3D HEAD POSE ESTIMATION WITH CONVOLUTIONAL NEURAL NETWORK TRAINED ON SYNTHETIC IMAGES

Xiabing Liu, *Wei Liang*,
Yumeng Wang, *Shuyang Li*, *Mingtao Pei*
School of Computer Science, Beijing Institute of Technology, Beijing 100081, P.R. China
{liuxiabing, liangwei, wangyumeng, lishuyang, peimt}@bit.edu.cn

ABSTRACT

In this paper, we propose a method to estimate head pose with convolutional neural network, which is trained on synthetic head images. We formulate head pose estimation as a regression problem. A convolutional neural network is trained to learn head features and solve the regression problem. To provide annotated head poses in the training process, we generate a realistic head pose dataset by rendering techniques, in which we consider the variation of gender, age, race and expression. Our dataset includes 74000 head poses rendered from 37 head models. For each head pose, RGB image and annotated pose parameters are given. We evaluate our method on both synthetic and real data. The experiments show that our method improves the accuracy of head pose estimation.

Index Terms— head pose estimation, convolutional neural network, synthetic images

1. INTRODUCTION

Head pose provides strong cues for human intention, motivation, and attention. Many applications rely on robust head pose estimation results, such as human behavior analysis, human computer interaction, gaze estimation etc.. In the field of computer vision, a head pose is typically interpreted as an orientation of a person's head, which is represented by three angles: pitch, yaw and roll. The task of head pose estimation is challenging because of the large head appearance variation (expression, race and gender) and environmental factors (occlusion, noise and illumination). The techniques of head pose estimation are well summarized in [1].

Recently, convolutional neural network (CNN) method has achieved great success in many computer vision tasks. It can handle large training data and perform well. While getting a large amount of annotated data for the CNN model training is difficult for most tasks. Applying CNN method to head pose estimation has the same obstacle. In this paper, we design a pipeline to generate annotated synthetic head images and apply a CNN model to estimate head poses, which is trained on synthetic head images. Our method has the similar pipeline with the work of [2, 3]. The framework of our method is shown in Fig. 1. Firstly, we collected 37 high resolution 3D head models, in which we consider gender, age, race and expression (left panel in Fig.1 (a)). Then, we generate a head pose dataset by rendering technique in 3DS Max, which consists of synthetic RGB images and annotated parameters (right panel in Fig.1 (a)). Finally, a CNN model is trained on the synthetic images to learn the mapping from the head images to head poses (Fig.1 (b)).

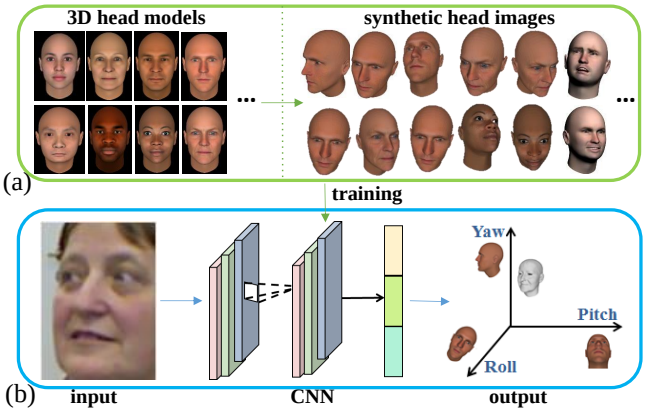

Fig. 1. The framework of our method. (a) The synthetic head poses data for CNN training are generated from head models. (b) For a given head image, a CNN model is used to estimate the head pose, which is represented by three angles: yaw, pitch and roll.

There are three contributions in this paper:

i) We design a rendering pipeline to generate synthetic head images. In 3DS Max, by sampling in the parametric space, we render high resolution and realistic RGB head images with annotated ground truth. This method can be generalized to other tasks.

ii) We generated a high resolution head pose dataset. In this dataset, we consider gender, age, race and expression, which includes 74000 annotated head images.

iii) We train a convolutional neural network to estimate head poses in real time.

This work is supported in part by the Natural Science Foundation of China (NSFC) under the Grant No.61472038 and No.61375044.

