Viewpoint Estimation for Objects with Convolutional Neural Network Trained on Synthetic Images

Yumeng Wang, Shuyang Li, Mengyao Jia, and Wei Liang(✉)

School of Computer Science, Beijing Institute of Technology, Beijing, China
liangwei@bit.edu.cn

Abstract. In this paper, we propose a method to estimate object viewpoint from a single RGB image and address two problems in estimation: generating training data with viewpoint annotations and extracting powerful features for the estimation. We first collect 1780 high quality 3D CAD object models of 3 categories. Then we generate a synthetic RGB image dataset with viewpoint annotations, in which each image is generated by placing one model in a realistic panorama scene and rendering the model with a random camera parameters. We train a CNN model on our synthetic dataset to predict the object viewpoint. The proposed method is evaluated on PASCAL 3D+ dataset and our synthetic dataset. The experiment results show good performance.

Keywords: Viewpoint estimation · Convolutional neural network · Synthetic image · Panorama scene rendering

1 Introduction

When a person observes objects in a scene, he or she tends to perceive the coarse information, e.g. viewpoints, layout etc., before getting more details [1]. Estimating the viewpoint from a single image is important for the applications of traffic surveillance, robotic perception, object recognition and so on. Take the task of manipulating a cup by a robot as an example, even if a robot localizes the position of the cup in a RGB image, it can't grasp or move the cup without knowing the viewpoint of it.

To estimate the object viewpoint, the Convolutional Neural Network (CNN) model is an optional method which benefits from its ability of handling powerful features. In recent years, CNN method has achieved great success on a variety of computer vision tasks, such as image classification, detection, pose estimation and segmentation. Handling large labeled datasets for training and extracting stronger features helps to outperform state-of-art methods in these topics. In this paper, we apply a CNN model to estimate object viewpoint from a single image.

The obstacle of CNN method application is to get large-scale training data. For the problem of the viewpoint estimation, it is more difficult because annotating viewpoints mannually is time-consuming and inaccuracy. Take the dataset

E. Chen et al. (Eds.): PCM 2016, Part II, LNCS 9917, pp. 169–179, 2016.
DOI: 10.1007/978-3-319-48896-7_17

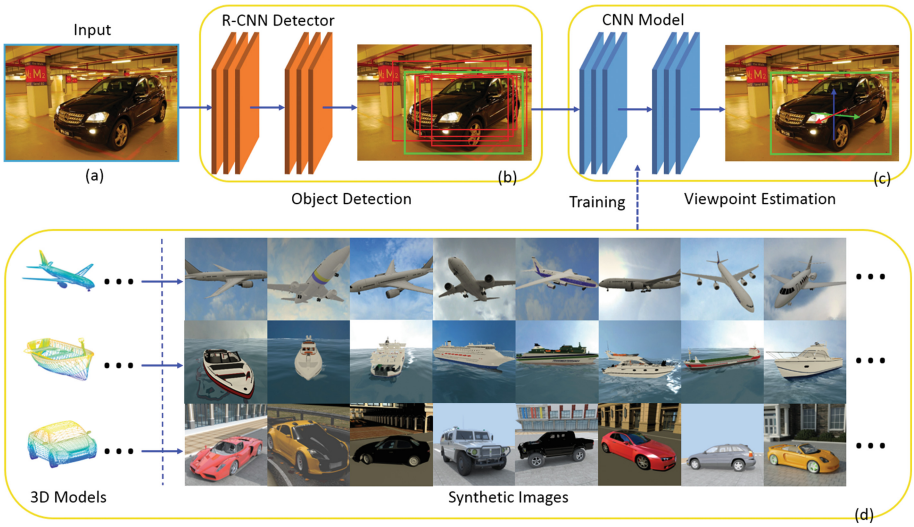

Fig. 1. The framework of our method. (a) Input a single RGB image. (b) A R-CNN detector. (c) The viewpoint estimation by a trained CNN. (d) Some examples of synthetic data for CNN training.

of PASCAL 3D+ [2] as an example, it provides about 22K annotated images for viewpoint estimation, which are far from meeting the requirement of CNN training. In this paper, we design a rendering pipeline which generates realistic RGB images with viewpoint annotations for CNN training.

