, which effectively solves the problem that the algorithm cannot track low textured objects in feature-based methods, but correspondingly, this kind of method cannot solve the object with weaker contour features (such as football) [24][25]. Besides, this kind of methods cannot handle the occlusion between objects well, because if the object is occluded, the similarity score of the area will be lower [12]. In addition, these kinds of methods completely discard the part of the image that contains the largest amount of information, resulting in inferior performance to the method based on feature points in most cases, so this method is generally used in special occasions where objects need to be detected. Most of the traces are weak textures [1]. Combining traditional methods may be a good way to learn from each other [22], but in the process of combining multiple methods, many hand-designed standards are used, such as when to use feature points, when to use contours, etc. This will further reduce the adaptability of the algorithm in complex scenarios and cannot solve the nature of the problem.

Recent approaches apply machine learning, especially deep learning, for 6D pose estimation. The common method of deep learning-based methods is to perform semantic segmentation on the image first, and fit the pose based on the information of the segmented image and the three-dimensional model. There are many ways to fit the pose, some are based on key point matching [18][8], some are based on the center of gravity of the object after segmentation [8][28], and some are regression based on pixel-level coordinates [27][3][5][2]. The main advantage of this kind of method is that the deep learning method makes judgments based on the high-order features of the object. Compared with the hand-crafted likelihood functions and features in traditional methods, deep learning is more able to adapt to complex environments and lighting conditions [26][21]. Deep learning methods mostly focus on predicting the pose of an object from a single picture, which is actually slightly different from tracking [15] [29]. A few methods are to track the object pose on the time series of the image, such as DeepIM [13], which can greatly improve the speed, while maintaining or even improving the quality of the pose. This type of method uses some classic backbones [7] to prove that deep learning can fit relative poses only based on the residuals of the images [32], and is a pioneer in applying deep learning for object pose tracking. This article will focus on the research in this direction and make improvements on the basis of such methods.

3 APPROACH

3.1 Definition

To represent the position and posture transformation of an object in a three-dimensional space, we can first represent the object as a set of data points X in a three-dimensional coordinate system.

The 6-degree-of-freedom(6D or 6DoF) transformation of an object can be decomposed into a 3-degree translation and a 3-degree rotation. The translation is simple, it represents the change of the position of the object in space, just add it to the coordinates of the data point, $x' = x + t$.

Rotation is relatively complicated, and there are many kinds of representation methods, such as rotation matrix, rotation vector, Euler angle, quaternion, etc. These representation methods have their own adaptive application scenarios. The rotation matrix, R , is a 3x3 matrix. You only need to multiply it with the original coordinates to get the rotated coordinates, $x' = Rx$. Because of its simple calculation, it is widely used in computer programs for visualization, and it is also the choice of many well-known data sets.

In this case, the entire 6D attitude transformation can be expressed as $x' = Rx + t$. It is further extended to the form of secondary coordinates. The 6D pose transformation of an object composed of a set of data points x can be written as

$$\begin{bmatrix} x' \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} \quad (1)$$

However, rotation has only 3 degrees of freedom, while the rotation matrix consists of 9 values, which means this representation is redundant [16]. For example, the rotation matrix must be an orthogonal matrix. Using such an expression as the output of a neural network is obviously an inefficient way.

The method of using 3 values to represent the rotation is the rotation vector. A three-dimensional vector is used to represent the rotation in the three-dimensional space, its direction points to the axis of rotation, and the mold length is the angle of rotation, in circumference.

In order to maximize the calculation efficiency, the neural network in the next section will use the rotation vector to represent the rotation, but it will be converted into a rotation matrix when the image is rendered and the result is calculated.

3.2 Neural Network Design

The neural network takes the image I_t of the current frame and the rendered image I_{t-1} generated based on the pose of the previous frame as input, as shown in Figure 2.

The Network has two parts: feature extraction layers that using convolutional neural networks(CNN), and backbone using Vision Transformer(ViT). Although many recent works based on the transformer structure claim to completely adopt the transformer structure and perform better than the best CNN-based networks [6][30][33], they all use a layer of mapping to map image blocks into tokens. Although the activation function is not added in this step, it is actually a convolution operation. In other words, in terms of feature extraction, convolution is still the most classic structure. Set the image size to 112×112 , superimpose the real image and the rendered image, and get the input data as $112 \times 112 \times 6$. Through 3-layer convolution, the data becomes $14 \times 14 \times 384$, and then input it into ViT backbone.