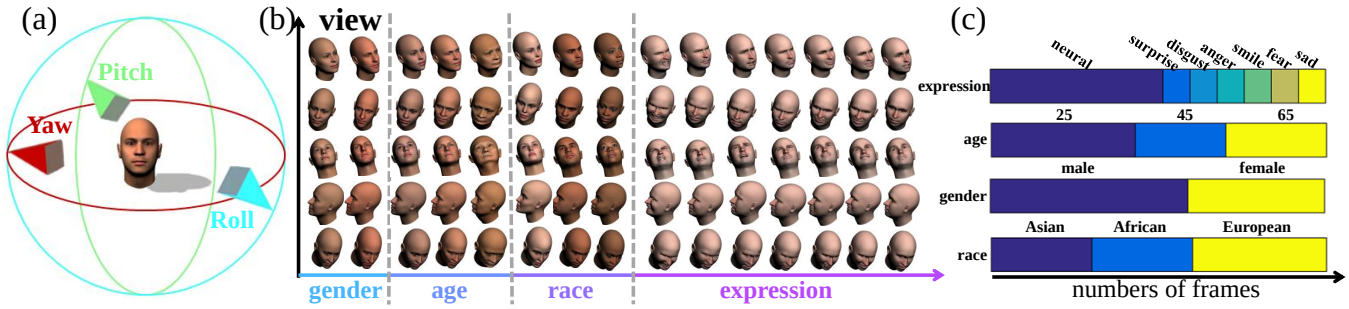

Fig. 2. The synthetic head pose dataset. (a) A head model is placed on a virtual ground. The camera is moved randomly on the surface of a sphere whose center is the same as the head model's center. (b) Some examples in our synthetic dataset. Each row demonstrates a head pose. Each column demonstrates a different style of our models. We consider gender, age, race and expression in our models. (c) The statistics of our dataset. The bar represents the percentage of one situation.

2. RELATED WORK

Random forest was used to solve head pose estimation problem because of the ability of handling large training dataset [4, 5, 6]. These methods could estimate head poses in real-time without specific hardwares like GPU. Besides, many different methods have been proposed. The approaches in [7, 8] sought low-dimensional manifolds to model the continuous variation in head poses. Geng et al. [9] associated a soft pose label, called multivariate label distribution (MLD) to each image, and minimized the weighted Jeffrey's divergence between the predicted MLD and the ground truth MLD. Martin et al. [10] combined head features and head model generations to build a detector which provided accurate results in a wide pose range. The work of Ahn et al. [11] used a CNN method to learn a mapping function between visual appearance and head pose angles. Ghiass et al. [12] performed pose estimation by fitting a 3D morphable model which included pose parameters.

Recent years, CNN method shows good performance in many computer vision tasks, such as classification [13, 14], object detection [15], face retrieval [16] and so on. It is also used in face related vision tasks, like pose estimation. Li et al. [17] proposed a cascade CNN which outperformed the state-of-the-art methods. Zhang et al. [18] used multimodal CNN to learn a mapping from head poses and eye images to gaze directions. Fan et al. [19] first achieved detection and localization on joint using CNN method, then combined these results from all image patches to estimate human pose.

3. METHOD

Given a RGB image of a head, our goal is to estimate its 3D pose. A head pose in 3D space is defined as $\Theta = (\theta_p, \theta_y, \theta_r)$, where θ_p, θ_y and θ_r represent three rotation angles of pitch, yaw and roll respectively. In this section, we model the estimation of head pose Θ as a regression problem. Then, we design a convolutional neural network to estimate Θ, which is trained on synthetic head images.

3.1. Synthetic Head Pose Dataset

In practice, the ground truth of head poses is difficult to measure. We propose to generate synthetic head poses by rendering techniques. There are three advantages: i) It provides accurate ground truth for each pose; ii) The precision of head poses can be controlled. For example, we can render every 1 or 0.1 degree; iii) Synthetic data is much denser than real data.

We utilize 3DS Max Script and VRAY renderer to render head models automatically. A head model is placed on a virtual ground. We move the camera randomly on the surface of a sphere whose center is the same as the head model's center. We get head images through the camera's view. It is equivalent to the head pose rotation along three axis.

Model. We collected 37 high quality 3D head models with texture. There are 15 females and 22 males in these models. Each model has more than 6500 vertices and 12000 faces. The age of these models is 25, 45 and 65. The race includes Europeans, Africans and Asians. We also consider seven kinds of expression: neural, anger, disgust, fear, sad, smile and surprise, which are not considered in the existing head pose dataset.

Parameters. According to human head motion constraints, we set different angle range for each rotation axis: $\theta_y \in [-75°, 75°]$, $\theta_p \in [-60°, 60°]$, and $\theta_r \in [-50°, 50°]$.