Figure 1 demonstrates the framework of our method. Given a RGB image in Fig. 1(a), our method runs R-CNN firstly to detect objects in this scene (Fig. 1(b)). Then the viewpoint is estimated by a trained CNN model in Fig. 1(c). We collected three categories of 3D models, comprising aeroplane, boat and car. These models are rendered in 3DS MAX for the CNN model training (Fig. 1(d)).

There are three contributions in this paper:

(i) We design a pipeline to generate realistic synthetic images with annotation in panorama scenes automatically. This method is able to be generalized to other computer vision tasks easily.
(ii) We generated a high resolution synthetic image dataset for viewpoint estimation. The dataset includes 2.76M annotated images of 3 categories totally.
(iii) We train a CNN model to estimate object viewpoints. The results of experiment show a good performance on the benchmark dataset.

2 Related Work

Viewpoint estimation. Recently, computer vision community has been growing interest of the problem of viewpoint estimation for objects [3,4]. Herdtweck

and Curio [5] used Random forest to estimate viewpoint. Fidler et al. [6] proposed a deformable 3D cuboid model for 3D object detection and viewpoint estimation. Some work [7,8] linked parts across views to represent continuous viewpoints. It is notable that CNN models have shown their remarkable performance in the estimation task [9,10].

Syntheitc data methods. In recent years, there is also a growing amount of work which used synthetic images to solve computer vision problems [11–16]. Peng et al. [13] used 3D models to render images for training object detectors. Stark et al. [17] used CAD models for recognizing 3D object. Gupta et al. [14] solved the problem of inferring 3D object pose with synthetic images. Lim et al. [15] used 250 chair models to render tens of thousands training images, which are then used to train deformable part models. In the meantime, Ubry et al. [16] expanded the number of models by one order of magnitude. They collected about 1.3K chair models for building key point correspondences between 2D images and rendered 3D views.

Dataset. ImageNet is a huge image dataset including 14.2M real images in 22K categories for object recognition [18]. Xiang et al. [2] provided a novel and challenging dataset which includes 22K images of 12 categories with 3D annotations. Besides, they also released an annotation tool to get continuous 3D viewpoint annotations. More recently, Su et al. [10] utilized the ShapeNet [19] to generate about 2.4M synthetic images for viewpoint estimation. They consider of adding background image after rendering models and then crop it, which improved the performance of the methods in [15,16]. Because images rendered with a fully transparent background cause the problem of overfitting. In this work, we design a synthetic image generating pipeline to get more realistic images, which renders models in a panoramic scene rather than overlapping the rendered images on a RGB image.

3 Approach

Given a single RGB image, our approach aims to estimate the viewpoint of an object in 3D space. The viewpoint of an object in 3D space is defined as a tuple $\Theta = (\theta_a, \theta_e, \theta_c)$, where $\theta_a, \theta_e, \theta_c$ are the azimuth, elevation and cyclorotation angle respectively. We model the estimation of viewpoint Θ as a regression problem. Compared with classification-based formulations [10,20], regression-based formulation output an exact value which is essential in most situations, e.g. robot grabbing. In this section, we design a CNN model to estimate Θ, which is trained on synthetic images.

3.1 Synthetic Dataset

Annotating training data with accurate viewpoint is impractical, because the process is expensive and involves ambiguous decisions. We propose to generate

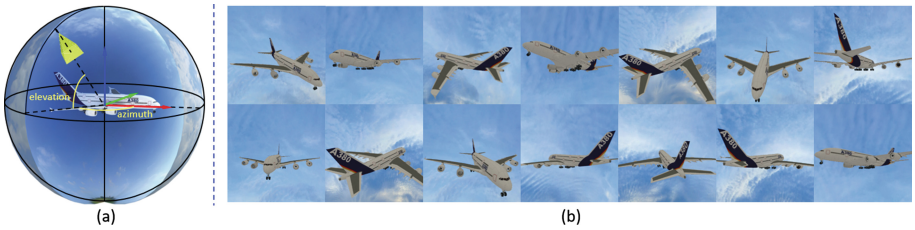

Fig. 2. Rendering synthetic dataset. (a) An illustration of our rendering setting in a panoramic scene. (b) Some rendered images in this scene.

synthetic images by rendering techniques for two reasons: (i) getting accurate ground truth for each viewpoint; (ii) getting much denser data than real data.

