Vol.2011-CVIM-178 No.15
2011/9/5

環境への自動適応を伴うアピアランスベース頭部姿勢推定

ISARUN CHAMVEHA,[†1] YUSUKE SUGANO,[†1]
DAISUKE SUGIMURA,[†1] TEERA SIRITEERAKUL,[†1]
TAKAHIRO OKABE,[†1] YOICHI SATO[†1]
and AKIHIRO SUGIMOTO [†2]

本稿では、アピアランスベース頭部姿勢推定における、推定対象環境への自動適応手法を提案する。アピアランスベースの推定手法は、ラベル付けされた正解学習データを推定器の学習時に必要とする。頭部領域の見えは撮影環境によって大きく変化するため、この学習データは推定対象となるテストデータと同じ環境で獲得されることが望ましいが、全ての設置環境で正解学習データを集めることは現実的には不可能な場合が多い。提案手法ではこの問題を解決するために、テスト映像中の歩行者の追跡結果を利用する。人物が進行方向を向いているという仮定のもとで姿勢ラベル付きの頭部画像を獲得し、頭部姿勢クラスの分類器を学習することで、自動的に対象環境に適した頭部姿勢推定器を構築することが可能になる。複数の分類アルゴリズムを使った実験を通して、テスト環境とは異なる環境で獲得された学習データを用いる場合に比べて提案手法の分類精度が高くなることを示す。

1. Introduction

Head pose can be an important factor in inferring the focus of attention of humans, and thus can be used in a wide range of applications. For this reason, techniques for estimating head pose have been considered an important research task for decades.

Although various image-based approaches have been proposed for estimating head pose (see[15] for a recent survey), one of the major remaining technical challenges is to deal with low resolution images. In some application scenarios like visual surveillance, it is often the case that head regions in input images are quite small. Small images contain limited information, thus it is still a challenging task to achieve accurate estimation results in such cases.

Recently it has become well known that the use of appearance-based approaches is a promising way to estimate head poses from low resolution images. Compared with model-based methods like active appearance models[9),14)] which rely on geometric facial models, appearance-based methods directly treat image features and are known to work even with low resolution images.

Usually, appearance-based estimation is carried out by learning a mapping function from the image space to the pose space. There are many approaches to establish the mapping and they use various techniques like classification[5),16),18)], regression[20),21)] and manifold embedding[2),3),23)]. For example, Robertson *et al*.[18)] used skin color as a descriptor and a binary tree algorithm to establish a head pose classifier. Benfold *et al*.[5)] proposed a descriptor which learns a model of skin color automatically and used a randomized fern algorithm for head pose classification. Orozco *et al*.[16)] proposed an image descriptor which does not require explicit segmentation of skin and hair pixels by using similarity distance maps with class-mean appearance templates, and used the descriptor with a multi-class SVM (support vector machine) for head pose classification. Their work has been applied to surveillance videos, and it is shown that head poses can be estimated even from low-resolution head images.

However, current appearance-based methods suffer from one important problem when they are used in realistic scenarios. That is, a large number of training images with ground truth labels, *i.e*., correct head orientations, are needed. For instance, Orozco *et al*.[16)] and Robertson *et al*.[18),19)] used 100 images for each head pose class as training data. To obtain ground truth labels, training data need to be manually labeled or an intrusive device is needed to record head pose directions. More importantly, head appearances can change significantly from scene to scene, and according to camera properties even in the same scene. Accordingly, head pose estimators work best if trained with data from the same camera and setting. However, it is prohibitively expensive to collect ground truth data manually every time a head pose estimation method is applied to different scenes.

To overcome the problem, we propose an appearance-based head pose estimation method that automatically collects training dataset from test scenes. Based on the observation that, most of the time, people turn their heads towards where they are walking, our method aims to collect head images of walking pedestrians as the training dataset. Pedestrians in the input image sequence are tracked first to achieve their head images and walking directions. After rejecting outliers which are facing different directions,

†1 Institute of Industrial Science, The University of Tokyo
†2 National Institute of Informatics

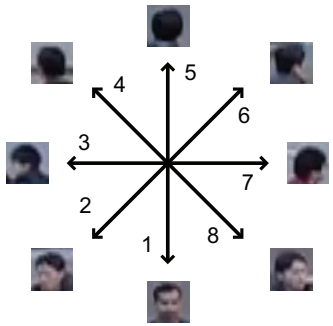


Fig. 1 Head pose definitions. Head poses are divided into 8 discrete classes.