Note that there is no pooling layers in the network, because the convolution structure after using pooling will lose the sensitivity of the network to position. If the object position has a few pixels of translation, or a small rotation caused a small feature Displacement will be completely filtered out under such a structure. However, the 6d object pose estimation problem requires very accurate estimation of these contents, especially in the first few feature extraction layers, the displacement of a few pixels may have an objective impact on the result.

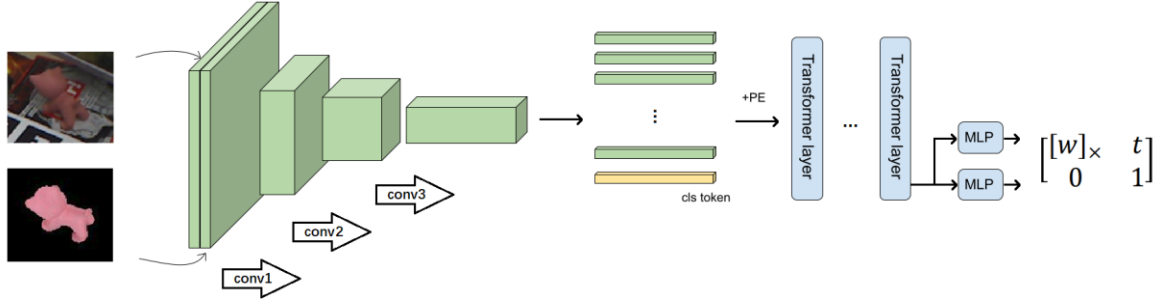


Figure 2: Proposed RMTrack architecture: It takes RGB images as input, return the relative change in posture.

The structure of the ViT backbone can be expressed as

$$T_0 = [t_{cls}; T] + PE \quad (2)$$

$$T_i = \text{MLP}(\text{MSA}(T_{i-1})) \quad (3)$$

$$y = fc(LN(T_b)) \quad (4)$$

where T_i denotes one Transformer layer ($0 \leq i \leq b$), MSA denotes the multihead self-attention operation with layer normalization and “MLP” is the multilayer perceptron with layer normalization in the standard Transformer [6]. E is Sinusoidal Position Embedding, LN is layer normalization, fc is one fully-connected layer for classification and y is the output prediction. The transformer here uses a deep-narrow structure, with layer number $b = 14$, hidden dim 384, mlp layer 1152, which proved to be very efficient in t2t-vit [30]

3.3 Image pre-processing

If the object in the input image is very small, it is difficult to extract useful features for matching. To obtain sufficient details for posture matching and reduce unnecessary computational expenses, it is a common practice to zoom in the object before sending it into the neural network. This article focuses on the tracking problem, so the position of the object in the current frame should be close to the position of the previous frame, so it can be directly intercepted at the position of $It-1$ in the previous frame. When the translation of the object is $t = [tx, ty, tz]$, under the pinhole camera model, the position of the object on the image can be expressed as

$$x = f \frac{t_x}{t_z} + \frac{x}{2} \quad (5)$$

$$y = f \frac{t_y}{t_z} + \frac{Y}{2} \quad (6)$$

where x, y are the pixels on the image, X, Y are the size of the image, f is the focal length. They are all measured in pixels.

The problem is that due to different camera angles, objects may appear in different positions in the picture. Under the pinhole camera model, the imaging of the same object under different camera angles is different, as shown in Figure 3. In order to deal with this problem, a further step of view field transformation is required before the image is intercepted.

$$x = x_0 + x' \frac{t_z^2 + t_z'^2}{t_z^2} \quad (7)$$

$$y = y_0 + y' \frac{t_z^2 + t_z'^2}{t_z^2} \quad (8)$$

where x', y' are the pixels on the image after the view field transformation, x, y are the pixels on the original image.