Model Collection. We collected 3D models from online repositories such as TurboSquid (about 300K models) and cgtrader (about 250K models). While many of the downloaded models are not in good visual quality, we handpick the data carefully and normalize the center of all 3D models to the origin. As a result, we retain in total of 1780 high-quality 3D models of 3 categories.

Image Rendering. We select an appropriate panoramic scene, and place our model in the center of the scene. Utilizing 3DS MAX Script and VRay renderer, we render the 3D models by adjusting the parameters of the camera automatically. Instead of assuming the distance between camera and model center be fixed, we adjust the camera distance automatically according to the model size. By self-adjusting the camera distance, we obtain a suitable camera view in which the model occupies most of the rendered image.

We also sample the number of light sources, their positions and energies to get close to the real scene. Moreover, we set the azimuth, elevation and cyclorotation angels correspond with the distribution of the statistics of PASCAL 3D+ dataset, rather than randomly sampling the model from all possible views. More details will be discussed in Sect. 4.1.

It is worth noting that, the images rendered in our method is better than the ones rendered in transparent background which is highly contrasted. Our rendering method prevents the CNN model from overfitting by rendering in a panoramic scene rather than overlapping the rendered image on a random RGB image. The result dataset contains a total of $2,670,000$ synthesized images, each annotated with the ID indicating the different style, as well as the viewpoint parameters Θ.

3.2 Building CNN Model

Inspired by the work in [21], we build a CNN model to estimate the viewpoint in this section.

Architecture of the CNN. The architecture of our CNN is demonstrated in Fig. 3. The structure of the network can be described by the size of feature maps

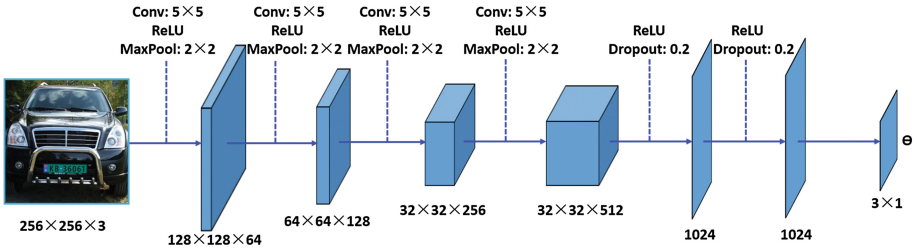

Fig. 3. Architecture of our proposed CNN. The input is a $256 \times 256 \times 3$ image cropped and rescaled from the detection process. The output is the viewpoint Θ.

at each layer as *input* ($256 \times 256 \times 3$) - *conv*1 ($128 \times 128 \times 64$) - *relu*1 - *pool*1 - *conv*2 ($64 \times 64 \times 128$) - *relu*2 - *pool*2 - *conv*3 ($32 \times 32 \times 256$) - *relu*3 - *pool*3 - *conv*4 ($32 \times 32 \times 512$) - *relu*4 - *pool*4 - *fc*5 (1024) - *relu*5 - *dropout*5 - *fc*6 (1024) - *relu*6 - *dropout*6 - *output* (3×1).

In details, the input of the network is a cropped single RGB image and rescaled from detection process with a fixed size of 256×256 pixels. The output is the estimated values of viewpoint Θ. Our CNN consists of four convolutional layers to extract features and two fully connected layers to estimate viewpoint. For each convolutional layer, the size of the kernel is 5×5 pixels and its stride is 2 pixel. We then apply max-pooling operations after convolutional layers to increase the performance despite the reduction of resolution. The stride for pooling is 2 and we set the pooling operator size as 3×3. For feature extraction stage, the numbers of features in these four convolutional layers are 64, 128, 256 and 512 respectively. Following the four convolutional layers, there are two fully connected layers with 1024 hidden units. Instead of traditional sigmoid neurons, we use Rectified Linear Units (ReLUs) [22] in the two fully-connected layers to accelerate the stage of training without making a significant difference to generalisation accuracy. To reduce overfitting in training the neural network, we bring in a regularization method called dropout, which has been proven to be very efficient in [23]. At last, in the output layer, there is a linear activation function to predict the parameters of viewpoint Θ.

