How Does A Camera Look At One 3D CAD Object?

Chang Xing[†], Chengjiang Long[‡], Hao Guo[†], Yongwei Nie[♯], Yuan Zhang[†], Dehai Zhu[†], Qin Ma[†], Mengxiao Tian[†]

[†]China Agricultural University
Haidian, Beijing, China 100083
guohaolys@cau.edu.cn

[‡] Kitware Inc.
Clifton Park, NY, USA 12065
chengjiang.long@kitware.com

[♯]South China University of Technology
Guangzhou, Guangdong, China 510630
nieyongwei@scut.edu.cn

***Abstract*—Camera pose and the camera's rotation angles and translation vector (RT), are one-to-one relation with a 2D real image when the intrinsic parameter is fixed. In this paper, we propose a novel convolutional neural network (CNN) based framework to intelligently estimate the 6-DOF RTs from images taken on one 3D CAD object directly and indirectly, as well as visually verifying the correctness of the predicted RTs. Such a framework enables us to accurately interpret how a camera looks at the object. The direct way is simple and obtains lower average errors for the predicted RTs experimentally, while the indirect way utilizes the POSIT algorithm via landmarks and is able to avoid the non-Euclidean issue in rotation angles. To our best knowledge, we are the first one to estimate camera's RTs and effectively interprets how a camera looks at one 3D CAD object from the images taken on it. The experiments on four models quantitatively and qualitatively demonstrate the efficacy of our proposed approach.**

***Keywords*—camera pose, 6-DOF, rotation angle, translation vector, CNN, 2D real images, 3D CAD object, synthetical images.**

I. Introduction

With the ability to capture everyone's favorite moments from a completely new and breathtaking perspective, like a photographer, a smart robot can adjust its camera focus, move to a special position and choose a suitable viewpoint to shoot a picture of some objects.

According to the camera's imaging principle, assuming that the focus is fixed, if the camera looks at objects with different poses, then the captured images on those objects should be tototally different. Such a one-to-one relationship is relfected that each image corresponds to a specific camera pose. This observation movitates us to seek a way to estimate camera pose and accurately interpret how a camera looks at 3D objects from the perspective of images, which has not been well explored.

Intuitively, a camera's pose corresponds to camera's extrinsic parameter, which is 6-DOF (degree of freedom) parameter as $(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$, where $(\theta_x, \theta_y, \theta_z)$ is rotation angles from x, y and z axes and (t_x, t_y, t_z) is translation vector. Assuming that the camera focus is fixed, to estimate a RT from an image under the supervised learning framework, we have to face two issues. First, how to obtain the supervised ground-truth information of camera's RTs from images? Second, what kinds of visual feature can be used for such a supervised learning?

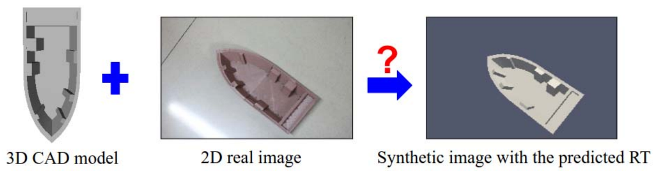


Fig. 1. Assuming that the camera's intrinsic parameter is fixed, given a 3D CAD model (left) and a real image (middle) taken on its object, how can we estimate its RT (extrinsic parameter), and then use the RT to generate a synthetic image (right) and visually verify its correctness?

However, there are no direct ways to obtain RT from images by manual annotation. Fortunately, the POSIT algorithm [2], which is able to estimate the 6-DOF pose of a 3D object in a single image. To make it simple, weuse the POSIT algorithm to obtain the RT from each image as supervised information for either training or evaluation. Now, as shown in Figure 1, we target the camera pose estimation problem as: given a 3D CAD model, the corresponding CAD object, the camera's intrinsic parameter and 2D view images, can we estimate the accurate RTs and visually check the correctness?

In recent years, the convolutional neural networks (CNN) [5] have won significant attention due to their success on learning feature representations. In particular, CNN has shown superior performance on standard object recognition tasks, which effectively learn complicated mappings while utilizing minimal domain knowledge. Hence, we can turn to CNN, in spite that most of 3D CAD models lack sufficient color information which make it not suitable to use traditional visual features like SIFT, LBP, SURF for images taken on them.