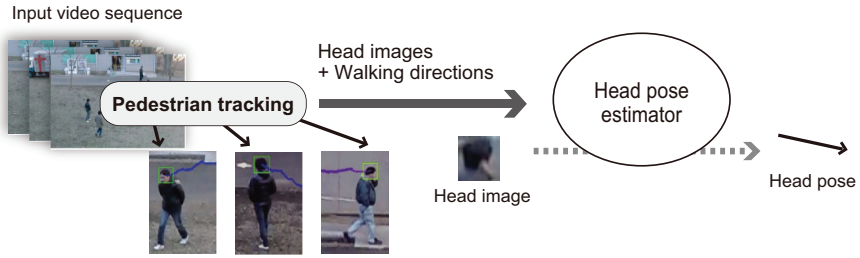


Fig. 2 Proposed framework. Given an input video sequence, our method first track pedestrians in the video and obtain their head images and direction they are walking. By using the walking directions as a cue to infer head pose directions, out method constructs an appearance-based head pose estimator.

their walking directions are used as ground truth labels of their head orientations. In this way, our method does not require a tedious and time-consuming task of collecting a large amount of ground truth data.

2. Scene-Specific Adaptation for Appearance-Based Head Pose Estimation

The appearance-based head pose estimation is a task to determine head pose p from feature vector $\boldsymbol{h}$ of head images. In the case of classification, p is defined as a discrete direction in image space as illustrated in Figure 1. Given a set of training samples, the mapping $p = f(\boldsymbol{h})$ from a feature vector to a head pose can be learned through various classification algorithms. The mapping function can be used to estimate an unknown head pose p^* from a new feature vector $\boldsymbol{h}^*$ in test scenes. As discussed above, an important problem that was largely ignored in the previous studies on head pose estimation from low resolution images is how to obtain appropriate training samples. Since we assume the underlying mapping function $f(\boldsymbol{h})$ is identical in both training and test scenes, classification accuracy highly depends on how similar these training and test scenes are. In other words, if lighting conditions and camera positions are significantly different between the scenes where training and test images are taken, mappings between pose and appearance would also become different. However, it is not always possible to collect training samples for every test case.

Our basic idea is to use walking direction as a cue to acquire training samples with automatically assigned labels of their head poses. Figure 2 shows a basic framework of our method. Given an input video sequence, we first track pedestrians in the video and obtain their head images and directions they are walking in. As these pedestrians are most likely to turn their head to their walking directions, the walking directions can be assumed to indicate the head poses of the images.

However, this idea cannot be applied in a straightforward manner. Since people can move their heads freely even while they are walking, it is obvious that our basic assumption does not always hold and the training labels contain a certain amount of noise. Head pose estimation algorithms are not always robust to such outliers, and thus it is ideal to reject them prior to the learning stage. Furthermore, walking directions are unevenly distributed in most of the scenes, and this can result in a biased estimation result with larger error.

To address this problem, we introduce a strategy to reject unreliable data from the tracking results. Each tracking trajectory is first divided into straight line segments in which each pedestrian walks in a straight line. Unreliable line segments are rejected and then one representative image per line segment is constructed and used as the training data. Oversampling is then applied to handle the imbalanced dataset. Details of the proposed strategies are described in the following sections.

2.1 Pedestrian Tracking

We used the Benfold *et al.*'s method[6] to track pedestrians in input videos. The method combines a head detector and velocity estimation with feature tracking. With this method, not only are the head of people in the video properly tracked but method also yields good results on centering the head, which is in most cases vital to appearance-based head pose estimation. Here the pedestrian tracking method is briefly explained for

reader's benefit. For more details, readers are referred to[6].

The tracking method is based on a Kalman Filter[12] with two types of measurements. The head location from the head detector based on a histogram of oriented gradients (HOG)[17] and head movement velocity by combining the velocities of multiple tracked corner features[13),22]. For every tracked frame, the head image $\boldsymbol{h}$, the head location $\boldsymbol{u} = (x, y)$, and the size of covariance matrix for each location measurement $\boldsymbol{c} = (c^{(x)}, c^{(y)})$ that can be used as an error measurement is then collected for analysis.

The pedestrian tracking algorithm is applied to the whole input sequence and a trajectory, *i.e.*, a set of head images $\{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N\}$, head locations $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N\}$ with error measurements $\{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_N\}$, is acquired for each pedestrian. N denotes the length of the trajectory and it varies for each trajectory.