Loss Function and Training. Assume that $D = \{(I_n, \Theta_n) | n \in [1, N]\}$ means the training set, where I_n is the synthesis image, Θ_n is the viewpoint parameters and N is the amount of train images. Then the learning task is to minimize our defined loss function $L(w)$:

$$L(w) = \frac{1}{N} \sum_{n=1}^{N} f(I_n; w) + \lambda r(w)$$
$$w^* = \arg\min_{w} L(w) \tag{1}$$

where $r(w)$ is a regularization term which penalizes large weights to improve generalization. The parameter λ represents the weight decay that determines

how you trade off between Euclidean loss and large weights. $f(I_n; w)$ is the loss term computed as the output of layers by the given w:

$$f(I_n; w) = \frac{1}{2}||\psi(I_n; w) - \Theta_n||_2^2 \tag{2}$$

Similar with the work in [23], we train our model by stochastic gradient descent (SGD) with momentum and update weight decay in the following form:

$$\begin{aligned} w_t &= w_{t-1} + V_t \\ V_t &= r_t V_{t-1} - \alpha \nabla L(w_t) \\ r_t &= \begin{cases} \frac{t}{T} r_e + (1 - \frac{t}{T}) r_s, & t < T \\ r_t, & t \geqslant T \end{cases} \end{aligned} \tag{3}$$

where w_t and V_t are weight and momentum variable respectively at epoch t, α is learning rate, r_t is momentum coefficient, r_s is starting momentums and r_e is ending momentums, T is a threshold to control how the momentum changes with the number of epochs.

4 Experiments

In this section, we demonstrate the results of our method for viewpoint estimation. We evaluate our viewpoint estimation system on both the PASCAL 3D+ dataset [2] and our synthetic dataset.

4.1 Dataset

Our dataset includes 2,670,000 synthetic images rendered from 1,780 high quality CAD models in 3 types: aeroplane, boat and car. In details, we collected 580 aeroplanes, 324 boats and 876 cars from web. Each model was rendered in total of 1500 times in specified camera parameters in different panoramic scenes.

The rendering parameters of viewpoints follow the statistical distribution of PASCAL 3D+ dataset. We use polar histogram to illustrate the distribution of azimuth angles of azimuth angles for both PASCAL 3D+ dataset and our rendered dataset in Fig. 4, where 0° corresponds to the front view of a model. The distribution of PASCAL 3D+ dataset is depicted by blue lines, whereas ours is depicted by red lines. Figure 4(b) denotes the distribution of elevation angles where 0° corresponds to the horizontal view of the model. Solid bars and shadowed bars represent PASCAL 3D+ dataset and our dataset respectively. Besides, bars in different colors mean different object category. As expected, the distribution of viewpoints is biased. For example, there is a high bias towards the front view (azimuth $\in [0°, 60°] \cup [300°, 360°]$) of car, and the elevation angles are most aggregated in $[-5°, 15°]$. These statisticses show that our dataset has a fairly good distribution in viewpoint variation to characterize the real scene.