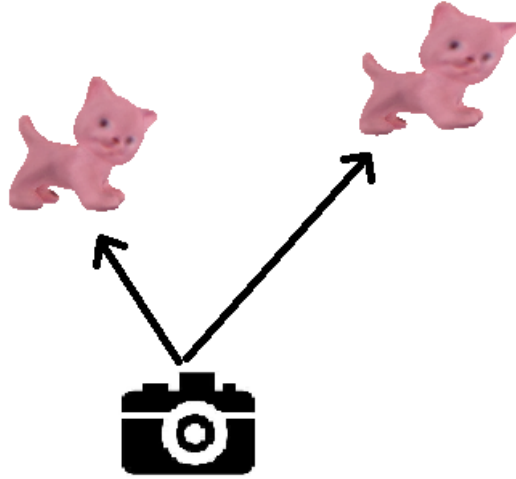


Figure 3: Images of the same field of view under different camera angles. It can be seen that the object close to the corner is significantly stretched.

The estimation of object rotation is not determined by the size of the object and the distance of the object from the camera, but this is not the case for translation. Different objects have different diameters. Under the combined effect of the size of the object and the distance from the camera, objects with a large difference in size may appear to be about the same size in the picture. In order to reduce the influence of object scale, we divide the change of position t' by the translation of the z -axis of the previous frame, t'/t_z as a label to train the neural network, which means the neural network can learn the relative value of position change.

4 EXPERIMENTS

4.1 Datasets

The LineMOD dataset [10] is a widely-used benchmark datasets for 6D pose estimation dataset. The data set contains a total of 13 objects, and each object contains approximately 1200 pictures. There are many challenges on the LineMOD dataset, such as low resolution, cluttered scenes, untextured objects, and changing lighting conditions.

Occlusion LineMOD [3] is a subset of the LineMOD data set. It contains 1214 pictures, and each picture contains 8 objects. The main challenge of this dataset is severe occlusion, especially for small targets.

The YCB-Video dataset [28] contains 21 objects with rich textures, distributed in 92 RGB-D videos. Usually, 80 of them are used for training and the remaining 12 videos are used for testing. These videos are split into frames of images, and each video has about 1500 images. There are many stacking scenes with partially occluded objects.



Figure 4: Visualization of some of the results on the Occlusion LINEMOD dataset.

4.2 Evaluation metric

We use two standard metrics to evaluate our method: the 2D projection metric [28] and the average 3D distance of model points (ADD, ADD-S) metric [10].

The most commonly used metrics for object pose estimation are ADD and ADD-S. ADD metric is defined as the average Euclidean distance between model points transformed with the ground truth and the predicted pose respectively:

$$ADD = \frac{1}{N} \sum_{i=1}^N ||(Rpi + t) - (\hat{R}pi + \hat{t})|| \quad (9)$$

where N is the number of model points pi , R and t are the rotation and translation of ground truth pose, \hat{R} and \hat{t} are the rotation and translation of predicted pose.

ADD-S metric is designed for symmetric objects and calculates the average distance with the closest point:

$$ADD - S = \frac{1}{N} \sum_{i=1}^N \min_{j=1}^N ||(Rpi + t) - (\hat{R}pj + \hat{t})|| \quad (10)$$

ADD(-S) means applying ADD for asymmetric objects and ADD-S for symmetric objects.

2D Projection metric focuses on the matching of pose estimation on 2D images. This metric is considered to be important for applications such as augmented reality.

$$Proj.2D = \frac{1}{N} \sum_{i=1}^N ||K(Rp_i + t) - K(\hat{R}p_i + \hat{t})|| \quad (11)$$

where K denotes the intrinsic parameter matrix of the camera, and $K(Rx + t)$ indicates transforming a 3D point onto the image. The unit of measurement here is pixels.

When calculating the accuracy, a threshold will be set. Generally, the threshold of ADD(-S) is set to be 10% of the diameter of the object, and the threshold of proj2d is set to be 5 pixels. If the result is less than the threshold, it is considered to be correct prediction.

4.3 Implementation Details

The training process randomly generates a rendered image with small translation and rotation for each frame of real image, so the rendered image will never be exhausted.

Note that, the Linemod dataset and Occlusion Linemod datasets are not continuous images. In other object tracking work, the usual approach is to randomly or use a baseline method to give an initial value. So when we verify these data sets, we perform random translation and rotation on the ground truth as the initialization of each frame.