2.2 Walking Direction Estimation

As discussed above, our method first divides the trajectories into straight line segments. More specifically, each trajectory is divided into M segments $\{S_1, \ldots, S_M\}$ by polyline simplification using the Douglas-Peucker algorithm[10]. Douglas-Peucker constructs a minimal set of lines so that the orthogonal distance from each point to the nearest line is less than a threshold. The algorithm starts by taking a set of points $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N\}$, then constructs a line from point $\boldsymbol{u}_1$ to $\boldsymbol{u}_N$. The algorithm then finds a point $\boldsymbol{u}_n$ with maximum orthogonal distance from the line. If the distance is more than a threshold, the algorithm divides the point set to $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n\}$ and $\{\boldsymbol{u}_n, \ldots, \boldsymbol{u}_N\}$ and repeats the above process on these two sets recursively. The algorithm stops when the maximum distance becomes less than the threshold.

Figure 3 shows an example of polyline simplification. It can be seen that the pedestrian in this image is not walking straightly, and thus treating all images as one direction will definitely be erroneous. With the polyline simplification algorithm, the trajectory is divided into 4 line segments in which the pedestrian walks straight. In the figure, the curved line shows the raw tracking result and straight lines show line segments obtained using the polyline simplification algorithm.

Next, the walking direction of each line segment is estimated. Since polyline simplification only considers the start and the end point of each segment, using polyline simplification to estimate the pose direction may yield an inaccurate result. Therefore, a line fitting method which considers all points in the segment is employed to analyze and estimate the pose direction for each segment. Given a set of T tracked head locations in the segment, a line which minimizes a sum of residuals is computed and the direction of the line p is assigned to the segment as the walking direction.

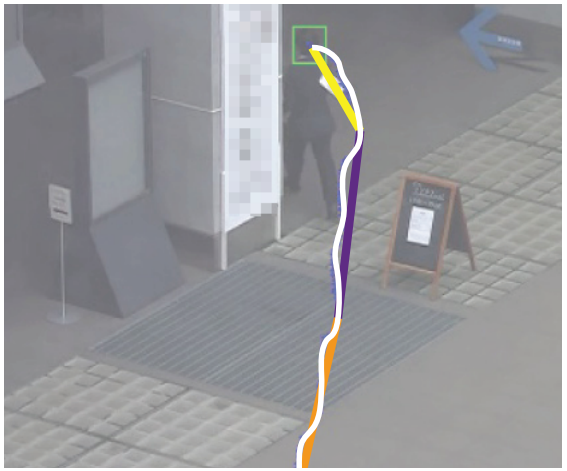


Fig. 3 Example of polyline simplification result. Curved line shows tracking result and straight lines show simplification result.

2.3 Outlier Segment Rejection

After the polyline simplification and the line fitting, walking directions can be estimated accurately for pedestrians and can be used as their head orientations. However, as discussed above, walking directions do not always correspond to head orientations and line segments are not always suitable for training samples and a scheme for rejecting outlier segments is necessary.

In this work, we apply four rules to reject segments: 1) with a high number of erroneous points, 2) in which a person walks short distances or walks slowly, 3) with large line fitting errors, and 4) with high image variance. The details of each rule are as follows.

Segments with a high number of erroneous points

Let us denote the T head locations in the segment as $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_T\}$. Segments with many erroneous points, *i.e.*, points that are regarded to be false positives of the tracking algorithm, are rejected because of low reliability. Specifically, a point $\boldsymbol{u}_t$ is judged as erroneous if the error measurement of the tracker is significantly large compared to head

sizes:

$$\frac{c_t^{(x)}}{s_x(\boldsymbol{u}_t)} > \alpha \text{ and } \frac{c_t^{(y)}}{s_y(\boldsymbol{u}_t)} > \alpha. \quad (1)$$

where $\boldsymbol{c}_t = (c_t^{(x)}, c_t^{(y)})$ is the error measurement of the corresponding frame and α is a constant value. $s_x(\boldsymbol{u})$ and $s_y(\boldsymbol{u})$ are position-dependent head width and height defined as:

$$s_x(\boldsymbol{u}_t) = Ax_t + By_t + C \text{ and } s_y(\boldsymbol{u}_t) = Rs_x(\boldsymbol{u}_t), \quad (2)$$

which assumes that heads are fixed size and moving on a plane under a perspective projection. The parameters A, B, C and R are manually set. Using this measure, reject segments if the number of erroneous segment points is larger than a predefined threshold τ_e.