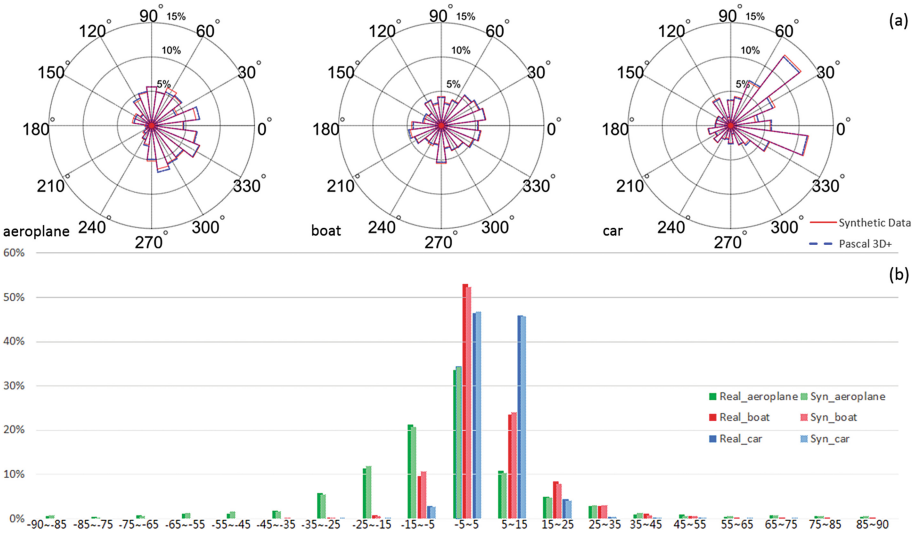

Fig. 4. Distribution of the data. The distribution of azimuths is displayed by three polar histograms in (a), blue lines represent PASCAL 3D+ dataset and red lines represent our dataset. (b) shows the elevation distribution histogram. Solid bars represent PASCAL 3D+ dataset and shadowed bars represent our dataset. Different color means different object category. (Color figure online)

4.2 Viewpoint Estimation

The detection results influence the viewpoint estimation. We consider the accuracy of viewpoint estimation in different detection settings in Fig. 5. In particular, the X-axis is the overlap ratio and the Y-axis is the accuracy. The overlap ratio O is the proportion of how much the detected bounding box overlapping the ground truth. We define an angel range δ. If the predicted value is within

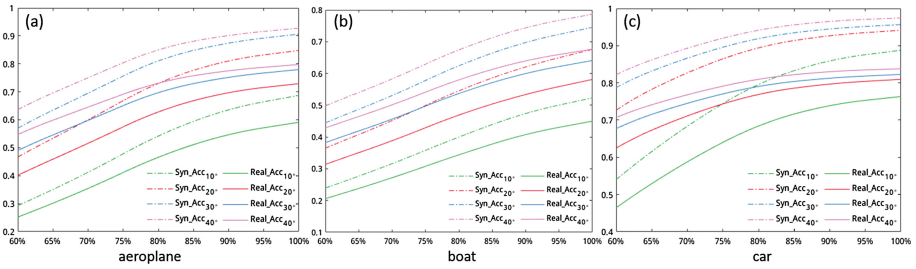

Fig. 5. Viewpoint estimation performance. The accuracy Acc_δ is a function of overlap ratio O and angel tolerance δ. The lines in different colors depict different angel tolerances. The solid lines and the dashed lines represent the accuracy on PASCAL 3D+ dataset and on synthetic dataset respectively. (Color figure online)

Table 1. Performance comparison. Model from Tulsiani [4] based on TNet was trained on VOC 12 train and ImageNet. Su [10] trained his model by synthetic images which were rendered from ShapeNet [19]. Ours used rendered panorama images for training.

Method	aero	boat	car	mean
Acc_{30° (Tulsiani et al.)	0.78	0.49	**0.90**	0.72
Acc_{30° (Su et al.)	0.74	0.52	0.88	0.71
Acc_{30° (Ours)	**0.79**	**0.64**	0.82	**0.75**

the range of $[G - \delta, G + \delta]$, where G is the ground truth, then we consider it as correct. The accuracy subjects to δ is represented by Acc_δ. The change of accuracy with different δ is shown in Fig. 5. The solid lines depict the accuracy on PASCAL 3D+ dataset and the dashed lines depict the accuracy on synthetic test set. The lines in different colors represent different error thresholds. For example, the green one means the predicted viewpoint error within 10°.

Fig. 6. The results of viewpoint estimation. The detected object is bounded by green boxes. We transform the object coordinate into the camera coordinate by the estimated parameters θ_a, θ_e, θ_c. The three colors of red, green, and blue represent three axes of objects in the camera coordinate. (Color figure online)

When the angel tolerance is fixed, the higher 2D detection performance (i.e. overlap ratio) of R-CNN is, the better accuracy we get. The performance on the synthetic test set is better than PASCAL 3D+, because the synthetic test images are more similar with the training data. Comparing with the results of synthetic test set in Fig. 5, when the overlap is 100% and angle range is 40°, the accuracy of car category on synthetic data is around 0.96 while the boat is just 0.79. We believe that the good performance of car category is because cars have similar shapes and more appearance features, such as headlights, automotive glass and so on. Meantime, we can see that the change of green lines in Fig. 5(c) is not as gentle as others, because global appearance features are necessary for cars. It's insufficient to estimate viewpoint if we just own local parts of cars.