4.4 Results and analysis

We compare with other state-of-the-art methods which take RGB images as input and output 6D object poses. The visualization results are shown in Figure 4.

Table 1: The accuracies of our method and the baseline methods on the LINEMOD dataset in terms of the ADD(-S) metric, where glue and eggbox are considered as symmetric objects.

Method	BB8	PoseCNN	PVNet	DeepIM	DPOD	RePose	Ours
Ape	27.9	-	43.62	77	87.7	79.5	79.3
Benchwise	62	-	99.9	97.5	98.5	100	95.5
Cam	40.1	-	86.86	93.5	96.1	99.2	93.4
Can	48.1	-	95.47	96.5	99.7	99.8	95.8
Cat	45.2	-	79.43	82.1	94.7	97.9	90.4
Driller	58.6	-	96.43	95	98.8	99	94.4
Duck	32.8	-	52.58	77.7	86.3	8.03	80.9
Eggbox	40	-	99.15	97.1	99.9	100	86.9
Glue	27	-	95.66	99.4	98.7	98.3	97.1
Holepuncher	42.4	-	81.92	52.8	86.9	96.9	87.8
Iron	67	-	98.88	98.3	100	100	94.3
Lamp	39.9	-	99.33	97.5	96.8	99.8	94.7
Phone	35.2	-	92.41	87.7	94.7	98.9	87.4
Mean	43.6	62.7	86.3	88.6	95.2	96.1	90.7

Table 1 shows the comparison of our method with other works on Linemod dataset, including BB8 [19], PoseCNN [28], PVNet [18], DeepIM [13], DPOD [31], and RePOSE [11]. It can be seen that, our method achieves comparable performance.

Table 2: The accuracies of our method and the baseline methods on the Occlusion LINEMOD dataset in terms of the ADD(-S) metric, where glue and eggbox are considered as symmetric objects.

Method	PoseCnn	PVNet	RePose	Ours
Ape	9.6	15.81	31.1	47.6
Can	45.2	63.3	80	49.2
Cat	0.93	16.68	25.6	36.5
Duck	19.6	25.24	73.1	56.8
Driller	41.4	65.65	43	45.9
Eggbox	22	50.17	51.7	56.5
Glue	38.5	49.62	54.3	63.4
Holepuncher	22.1	39.67	53.6	59.9
Mean	24.9	40.77	51.6	52.8

Table 2 shows the comparison of our methods with PoseCNN [28], PVNet [18], RePOSE [11] in terms of the ADD(-S) metric. It can be seen that among RGB only methods, our method achieves the state-of-the-art.

Table 3: The accuracies of our method and the baseline methods on the YCBVideo dataset in terms of 2D projection and ADD(-S) AUC.

Method	PoseCnn	Oberweger	PVN	Ours
ADD(-S)	61	72.8	73.4	75.1
2D Projection	3.72	39.4	47.4	47.2

Table 3 shows the results on YCB dataset. It can be seen that among RGB only methods, our method achieves state-of-the-art under ADD(-S) metric, and very close to the state-of-the-art under 2D Projection metric.

4.5 Applying to out of training objects

In our method, the algorithm does not rely on memorizing the shape of each object, but returns to the posture change based on the feature movement between the two images. Therefore, our algorithm can also achieve good performance when applying to out of training objects. Figure 5 shows the performance of each object under the cross data set under the Occlusion Linemod data set. Here, for each object, the neural network was trained on all other objects in both Linemod dataset and Occlusion Linemod dataset, and then applied to this object. The results can prove that even for an untrained object, RMTrack can still return the pose change. However, due to the training data is not enough, the result is slightly lower than the trained group, especially when the threshold is small.

4.6 Applying to out of training objects

On RTX 2070 GPU, RMTrack is running at 21 fps per object.

Since the rendering of the objects can be implemented in many ways in the application, and further time can be saved through multiple processes, the above-mentioned running time does not include the rendering time.