Segments with short distance or slow movement

Short segments are better to be rejected since they do not have enough information for the line fitting. Similarly, segments with slow walking speed are rejected since the pedestrians are likely to be doing something else, *e.g.*, talking with each other and not facing straight. Specifically, reject segment if

$$\frac{|\boldsymbol{u}_T - \boldsymbol{u}_1|}{\bar{s}} \leq \tau_n \text{ or } \frac{|\boldsymbol{u}_T - \boldsymbol{u}_1|}{T \cdot \bar{s}} \leq \tau_v \quad (3)$$

where τ_n and τ_v are predefined thresholds. $\bar{s} = \sum_{t=1}^{T} \sqrt{s_x(\boldsymbol{u}_t)^2 + s_y(\boldsymbol{u}_t)^2}/T$ is the average head size factor of the segment and introduced to make the measurement scale-invariant.

Segments with large line fitting error

If the line fitting error is large, it is natural to assume that the person is moving in a curve or the tracker failed to track the head. Therefore, reject segments if

$$\sum_{t=1}^{T} \frac{|y_t - g(x_t)|}{\sqrt{m^2+1} \cdot |\boldsymbol{u}_T - \boldsymbol{u}_1|} \geq \tau_l, \quad (4)$$

where τ_l is a threshold and the left side of the equation is a scale-independent line fitting error of the estimated line $y = g(x) = mx + c$.

Segments with high image variance

Segments with high image variance imply unstable head images due to, *e.g.*, frequent changes of head orientations. Specifically, we calculate the variance of resized I-dimensional head image vectors $\{\hat{\boldsymbol{h}}\}$. Segments are considered to have high variance if

$$\frac{\sum_{t=1}^{T} |\hat{\boldsymbol{h}}_t - \bar{\boldsymbol{h}}|^2}{T \cdot I} \geq \tau_{var}, \quad (5)$$

where $\hat{\boldsymbol{h}}_t$ denotes the t-th resized image, and $\bar{\boldsymbol{h}}$ is a mean image calculated from all resized images $\hat{\boldsymbol{h}}$ in the segment.

2.4 Selecting Representative Images

With the rules above, most outlier segments are rejected and the remaining segments contain correct data. One representative image per segment is then selected and used as training data. Since only one orientation is assigned to each segment, most of the images in the accepted segments are redundant. Moreover, it is beneficial to use only one image per a segment in order to reduce computational cost of training classifiers.

In this work, we propose and examine three different selection methods. Basically, we select the image which is most similar to the mean image of the segment. For each segment, the Mahalanobis distance from the mean image is calculated for every resized image $\hat{\boldsymbol{h}}_t$ in the segment and the image with lowest distance is selected. This enables us to select the most representative image without suffering from effects that can be seen in the mean image, *e.g.*, blur or distortion. However, it is not always the case that blur and distortion cause poor estimation results. We also found simply using the mean or median image as the representative image can be an option in some cases. Further discussion will be given in Section 3.

2.5 Handling Imbalanced Data by Oversampling

By applying these processes to all of the successfully tracked pedestrians, a scene-specific dataset is acquired. Figure 4 shows an example of a distribution of walking directions in the sequence 1 (See section 3 for more details). In this sequence, the majority of the pedestrians walk in down-left (class 2) and up-right (class 6) directions. Imbalanced data causes low accuracy of head pose estimation for directions with few training samples. In order to handle this problem, oversampling is used with our classifiers. Oversampling technique is proved to be beneficial in reducing the effect of imbalanced data for classification task[1),24)]. The oversampling technique resamples data from classes with small amounts of data until every class has an equal number of data. By using oversampling, the accuracy of the classifiers is significantly improved.

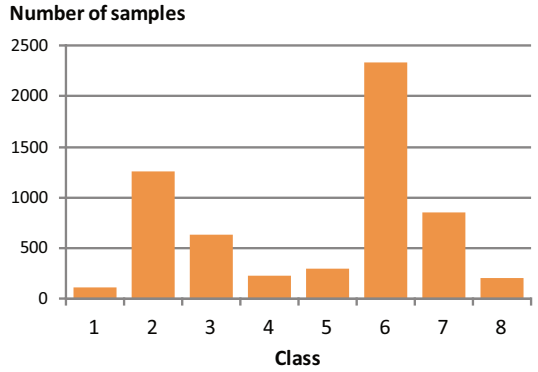


Fig. 4 Head pose frequency captured from the video sequence 1. The horizontal axis indicates tracked walking directions and the vertical axis indicates numbers of samples obtained from the video sequence. Each class number is defined in same way as in Figure 1.