In this paper, we propose a novel CNN-based framework to estiamte RT from images automatically and intelligently, as well as visually verify the correctness of the predicted RTs. Firstly, to obtain the supervised RT information, we resort to the POSIT algorithm to indirectly calculate the RTs for each image with annotating a few number of image points associated with the correponding 3D points. Secondly, CNN has been proved experimentally as a very powerful feature extraction approach for visual tasks. And hence we design a AlexNet-variant network as a multi-output regression model which is able to obtain RTs directly and indirectly. In the direct way, the output are 6-DOF RTs. In the indirect way, we firstly obtain the predicted M landmarks, and then we calculate RTs via the POSIT algorithm using the predicted M landmarks and their corresponding 3D coordinates. Finally, combining with the 3D CAD model and the fixed intrinsic camera parameter, we generate synthetic images based on the predicted RTs and compare the synthetic images with the real images to verify the correctness. In this way, we are able to effectively intepret how a camera looks at one 3D CAD object.

To sum, the contribution of this paper lies on four-folds:

1) We are the first one to propose a CNN-based framework to estimate the camera's RTs from images directly and indirectly and flexiblely interprets how a camera looks at 3D object.
2) We are able to calculate RTs indirectly via the POSIT algorithm. By this way, the introduction of landmarks solves the non-Euclidean issue of rotation angles.
3) We also develop a visualization tool to visually verify the correctness of our predicted RTs. This enables a way to accurately intepret how a camera looks at the 3D object visually.
4) The experimental results on four groups of experiments on four different CAD models strongly show the efficacy of our proposed method.

II. Related work

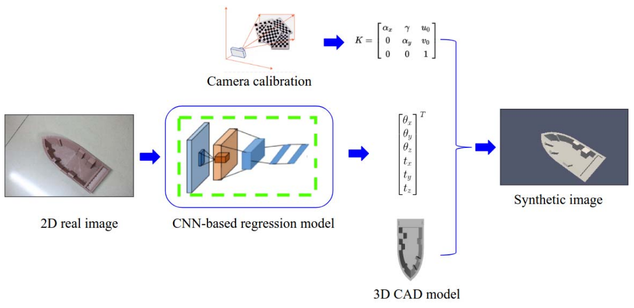
The related work falls into two categories: *object/camera pose estimation* and *deep learning*.

Object/camera pose estimation. Most of the existing approaches are mainly designed on the predefined discrete orientation/viewpoint bins in the viewing circle and formulated as a multi-class classification problem [10], [1] or regression function from local feature [14], [15]. Recently, there are several related work [8], [7] to estiamte the 6-DOF camera poses. In contract, we aim to obtain the camera's accurate and continous RTs of the single camera estimated from real images focus on a single 3D CAD object and accurately interpret how a camera looks at the object.

Deep learning models especially the convolutional neural network (CNN) have been developed to address the vision problems [4], [6] successfully. Even though CNN is good at extracting global features, it doesn't necessarily emphasize local discriminative features which are very critical for fine-grained tasks like pose estimation. Moreover, the capability of CNN for continuous multi-output regression tasks especially for RT estimation from images has not been fully explored, in spite of its success in dealing with multi-class classification.

III. Proposed approach

As illustrated in Figure 2, our proposed CNN-based framework has three components, *i.e.*, camera calibration, RT estimation through a CNN-based multi-output regression model, and visual verification of the predicted RTs via generating synthetic images and comparing it with the real image visually.



Camera calibration
$K = \begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$
2D real image
CNN-based regression model
$\begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \\ t_x \\ t_y \\ t_z \end{bmatrix}^T$
Synthetic image
3D CAD model

Fig. 2. The pipeline for our proposed CNN-based framework.

A. *Camera calibration*

We resort to a Camera Calibration Toolbox for Matlab[1] to estimate the camera intrinsic parameter K which is fixed and real images are captured around a 3D object in roughly semi-sphere space. We adjust the camera focus used for experiments initially and then fix it before collecting 20-25 checkboard images for camera calibration and a large number of real images shooting on the object. And hence we only need to take one-time camera calibration and then use the same obatined parameter K directly throughout the whole group of exepriments.

B. *CNN-based regression model for RT estimation*

In this paper, for each 3D CAD object, we firstly determine M non-coplanar 3D points $\{(x_k, y_k, z_k)\}_{k=1}^{M}$ arbitrarily from its CAD model, and use a camera with the fixed intrinsic parameter to collect a large number of 2D images taken on the object. Then we manually annotate the corresponding landmarks on each image associated with each 3D point and hence get image points $\{(x_{im}^k, y_{im}^k)\}_{k=1}^{M}$, where x_{im}^k and y_{im}^k indicate the projection position of k-th point in the image, respectively. With the availability of parameter K and M pairs of 3D points and 2D landmarks, we run the POSIT algorithm to calculate the RTs and take them as the ground-truths.