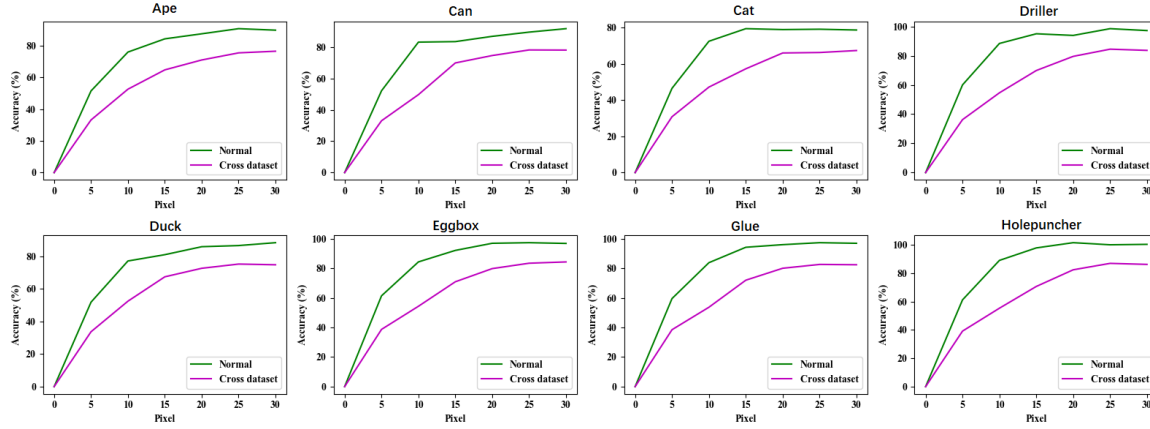


Figure 5: Performance on out of training dataset on the Occlusion LINEMOD dataset.

5 CONCLUSION

In this work, we propose an efficient and robust long-term 6D object attitude tracking framework RMTrack. Given the initial 6D pose estimation of an object, we design a new transformer-based deep neural network to directly output a relative pose transformation to improve the pose estimation. Experimental results show that the combination of learnable features and efficient non-linear optimization can produce accurate 6D object poses. In addition, our experiments on the crossdata set verify that the neural network can perform feature-to-posture fitting, rather than simply remembering the shape of each object. The future direction is to completely separate the dependence of the neural network on the training object, so that it can match the trained object on the untrained object based only on the characteristics of the image.

REFERENCES

- [1] Kittipat Apichatrisorn, Xukan Ran, Jiasi Chen, Srikanth V Krishnamurthy, and Amit K Roy-Chowdhury. Frugal following: Power thrifty object detection and tracking for mobile augmented reality. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, pages 96–109, 2019.
- [2] Erin Barsan and Hilary Wang. Dpoe-n: Providing professional development funding for digital preservation. In *MAC Annual Meeting Presentations*, volume 2021. Iowa State University Digital Press, 2021.
- [3] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer, 2014.
- [4] Zhe Cao, Yaser Sheikh, and Natasha Kholgade Banerjee. Real-time scalable 6dof pose estimation for textureless objects. In *2016 IEEE International conference on Robotics and Automation (ICRA)*, pages 2441–2448. IEEE, 2016.
- [5] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.
- [6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [7] 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 *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.
- [8] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11632–11641, 2020.
- [9] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE transactions on pattern analysis and machine intelligence*, 34(5):876–888,

2011.