3. Experimental Results

In this section, we present experimental results to demonstrate the effectiveness of our method. As mentioned before, it is hard to collect ground truth data manually for every scene, thus we show that training data from our method which automatically generates scene-specific training data performs better than using available data from other scenes. In order to demonstrate that our method is not limited to one classifier, a multi-class SVM classifier and a Random Trees classifier are also tested. The effectiveness of the head image selection method is also compared to other alternatives. The effect of imbalanced data and the effectiveness of oversampling method are also shown.

3.1 Experiment Settings

We conducted experiments using 3 video sequences with different length, which were recorded using different cameras in different scenes. Example frames in the video are shown in Figure 5. The scene images are input video frames with pedestrian tracking results overlaid. Examples of obtained head images are also shown with its estimated walking direction shown on their right part of the image.

The resolution of sequence 1 was 1920×1080 pixels and recorded at 30 fps for approximately 7 hours. Sequence 2 was 1120×780 pixels and recorded at 30 fps for approximately 10 minutes. Sequence 3 was 1280×720 pixels and recorded at 30 fps for approximately 10 minutes. As a result of pedestrian tracking, direction estimation and image selection, 5930 head images were captured from sequence 1, 1265 head images were captured from sequence 2, and 564 head images were captured from sequence 3. At the same time, test images (320 for sequence 1, 314 for sequence 2 and 227 for sequence 3) with manually-labeled ground-truth head poses were acquired from the same sequence and estimation accuracy was evaluated using these test images. To construct a generic dataset, 1477 samples were taken from Gaze Direction Dataset [4], which has been used in [6]. The set is divided by head pose into 8 classes and each class contains $100 \sim 200$ images. Figure 6 shows examples of head images included in the generic dataset.

Table 1 Settings of scene-dependent parameters.

Sequence	1	2	3
A	0	0	0.008
B	0.014	0.04	0.04
C	36.4	8	33
R	1.1	1.1	1.1
τ_n	3.0	0.5	0.5
τ_v	0.03	0.03	0.01
τ_{var}	0.0035	0.0055	0.0035

In the experiments, the parameters were empirically set as follows; $\alpha = 1.0$, $\tau_l = 0.8$, $\tau_e = 0.4$, and other scene-dependent parameters were set as summarized in Table 1.

3.2 Head Pose Estimation Test

For the head pose estimation method, we conducted two classification tests. The first classification test uses a linear SVM as classifier from the Liblinear library[11] and the second test uses a Random Trees classifier[8] from the OpenCV library[7]. All of the head images were converted to gray scale, normalized and resized to 20×22 pixels. Feature vector was defined as a 440-dimensional raster-scanned and normalized image vector in the following experiments. To evaluate the effectiveness of the proposed method, we compared two results: **Generic** result based on the generic dataset and **Proposed** result based on our method.

Classification accuracy comparison between our proposed method and the generic dataset is summarized in Figure 7. The accuracy is calculated from the average accuracy of each class. Standard deviations are indicated as error bars.

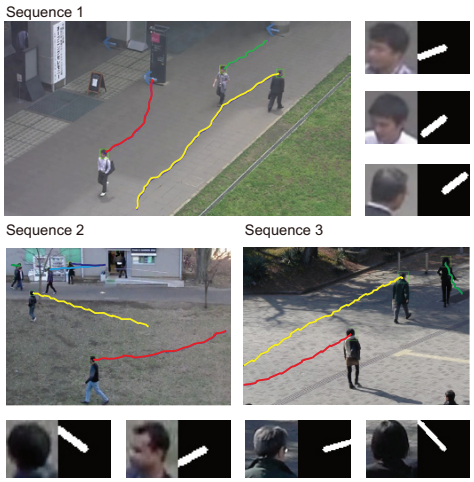
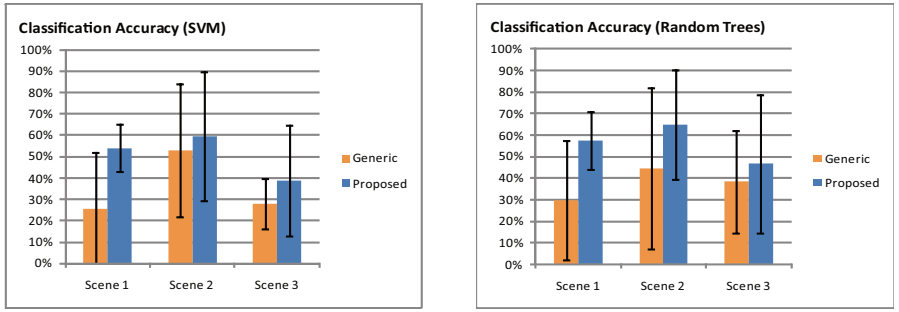