In order to get different head pose images, we draw three random numbers from the standard uniform distribution on interval (0,1). Each number is scaled to the corresponding range of each rotation axis. Then, the angle of each rotation axis is transformed to the angle of the camera's view. We get head images in each direction through the camera's view. Some examples and the statistics of our dataset are shown in Fig.2.

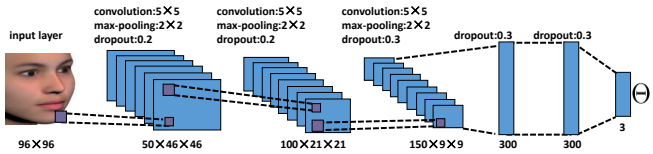

Fig. 3. Architecture of the proposed CNN. The input is a 96×96 pixels head image. The output is a head pose vector Θ, which consists of yaw, pitch, roll.

3.2. Architecture of the CNN

We design a CNN to estimate head pose. The architecture of our CNN is demonstrated in Fig. 3. The input of the network is a head RGB image with a fixed size of 96×96 pixels. The output of the network is a 3D head pose Θ. The CNN consists of 3 convolutional layers to extract features and 3 max pooling layers to increase performance despite the reduction of resolution. For each convolutional layer, there is a kernel with the size of 5×5 pixels and a stride of 1 pixel. Max pooling layer has a kernel with the size of 2×2 pixels. The numbers of features in these three convolutional layers are 50, 100 and 150 respectively. Following the three convolutional layers, there are two fully connected layers with 300 hidden units.

To reduce overfitting, we employ a regularization method called "dropout" which is proved to be very efficient [13]. In addition, both the convolutional layers and the fully connected layers consist of a linear transformation followed by a non-linear one. In the output layer, there is a linear activation function to predict the parameters of head poses Θ.

3.3. Training

Assume that $D = \{(I_i, \Theta_i)|i \in [1, N]\}$ is the training set, where N is the number of training data. The learning task is to minimize the loss function $L(w)$:

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

$r(w)$ is a regularization term which penalizes large weights to improve generalization. The parameter λ represents the weight decay. It 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 weight w and head image I_n:

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

where $\psi(I_n; w)$ is the predicted result of I_n with the network weight w.

Similar with the work in [13], we train our model by stochastic gradient descent with momentum and update weights by the following form:

$$w_{t+1} = w_t + V_{t+1} \tag{3}$$
$$V_{t+1} = \mu V_t - \alpha \nabla L(w_t),$$

where w_t and V_t are weight and momentum variable at epoch t respectively, α is learning rate and μ is momentum coefficient.

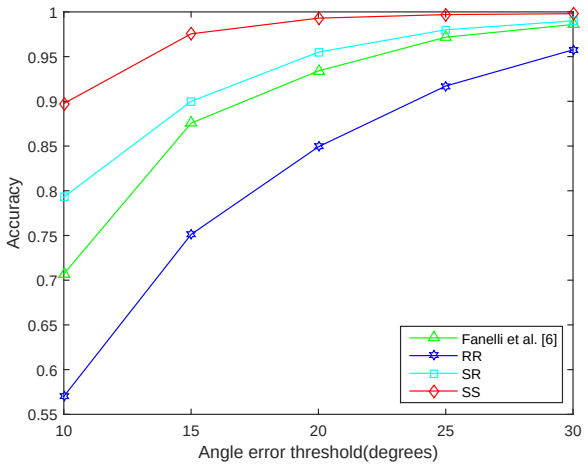

Fig. 4. Accuracy of head pose estimation for different methods. Fanelli et al. [6], SR and RR are tested on the Biwi dataset. SS are tested on synthetic RGB images.

4. EXPERIMENTS

4.1. Settings

Dataset. We evaluate our method on two kinds of data: synthetic data from our dataset and real data from the Biwi Kinect Head Pose Dataset [6].

Synthetic dataset: There are 74000 images of 37 models in our synthetic dataset. The dataset consists of 37 sequences. Each sequence includes 2000 frames. The resolution of the RGB images is 640×480 pixels. The head pose range covers ±75° of yaw, ±60° of pitch and ±50° of roll.

Real image dataset: The Biwi Kinect Head Pose Dataset includes 15678 images of 20 people (6 females and 14 males). The dataset consists of 24 sequences. There are about 600 frames in each sequence. The RGB images have a resolution of 640×480 pixels, and a head typically consists of around 90×110 pixels. The head pose range covers ±75° of yaw, ±60° of pitch and ±50° of roll.