- [10] 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, pages 548–562. Springer, 2012.
- [11] Shun Iwase, Xingyu Liu, Rawal Khiradkar, Rio Yokota, and Kris M Kitani. Repose: Real-time iterative rendering and refinement for 6d object pose estimation. arXiv preprint arXiv:2104.00633, 2021.
- [12] George Alex Koulrieris, Kaan Akşit, Michael Stengel, Rafał K Mantiuk, Katerina Mania, and Christian Richardt. Near-eye display and tracking technologies for virtual and augmented reality. In Computer Graphics Forum, volume 38, pages 493–519. Wiley Online Library, 2019.
- [13] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In Proceedings of the European Conference on Computer Vision (ECCV), pages 683–698, 2018.
- [14] David G Lowe. Object recognition from local scale-invariant features. In Proceedings of the seventh IEEE international conference on computer vision, volume 2, pages 1150–1157. Ieee, 1999.
- [15] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In Proceedings of the European Conference on Computer Vision (ECCV), pages 800–815, 2018.
- [16] Diego Marcos, Michele Volpi, Nikos Komodakis, and Devis Tuia. Rotation equivariant vector field networks. In Proceedings of the IEEE International Conference on Computer Vision, pages 5048–5057, 2017.
- [17] Georgios Pavlakos, XiaoWei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In 2017 IEEE international conference on robotics and automation (ICRA), pages 2011–2018. IEEE, 2017.
- [18] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pynet: Pixel-wise voting network for 6dof pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4561–4570, 2019.
- [19] 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 Proceedings of the IEEE International Conference on Computer Vision, pages 3828–3836, 2017.
- [20] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. International journal of computer vision, 66(3):231–259, 2006.
- [21] Weijian Ruan, Wu Liu, Qian Bao, Jun Chen, Yuhao Cheng, and Tao Mei. Poinet: pose-guided ovonic insight network for multi-person pose tracking. In Proceedings of the 27th ACM International Conference on Multimedia, pages 284–292, 2019.
- [22] Manuel Stoiber, Martin Pfanne, Klaus H Strobl, Rudolph Triebel, and Alin Albu-Schäffer. A sparse gaussian approach to region-based 6dof object tracking. In Proceedings of the Asian Conference on Computer Vision, 2020.
- [23] Fulin Tang, Yihong Wu, Xiaohui Hou, and Haibin Ling. 3d mapping and 6d pose computation for real time augmented reality on cylindrical objects. IEEE Transactions on Circuits and Systems for Video Technology, 30(9):2887–2899, 2019.
- [24] Henning Tjaden, Ulrich Schwanecke, and Elmar Schomer. Real-time monocular pose estimation of 3d objects using temporally consistent local color histograms. In Proceedings of the IEEE international conference on computer vision, pages 124–132, 2017.
- [25] Henning Tjaden, Ulrich Schwanecke, Elmar Schömer, and Daniel Cremers. A region-based gauss-newton approach to real-time monocular multiple object tracking. IEEE transactions on pattern analysis and machine intelligence, 41(8):1797–1812, 2018.
- [26] Bowen Wen, Chaitanya Mitash, and Kostas Bekris. Data-driven 6d pose tracking by calibrating image residuals in synthetic domains. arXiv preprint arXiv:2105.14391, 2021.
- [27] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3109–3118, 2015.
- [28] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199, 2017.
- [29] Yuliang Xiu, Jiefeng Li, Haoyu Wang, Yinghong Fang, and Cewu Lu. Pose flow: Efficient online pose tracking. arXiv preprint arXiv:1802.00977, 2018.
- [30] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. arXiv preprint arXiv:2101.11986, 2021.
- [31] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1941–1950, 2019.
- [32] Jirong Zhang, Chuan Wang, Shuaicheng Liu, Lanpeng Jia, Nianjin Ye, Jue Wang, Ji Zhou, and Jian Sun. Content-aware unsupervised deep homography estimation. In European Conference on Computer Vision, pages 653–669. Springer, 2020.
- [33] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020.
- [34] Kazumasa Oniki, Toshiaki Kikuchi, and Yuko Ozasa, "Training Data Generation Based on Observation Probability Density for Human Pose Refinement," Journal of Image and Graphics, Vol. 9, No. 2, pp. 50-54, June 2021. doi: 10.18178/joig.9.2.50-54
- [35] Naresh Kumar and Nagarajan Sukavanam, "Motion Trajectory for Human Action Recognition Using Fourier Temporal Features of Skeleton Joints," Journal of Image and Graphics, Vol. 6, No. 2, pp. 174-180, December 2018. doi: 10.18178/joig.6.2.174-180
- [36] Seyma Yucer and Yusuf Sinan Akgul, "3D Human Action Recognition with Siamese-LSTM Based Deep Metric Learning," Journal of Image and Graphics, Vol. 6, No. 1, pp. 21-26, June 2018. doi: 10.18178/joig.6.1.21-26

Authors' Background

(Please fill in the information of all authors.)

Name	Email	Position (Prof , Assoc. Prof. etc.)	Research Field	Personal Website
Enyuan Cao	caoey7@gmail.com	master student		
Xiaoyang Zhu	Xiaoyang.zhu@ia.ac.cn	researcher		
Haitao Yu	Haitao.yu@ia.ac.cn	researcher		
Yongshi Jiang	Yongshi.jiang@ia.ac.cn	researcher		

*This form helps us to understand your paper better; **the form itself will not be published.**