Fig. 5 Example frames in the test video sequences. The input video frames are overlaid with pedestrian tracking results . Examples of obtained head images are also shown, the right part of each image presents its estimated walking direction.

Fig. 6 Example images in the generic dataset.

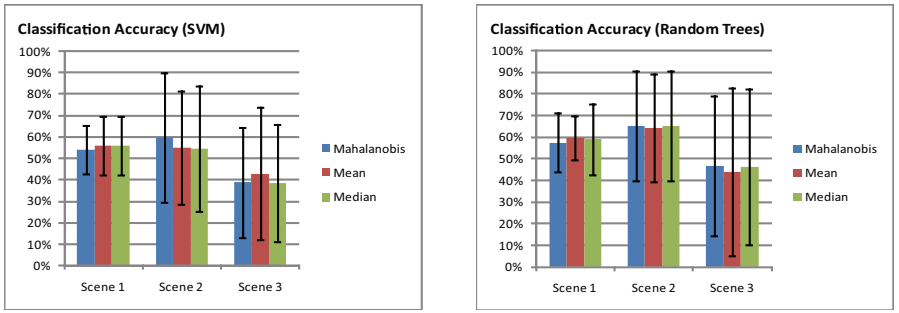
As can be seen, the accuracy of classification using the generic dataset is significantly lower. In contrast, our proposed method achieved higher accuracy than the generic result. The result also shows that the accuracy improved a lot for sequence 1 which utilizes 7 hours video. It is also shown that even for only 10 minutes of video as in sequences 2 and 3, the accuracy is significantly improved over using the generic dataset. Standard deviations also become smaller in the proposed result.

We also compared our representative image selection method with 2 other selection methods. For the first alternative, we calculate the mean image from images in each segment and use it as the representative image. For the second alternative, we calculate



(a) SVM classifier (b) Random Trees classifier

Fig. 7 Accuracy of the head pose classification for SVM classifier and Random Trees classifier. **Generic** result is based on the generic dataset, **Proposed** result is based on our proposed method. The accuracy is based on normalized average of 8 classification classes. Standard deviations are indicated as error bars.



(a) SVM classifier (b) Random Trees classifier

Fig. 8 Accuracy of head pose estimation for each image selection method. **Mahalanobis** result is based on using Mahalanobis distance to select sample. **Mean** result is based on using mean image to select sample. **Median** result is based on using median image to select sample. The accuracy is based on normalized average of 8 classification classes. Standard deviations are indicated as error bars.

the median image. Each pixel in the median image is constructed from the median pixel intensity at its location over all images in the segment.

Classification accuracy for each image selection method is summarized in Figure 8. It can be seen that the Mahalanobis image selection method performs comparatively well to other methods. However, it should be also noted that the other two methods also show

better results than the generic results in Figure 7. It indicates that our proposed idea has robustness to the image selection method.

Figure 9 shows confusion matrices of the classifiers. We compared confusion matrices of classifiers applied to test data from scene 1. The **Generic** result uses generic dataset as training data for the classifiers, the **Imbalanced** result uses data obtained from our method without oversampling data to train the classifiers, and the **Proposed** result uses data obtained from our method applied with oversampling to train the classifiers. The effects of imbalanced data can be seen in Figure 9(c) 9(d). As the majority of pedestrians in the sequence walked in the direction as shown in Figure 4, the results are biased towards class 2 and 6. The improvement using oversampling can clearly be seen in Figure 9(e) and 9(f).

4. Conclusions

In this paper, we proposed a method of appearance-based head pose estimation which can be automatically adapted to test scenes. The key idea behind the proposed framework is to use walking directions as a cue to infer head pose directions in collecting a scene-specific training dataset. A pedestrian tracker is first applied to the input video sequence, and a scene-specific dataset of head images labeled with their walking directions is automatically acquired.