Implementation Details. We use the open-source CNN library Caffe [20] to implement our CNN. We initialize all weights in each convolutional layer by a zero-mean Gaussian distribution with standard deviation 0.01. In each fully-connected layer, we use XavierFiller [20] to initialize the weights. The neuron biases in each convolutional layer and

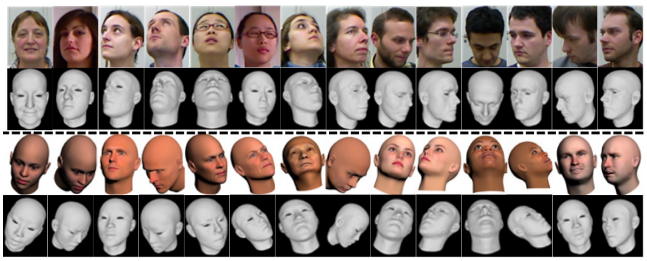

Fig. 5. Some head pose estimation results. The first and the third row are the input from the Biwi Kinect Head Pose Dataset and our synthetic dataset respectively. We visualized the estimation results in the second and the fourth row by rotating a head model. The rotation parameters are the same as the predicted values from our CNN.

fully-connected layer are set to the constant 1 and the remainings are set to 0. The stride is initialized as 1 in convolutional layers and 2 in the pooling layers.

We train our model using stochastic gradient descent with a batch size of 96, momentum of 0.9, and weight decay of 0.0005. The learning rate is initialized to 0.01 and decreases by a factor of ten after every certain number of iterations.

4.2. Experimental Results

Generally, there are two strategies to split the training and testing dataset [6, 11]. The difference is that whether the training process has seen all the people. The first strategy is to mix all data together and split them into training and testing set randomly [11]. The second strategy is to choose some sequences (including some people) as training set and the remaining sequences as testing set (the other people) [6]. For the second strategy, in the testing process, there are some person who never appeared in the training set before. It is more challenging. We use the second strategy to evaluate our method.

There are two kinds of data in our experiments: synthetic RGB data (S) and real RGB data (R). We use combination of two letters to represent the type of training data and testing data. We evaluate our method in three data settings:

SS: The training data is synthetic RGB data and the testing data is synthetic RGB data as well. We randomly choose 30 sequences as training set and the remaining 7 sequences are testing set.

SR: The training data is synthetic RGB data and the testing data is real RGB data. We use the same training set as SS uses. All the 24 sequences in the Biwi dataset are used to test.

RR: The training data is real RGB data and the testing data is real RGB data as well. We randomly choose 22 sequences as training set from all the sequences of the Biwi dataset. The remaining 2 sequences are used as testing set.

In Fig.4, we present our estimation accuracy changes with different error threshold settings. The threshold measures the large deviation in estimation. For example, the minimum threshold is 10 degrees. With this threshold, the error within 10 degrees is considered as correct. We evaluate the experiment results by increasing 5 degrees each time. We can see that the proposed method performs well compared with state-of-the-art method on the same dataset. When the threshold is 10 degrees, our results of SR get an accuracy of 79.3%, whereas the Fanelli et al. [6] achieves only 70.5%. For a success threshold of 20 degrees, our method achieves 95.5% compared to 93.4% of Fanelli et al. [6]. Through observing the results of RR and SR, we can find that SR has a better performance. While SR uses a bigger training set compared with RR, the observation confirms that large scale synthetic RGB data is useful in the training of the CNN. Table 1 shows the mean and standard deviation of yaw, pitch and roll errors together. Noted that Drouard et al. [21] use color information. Fanelli et al. [6] use depth information and Wang et al. [22] use both of them. The table presents that the results of SR is better than any one of them. More qualitative examples are shown in Fig. 5.

Method	Yaw(°)	Pitch(°)	Roll(°)
Wang et al. [22]	8.8±14.3	8.5±11.1	7.4±10.8
Fanelli et al. [6]	6.6±12.6	5.2±7.7	6.0±7.1
Drouard et al. [21]	4.9±4.1	5.9±4.8	4.7±4.6
RR	6.1±5.2	6.0±5.8	5.7±7.3
SR	4.5±3.8	4.3±2.7	2.4±1.9
SS	2.7±2.4	3.4±3.1	2.2±2.3

Table 1. The estimation errors of each rotation angle. In each cell, the first number is the mean of errors and the second one is the standard deviation of errors.

Experiments show that testing a frame in our model only needs 0.76 ms on a PC with 3.3GHz CPU and GTX TITAN Black GPU. Therefore our approach can achieve real time processing with the help of GPU.