We propose two versions (*i.e.*, direct and indirect) of multi-output CNN regression model to estimate the RT for an arbitrary real image taken on the same 3D CAD object. Theoritically, we can modify most of advanced CNN network like VGG [12], Google-Net [13] and ResNet [3] and use them as our multi-output regression models. In this paper, we choose AlexNet network [9] due to two reasons. (1) AlexNet network is widely used and there are pre-trained models available online. (2) The number of our collected images are not large enough to fine-train a more complicated CNN networks which have a much larger number of parameters to learn.

We design two versions of AlexNet-variant network as multi-output regression model to predict RTs directly and indirectly. Different from the original Alexnet work, our CNN model is designed to solve the multi-output regression probem. We change the number of output in the last second layer "fc8" from 1000 into 6 in the direct model indicated as CNN-direct and $2M$ in the indirect model indicated as CNN-indirect. As

[1]Camera Calibration Toolbox for Matlab with URL: http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example.html.

for loss function, we use the sum of the individual Euclidean losses that measure the distance between the predicted vector and the actual vector. For CNN-indirect, we need the POSIT algorithm as one more step to calculate RTs from the predicted M landmarks indirectly.

It worths mentioning that we benefit from fine-tuning the existing online available AlexNet caffe model trained from large-scale ImageNet dataset.

Remark: CNN-direct is simpler than CNN-indirect which may pass forward the prediction landmark errors to the prediction RT errors due to the system error of the POSIT algorithm. However, CNN-direct has its own defect due to its ambiguous angle represenation for rotation angle, which is non-Euclidiean in some cases. For example, $-\pi$ and π are identical to each other in physical space, while their difference value is too large. This may cause failures sometimes.

C. Visual verification of correctness via synthetic image

Once we obtain the camera's RT $(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z)$, we can calculate the camera view-up normal vector, camera positon, and the camera's focal point. Combining with the fixed camera intrinsic parameter K, any 3D CAD object's coordinates can be projected into 2D image points in a render window.

In implementation, we use VTK. First, we load the corresponding 3D CAD model and set the intrinsic parameter of the virtual active camera as our camera's parameter K. Then we are able to update the virtual active camera's view-up normal vector, position and focal point based on the estimated RTs. Finally we get an active view on the CAD model and the render window can be screenshoted or saved as the synthetic images.

In this way, we flexiblely simulate how a camera looks at the 3D CAD object and takes photos. Then we visually compare the generated synthetic image with the testing image. If these two images look similar, then that is able to demonstrate our estimated RT is correct and strongly support the claim that our proposed model can accurately intepret how a camera looks at one 3D CAD model in the realworld situation.

IV. Experiments

We conduct experiments on three 3D CAD objects with the corresponding CAD models, *i.e.*, small ship, Bruce Lee and fish statue and boy angel, as shown in Figure 3. The vertex count and triangle count for these four models are summarized in the Table I.

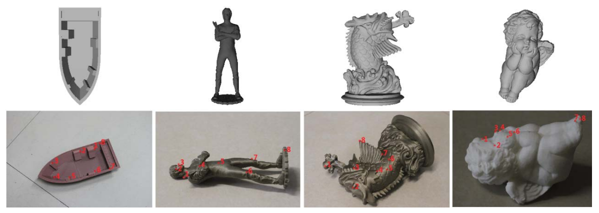


Fig. 3. 3D CAD models (top) and real images (bottom) taken on the models.

A. Dataset

For each 3D CAD object, we take more than 1000 real images with the size 5184×3465 on it from different viewpoints. The number of real images taken on these three models are 1286, 1480, 1582, and 1643, respectively.

As shown in Figure 3, we annotate $M = 8$ landmarks on each image to calculate the RTs via POSIT algorithm. Based on the ground-truth RTs, we summarize the range for each element of RT over all the images for each model in Table I. As we can see, the range for both x and z axis are roughly $(-\pi, \pi)$, and the range for y are within $(-\frac{1}{2}\pi, \frac{1}{2}\pi)$, which indicates all the images are taken around a half sphere.