We applied our framework to SVM-based and Random Tree-based classification tasks. The results of both experiments show that our method estimates head pose more accurately than directly using generic datasets on the test scenes. In this sense, accuracy can be improved without the need to manually collect a ground-truth dataset in real scenes. This is a great advantage of our method compared to existing methods when applied to practical scenarios.

Appearance-based head pose estimation from low-resolution images is itself a difficult task, and there is a much room for improvement in both feature description and classification/regression techniques. We believe that investigating the estimation algorithm itself based on the proposed idea is an important future task.

References

1) Akbani, R., Kwek, S. and Japkowicz, N.: Applying Support Vector Machines to Imbalanced Datasets, *Machine Learning: ECML 2004* (Boulicaut, J.-F., Esposito, F., Giannotti, F. and Pedreschi, D., eds.), Lecture Notes in Computer Science, Vol.3201, Springer Berlin / Heidelberg, pp.39–50 (2004).
2) Balasubramanian, V., Ye, J. and Panchanathan, S.: Biased manifold embedding: a framework for person-independent head pose estimation, *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR2007)*, pp.1–7 (2007).
3) BenAbdelkader, C.: Robust Head Pose Estimation Using Supervised Manifold Learning, *Proc. 11th European Conference on Computer Vision (ECCV2010)*, pp.518–531 (2010).
4) Benfold, B. and Reid, I.D.: http://www.robots.ox.ac.uk/˜lav/Research/Projects/2009bbenfold_headpose/project.html.
5) Benfold, B. and Reid, I.D.: Colour Invariant Head Pose Classification in Low Resolution Video, *Proc. British Machine Vision Conference (BMVC2008)* (2008).
6) Benfold, B. and Reid, I.D.: Guiding Visual Surveillance by Tracking Human Attention, *Proc. British Machine Vision Conference (BMVC2009)* (2009).
7) Bradski, G.: The OpenCV Library, *Dr. Dobb's Journal of Software Tools* (2000).
8) Breiman, L.: Random Forests, *Machine Learning*, Vol.45, pp.5–32 (2001).
9) Cootes, T.F., Edwards, G.J. and Taylor, C.J.: Active Appearance Models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.23, No.6, pp.681–685 (2001).
10) Douglas, D.H. and Peucker, T.K.: ALGORITHMS FOR THE REDUCTION OF THE NUMBER OF POINTS REQUIRED TO REPRESENT A DIGITIZED LINE OR ITS CARICATURE, *Cartographica: The International Journal for Geographic Information and Geovisualization*, Vol.10, pp.315–354 (1973).
11) Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R. and Lin, C.-J.: LIBLINEAR: A Library for Large Linear Classification, *Journal of Machine Learning Research*, Vol.9, pp.1871–1874 (2008).
12) Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems, *Transactions of the ASME–Journal of Basic Engineering*, Vol.82, No.Series D, pp.35–45 (1960).
13) Lucas, B.D. and Kanade, T.: An Iterative Image Registration Technique with an Application to Stereo Vision, *Proc. 7th international joint conference on Artificial intelligence*, Vol.2, pp. 674–679 (1981).
14) Matthews, I. and Baker, S.: Active appearance models revisited, *International Journal of Computer Vision*, Vol.60, No.2, pp.135–164 (2004).
15) Murphy-Chutorian, E. and Trivedi, M.M.: Head Pose Estimation in Computer Vision: A Survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.31, No.4, pp. 607 –626 (2009).
16) Orozco, J., Gong, S.G. and Xiang, T.: Head Pose Classification in Crowded Scenes, *Proc. British Machine Vision Conference (BMVC2009)* (2009).
17) Prisacariu, V. and Reid, I.: fastHOG - a real-time GPU implementation of HOG, Technical Report 2310/09, Department of Engineering Science, Oxford University (2009).
18) Robertson, N.M. and Reid, I.D.: Estimating gaze direction from low-resolution faces in video, *Proc. 9th European Conference on Computer Vision (ECCV2006)*, Vol.3952/2006, pp.

402–415 (2006).

19) Robertson, N.M., Reid, I.D. and Brady, J.M.: What are you looking at? Gaze estimation in medium-scale images, *Proc. HAREM05, 16th British Machine Vision Conference, Oxford, September 2005* (2005).