5. CONCLUSION

In this paper, we design a convolutional neural network to estimate head poses which is trained on synthetic RGB images. To provide annotated head poses in the training process, we generate a realistic head pose dataset by rendering techniques, which considers the variation of gender, age, race and expression. Our dataset includes 74000 head poses rendered from 37 head models. We evaluate our method on both synthetic and real data. The experiments show that our method improves the accuracy of head pose estimation. Benefit from the use of GPU, real time processing can be achieved while maintaining performance.

6. REFERENCES

[1] Erik Murphy-Chutorian and Mohan Manubhai Trivedi, "Head pose estimation in computer vision: A survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 4, pp. 607–626, 2009.

[2] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox, "Learning to generate chairs with convolutional neural networks," in *CVPR*, 2015, pp. 1538–1546.

[3] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox, "Flownet: Learning optical flow with convolutional networks," *arXiv:1504.06852*, 2015.

[4] Gabriele Fanelli, Juergen Gall, and Luc Van Gool, "Real time head pose estimation with random regression forests," in *CVPR*. IEEE, 2011, pp. 617–624.

[5] Gabriele Fanelli, Thibaut Weise, Juergen Gall, and Luc Van Gool, "Real time head pose estimation from consumer depth cameras," in *Pattern Recognition*, pp. 101–110. Springer, 2011.

[6] Gabriele Fanelli, Matthias Dantone, Juergen Gall, Andrea Fossati, and Luc Van Gool, "Random forests for real time 3d face analysis," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 437–458, 2013.

[7] Chiraz BenAbdelkader, "Robust head pose estimation using supervised manifold learning," in *Computer Vision–ECCV*, pp. 518–531. Springer, 2010.

[8] Jiwen Lu and Yap-Peng Tan, "Ordinary preserving manifold analysis for human age and head pose estimation," *Human-Machine Systems, IEEE Transactions on*, vol. 43, no. 2, pp. 249–258, 2013.

[9] Xin Geng and Yu Xia, "Head pose estimation based on multivariate label distribution," in *CVPR*. IEEE, 2014, pp. 1837–1842.

[10] Manuel Martin, Florian Van De Camp, and Rainer Stiefelhagen, "Real time head model creation and head pose estimation on consumer depth cameras," in *3D Vision (3DV), 2014 2nd International Conference on*. IEEE, 2014, vol. 1, pp. 641–648.

[11] Byungtae Ahn, Jaesik Park, and In So Kweon, "Real-time head orientation from a monocular camera using deep neural network," in *Computer Vision–ACCV 2014*, pp. 82–96. Springer, 2015.

[12] Reza Shoja Ghiass, Ognjen Arandjelovic, Denis Laurendeau, and St Andrews, "Highly accurate and fully automatic head pose estimation from a low quality consumer-level rgb-d sensor," in *Proceedings of the 2nd Workshop on Computational Models of Social Interactions: Human-Computer-Media Communication*. ACM, 2015, pp. 25–34.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] Zhen Dong, Yuwei Wu, Mingtao Pei, and Yunde Jia, "Vehicle type classification using a semisupervised convolutional neural network," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 16, no. 4, pp. 2247–2256, 2015.

[15] Wanli Ouyang, Xiaogang Wang, Xingyu Zeng, Shi Qiu, Ping Luo, Yonglong Tian, Hongsheng Li, Shuo Yang, Zhe Wang, Chen-Change Loy, et al., "Deepid-net: Deformable deep convolutional neural networks for object detection," in *CVPR*, 2015, pp. 2403–2412.

[16] Zhen Dong, Su Jia, Tianfu Wu, and Mingtao Pei, "Face video retrieval via deep learning of binary hash representations," in *AAAI*, 2016.

[17] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua, "A convolutional neural network cascade for face detection," in *CVPR*, 2015, pp. 5325–5334.

[18] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling, "Appearance-based gaze estimation in the wild," *arXiv:1504.02863*, 2015.

[19] Xiaochuan Fan, Kang Zheng, Yuewei Lin, and Song Wang, "Combining local appearance and holistic view: Dual-source deep neural networks for human pose estimation," *arXiv:1504.07159*, 2015.

[20] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014, pp. 675–678.

[21] Vincent Drouard, Sileye Ba, Georgios Evangelidis, Antoine Deleforge, and Radu Horaud, "Head pose estimation via probabilistic high-dimensional regression," in *ICIP*. IEEE, 2015, pp. 4624–4628.

[22] Bingjie Wang, Wei Liang, Yucheng Wang, and Yan Liang, "Head pose estimation with combined 2d sift and 3d hog features," in *ICIG*. IEEE, 2013, pp. 650–655.