For the data split for training and testing, we randomly select 90% of the total images for training and the rest are used for evaluation. Since there is no previous work to solve our problem, we implement two similar versions of SVM method (*i.e.* SVM-direct and SVM-indirect) using the bag-of-words feature combining both the SIFT and LBP descriptors from each image and take them as the baselines. To specify, SVM-direct uses SVM to predict RTs directly, and SVM-indirect uses SVM to predict landmarks and then run the POSIT algorithm to obtain RT indirectly.

The measurement used in this paper is average absolute regression error.

B. Experiments on the small ship model

We start experiments on the small ship model to evaluate the regression errors and take quantitative analysis. We also visualize the result to evaluate the correctness qualitatively.

In order to further verify our CNN-indirect to estimate the RTs, we summarize the average absolute landmark errors in Table II. The observations show that the predicted landmarks predicted by our CNN-indirect are much closer to the ground-truth landmarks, when compared to those landmarks predicted by SVM-indirect. Also, the average errors in x and y pixel position of landmarks are small to 2% of the image size 5184×3465. This indicates our proposed CNN-indirect performs well to predict the landmarks on image.

TABLE II
THE REGRESSION ERRORS OF LANDMARKS ON THE SMALL SHIP MODEL.

	SVM-indirect		CNN-indirect	
Landmark	x	y	x	y
1	486.644	443.346	103.879	70.935
2	375.221	336.351	85.591	64.372
3	635.839	532.093	101.433	81.571
4	609.213	599.830	104.455	83.095
5	427.491	396.943	89.598	71.911
6	480.253	367.553	93.728	74.155
7	683.239	538.170	118.916	89.815
8	718.722	601.149	118.915	90.845
Average	552.078	476.929	102.064	78.337

We also summarize the average absolute error of RTs obtained by four different methods in Table III. As we can see, both CNN-indirect and CNN-direct perform better than SVM-indirect and SVM-direct, and CNN-direct performs the best. Its average rotation angle error is 0.337, *i.e.*, approximately $19.3°$ in the range $[0°, 360°)$, and its average of translation

TABLE I
THE BRIEF INFORMATION FOR EACH CAD MODEL AND REAL IMAGES USED FOR EXPERIMENTS.

	small ship	Bruce Lee	fish statue	boy angel
Vertex number	9,378	703,248	233,184	554,082
Triangle number	3,126	234,416	77,728	184,694
Training number	1,157	1,332	1,423	1,478
Testing number	129	148	159	165
θ_x range	[-3.137, 3.136]	[-3.141, 3.140]	[-3.141, 3.141]	[-3.104, 3.138]
θ_y range	[-1.128, 1.367]	[-1.239, 1.045]	[0.505, 1.557]	[-1.402, 0.512]
θ_z range	[-3.139, 3.141]	[-3.136, 3.131]	[-3.129, 3.124]	[-3.135, 3.132]
t_x range	[0.648, 11.828]	[0.358, 7.061]	[4.632, 222.456]	[19.033, 156.619]
t_y range	[0.417, 8.659]	[0.048, 3.955]	[0.637, 164.567]	[6.471, 131.387]
t_z range	[14.983, 39.193]	[2.642, 21.057]	[194.158, 776.526]	[216.857, 635.711]

TABLE III
THE REGRESSION ERROR OF RTS ON THE SMALL SHIP MODEL.

	SVM		CNN	
Landmark	indirect	direct	indirect	direct
θ_x	2.270	2.102	0.866	**0.515**
θ_y	0.687	0.403	0.167	**0.128**
θ_z	1.659	1.425	**0.311**	0.368
θ_{avg}	1.539	1.310	0.448	**0.337**
t_x	3.427	1.209	**0.540**	0.561
t_y	2.257	1.010	0.469	**0.403**
t_z	15.633	2.951	3.652	**1.632**
t_{avg}	7.106	1.723	1.553	**0.865**

vector error is low to 0.865. All these observations strongly demonstrate the efficacy of our proposed multi-output CNN regression framework.

C. Experiments on other three models

To extend the evaluation on more complicated CAD models, we continue to run experiments on the rest of three models.

We summarize the prediction error of RTs for these three models in Table IV, V and VI, respectively. As we can see, both CNN-indirect and CNN-direct obtain much smaller errors in both rotation angles and translation vectors than both SVM-indirect and SVM-direct. CNN-direct achieves the smaller average errors, which is consistent with the observations on the ship model. The best average rotation angle errors for these three models are 0.219, 0.136 and 0.140 respectively. In other words, the rotation angle errors are $12.5°$, $7.8°$ and $8.0°$ in the range $[0°, 360°)$, respectively. Also, the average error of translation vectors estimated by the two versions of CNN is also acceptable when compared the real ranges shown in Table I. Obviously, such observation strongly demonstrates the ability of convolutional neural netowrk to estimate how the camera looks at the CAD models.