Although we train the CNN model on pure synthetic data, our method achieves good performance on real data since the synthetic data provides enough features to estimate the viewpoint. The performance of our method and the comparative methods are shown in Table 1. We compare our method with two recent state-of-the-art work as Tulsiani [4] and Su [10]. They used similar network architectures (TNet/AlexNet) while their loss layer is different. From the comparison results, it is clear that our method outperforms the existing methods.

5 Conclusions

In this paper, we propose a pipeline to generate synthetic panorama images with annotations and design a CNN model trained on synthetic images to estimate object viewpoint. Our pipeline overcomes the limitation of lacking large-scale real images with annotations in CNN training. Our dataset includes in total of 2,760,000 images rendered from 1,780 models. It's worth mentioning that our synthetic images are rendered in panoramic scene using Vray renderer, they are more realistic and have high resolution. We evaluated our method on both synthetic and real data. Our method shows good performance on three object classes form both synthetic and real dataset and outperforms existing methods with the mean accuracy of 0.75.

References

1. Navon, D.: Forest before trees: the precedence of global features in visual perception. Cogn. Psychol. **9**(3), 353–383 (1977)
2. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond PASCAL: a benchmark for 3d object detection in the wild. In: 2014 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 75–82. IEEE (2014)
3. Gu, C., Ren, X.: Discriminative mixture-of-templates for viewpoint classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6315, pp. 408–421. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15555-0_30
4. Tulsiani, S., Malik, J.: Viewpoints and keypoints. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1510–1519. IEEE (2015)

5. Herdtweck, C., Curio, C.: Monocular car viewpoint estimation with circular regression forests. In: 2013 IEEE Intelligent Vehicles Symposium (IV), pp. 403–410. IEEE (2013)
6. Fidler, S., Dickinson, S., Urtasun, R.: 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In: Advances in Neural Information Processing Systems, pp. 611–619 (2012)
7. Payet, N., Todorovic, S.: From contours to 3d object detection and pose estimation. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 983–990. IEEE (2011)
8. Su, H., Sun, M., Fei-Fei, L., Savarese, S.: Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 213–220. IEEE (2009)
9. Mottaghi, R., Xiang, Y., Savarese, S.: A coarse-to-fine model for 3d pose estimation and sub-category recognition. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 418–426. IEEE (2015)
10. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: viewpoint estimation in images using cnns trained with rendered 3d model views. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2686–2694 (2015)
11. Oliver, N.M., Rosario, B., Pentland, A.P.: A Bayesian computer vision system for modeling human interactions. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 831–843 (2000)
12. Nevatia, R., Binford, T.O.: Description and recognition of curved objects. Artif. Intell. **8**(1), 77–98 (1977)
13. Peng, X., Sun, B., Ali, K., Saenko, K.: Exploring invariances in deep convolutional neural networks using synthetic images. CoRR, abs/1412.7122, vol. 2 (2014)
14. Gupta, S., Arbeláez, P., Girshick, R., Malik, J.: Inferring 3d object pose in RGB-D images. arXiv preprint arXiv:1502.04652 (2015)
15. Lim, J.J., Khosla, A., Torralba, A.: FPM: fine pose parts-based model with 3D CAD models. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 478–493. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10599-4_31
16. Aubry, M., Maturana, D., Efros, A., Russell, B., Sivic, J.: Seeing 3d chairs: exemplar part-based 2d–3d alignment using a large dataset of cad models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3762–3769 (2014)
17. Stark, M., Goesele, M., Schiele, B.: Back to the future: learning shape models from 3d cad data. In: BMVC, vol. 2, p. 5 (2010)
18. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 248–255. IEEE (2009)
19. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: an information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
20. Juranek, R., Herout, A., Dubska, M., Zemcik, P.: Real-time pose estimation piggybacked on object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2381–2389 (2015)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp. 1097–1105 (2012)

22. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814 (2010)
23. Zhang, X., Sugano, Y., Fritz, M., Bulling, A.: Appearance-based gaze estimation in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4511–4520 (2015)