TABLE IV
THE REGRESSION ERROR OF RTS ON THE "BRUCE LEE" MODEL.

	SVM		CNN	
Landmark	indirect	direct	indirect	direct
θ_x	1.290	1.468	0.532	**0.407**
θ_y	0.297	0.221	0.089	**0.076**
θ_z	0.855	0.921	**0.132**	0.174
θ_{avg}	0.812	0.870	0.251	**0.219**
t_x	2.030	0.578	**0.188**	0.237
t_y	1.360	0.516	**0.122**	0.156
t_z	8.978	1.139	0.907	**0.713**
t_{avg}	4.123	0.744	0.406	**0.369**

TABLE V
THE REGRESSION ERROR OF RTS ON THE FISH STATUE MODEL.

	SVM		CNN	
Landmark	indirect	direct	indirect	direct
θ_x	1.035	0.703	0.510	**0.209**
θ_y	0.311	0.146	0.073	**0.057**
θ_z	0.561	0.415	0.273	**0.143**
θ_{avg}	0.635	0.421	0.285	**0.136**
t_x	55.663	24.368	**5.916**	6.996
t_y	37.954	24.319	**3.867**	6.803
t_z	234.310	42.035	32.196	**16.690**
t_{avg}	140.514	30.241	13.993	**10.163**

TABLE VI
THE REGRESSION ERROR OF RTS ON THE BOY ANGEL MODEL.

	SVM		CNN	
Landmark	indirect	direct	indirect	direct
θ_x	0.475	0.300	0.186	**0.125**
θ_y	0.374	0.210	0.127	**0.095**
θ_z	0.922	0.910	0.234	**0.201**
θ_{avg}	0.590	0.473	0.182	**0.140**
t_x	46.828	16.157	6.514	**5.769**
t_y	30.575	15.234	**3.986**	4.969
t_z	205.063	35.164	34.995	**17.368**
t_{avg}	94.155	22.185	15.165	**9.369**

D. Visualization

Based on the estimated RTs, we generate the corresponding synthetic images to visually verify the correctness. Due to the space limit, we only randomly select 4 real images of different viewpoints and show their synthetic images in Figure 4. As we can observe, regarding pose estimation, our proposed CNN-indirect and CNN-direct models perform much better than SVM-indirect and SVM-direct, which is consistent with both real image taken on the objects and the synthetic image generated from the CAD models by ground-truth RT. This strongly demonstrates our proposed multi-output CNN regression framework is robust to interpret how cameara looks at objects.

E. Discussion

Regarding that CNN-indirect obtains a little larger regression RT errors than CNN-direct, this can be explained by the fact that the landmark prediction errors have been passed forward and even might be exaggerated during the process of running the POSIT algorithm, while CNN-direct is able to predict RTs directly and avoid the possible errors caused by the POSIT algorithm.

Furthermore, even the ground-truth landmarks may have a little annotation noise, that's why objects in the synthetic

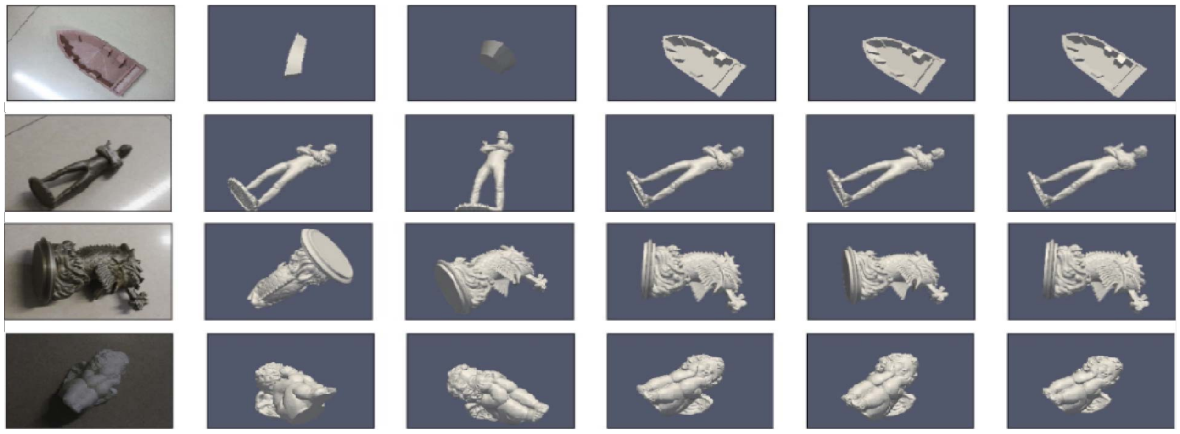


Fig. 4. Visualizations of the RT estimated. At each row, the leftmost column is real image, and the following five synthetic images are generated by the RTs estimated by SVM-indirect, SVM-direct, CNN-indirect and CNN-direct, and the ground-truth RT in order from left to right.

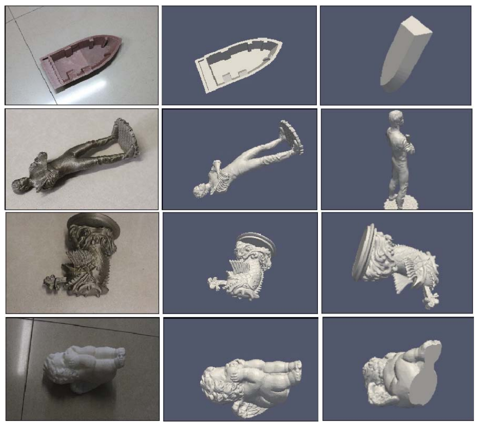


Fig. 5. Failure cases for CNN-direct in the four models. At each row, the images from left to right are real image, synthetic images generated by CNN-indirect's RT, and synthetic image generated by CNN-direct's RT.

images generated by the ground-truth RTs sometimes looks a little different from the corresponding real images, as seen in Figure 4.We also observe some failure cases for CNN-direct, as shown in Figure 5. This is because the space of rotation angle is non-Euclidean, and in some cases the angle representation is ambiguous [11]. For example, $-\pi$ and π are identical to each other in physical space, while the difference value is too large. Instead, the space of M landmarks is Euclidean so that CNN-indirect can avoid the ambiguity issue in the angle representation. One reason to explain the position shift of object in the CNN-indirect's synthetic images is that there are a few number of the specific M landmarks invisible in real images and cannot be predicted correctly in our current CNN regression model, which may affect the precision of estimated translation vector.

V. Conclusion

We propose a CNN-based multi-output regression framework to estimate the camera's RTs directly and indirectly from images. What's more, we are able to generate the synthetic images to visually verify the correctness and interpret how the camera looks at the 3D CAD object effectively and accurately. The experiments conducted on real images taken on four CAD models demonstrate our proposed framework's ability to interpret how the camera looks at one CAD object. Our future work includes investigating and developing more powerful CNN regression models to reduce the regression errors, extending the current setting from a single CAD object to multiple CAD objects.

Acknowledgement

This work was supported by NSFC Grant No. 41601491 and 61602183; National Key Technology R&D Program of China Grant No. 2015BAH28F0103335; State Key Laboratory of Virtual Reality Technology and System, Beihang University Grant No. BUAA-VR-16KF-16; the Science and Technology Project of Guangzhou City (No. 201707010140); and the Fundamental Research Funds for the Central Universities Grant No. 2017QC061 and 2017QC057.

References

[1] $3d^2pm$ - 3d deformable part models. In *ECCV*, October 2012.
[2] D. F. Dementhon and L. S. Davis. Model-based object pose in 25 lines of code. *IJCV*, 1995.
[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
[4] Q. Hu, P. Wang, C. Shen, A. van den Hengel, and F. Porikli. Pushing the limits of deep cnns for pedestrian detection. *TCSVT*, 2017.
[5] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009.
[6] A. Jourabloo and X. Liu. Pose-invariant face alignment via cnn-based dense 3d model fitting. *IJCV*, 2017.
[7] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. *arXiv:1704.00390*, 2017.
[8] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *CVPR*, 2015.
[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*. 2012.
[10] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3d geometry to deformable part models. In *CVPR*, June 2012.
[11] A. Saxena, J. Driemeyer, and A. Y. Ng. Learning 3-d object orientation from images. In *Robotics and Automation*. IEEE, 2009.
[12] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014.
[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
[14] D. Teney and J. H. Piater. Continuous pose estimation in 2d images at instance and category levels. In *The Conference on Computer and Robot Vision*, 2013.
[15] M. Torki and A. Elgammal. Regression from local features for viewpoint and pose estimation. In *ICCV*, 2011.