20) Seemann, E., Nickel, K. and Stiefelhagen, R.: Head pose estimation using stereo vision for human-robot interaction, *Proc. Sixth IEEE International Conference on Automatic Face and Gesture Recognition (FG'04)*, pp.626–631 (2004).

21) Tian, Y.-L., Brown, L., Connell, J., Pankanti, S., Hampapur, A., Senior, A. and Bolle, R.: Absolute Head Pose Estimation From Overhead Wide-Angle Cameras, *Proc. IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, pp.92–99 (2003).

22) Tomasi, C. and Kanade, T.: Detection and Tracking of Point Features, Technical report, CMU-CS-91-132, Carnegie Mellon University (1991).

23) Wang, X., Huang, X., Gao, J. and Yang, R.: Illumination and Person-Insensitive Head Pose Estimation Using Distance Metric Learning, *Proc. 10th European Conference on Computer Vision (ECCV2008)*, Vol.2, pp.624–637 (2008).

24) Weiss, G.M. and Provost, F.: Learning when Training Data are Costly: The Effect of Class Distribution on Tree Induction, *Journal of Artificial Intelligence Research*, Vol.19, pp.315–354 (2003).

	1	2	3	4	5	6	7	8
1	0.15	0.25	0.03	0.13	0.23	0.2	0.03	0
2	0.03	0.1	0.05	0.3	0.23	0.2	0.1	0
3	0.03	0.15	0.03	0.13	0.38	0.25	0.05	0
4	0	0.03	0	0.08	0.53	0.38	0	0
5	0	0	0	0	0.53	0.48	0	0
6	0	0	0	0	0.13	0.83	0.05	0
7	0	0.03	0.08	0.03	0.13	0.6	0.15	0
8	0.1	0.03	0.03	0.08	0.2	0.2	0.3	0.08

(a) **SVM - Generic**

	1	2	3	4	5	6	7	8
1	0.25	0.5	0	0	0.1	0.13	0.03	0
2	0.1	0.28	0	0.2	0.33	0.1	0	0
3	0.03	0.2	0.03	0.33	0.4	0.03	0	0
4	0	0.05	0	0.05	0.9	0	0	0
5	0	0	0	0	0.85	0.15	0	0
6	0	0	0	0	0.35	0.58	0.08	0
7	0.13	0	0	0	0.1	0.43	0.35	0
8	0.25	0	0	0	0.05	0.15	0.55	0

(b) **Random Trees - Generic**

	1	2	3	4	5	6	7	8
1	0	0.8	0	0	0	0.13	0.08	0
2	0	0.95	0	0	0	0.05	0	0
3	0	0.6	0.18	0	0	0.23	0	0
4	0	0.08	0.05	0	0	0.88	0	0
5	0	0	0	0	0	1	0	0
6	0	0	0	0	0	0.93	0.08	0
7	0	0.1	0	0	0	0.43	0.48	0
8	0	0.13	0	0	0	0.15	0.73	0

(c) **SVM - Imbalanced**

	1	2	3	4	5	6	7	8
1	0	0.95	0	0	0	0	0.05	0
2	0	1	0	0	0	0	0	0
3	0	0.8	0.1	0	0	0.1	0	0
4	0	0.08	0.1	0	0	0.8	0.03	0
5	0	0	0	0	0	1	0	0
6	0	0	0	0	0	0.98	0.03	0
7	0	0.13	0	0	0	0.33	0.55	0
8	0	0.3	0	0	0	0.08	0.63	0

(d) **Random Trees - Imbalanced**

	1	2	3	4	5	6	7	8
1	0.58	0.08	0	0	0	0.03	0.03	0.3
2	0.23	0.65	0	0.03	0	0.03	0.03	0.05
3	0.03	0.23	0.43	0.18	0.03	0	0	0.13
4	0	0	0.05	0.7	0.13	0.08	0	0.05
5	0	0.05	0.03	0.2	0.38	0.3	0.05	0
6	0.03	0	0	0.08	0.05	0.5	0.33	0.03
7	0	0	0.03	0.03	0.03	0.15	0.65	0.13
8	0.08	0.05	0	0	0	0.03	0.4	0.45

(e) **SVM - Proposed**

	1	2	3	4	5	6	7	8
1	0.55	0.05	0.03	0	0	0	0	0.38
2	0.25	0.68	0	0.03	0	0	0	0.05
3	0.08	0.23	0.58	0.1	0.03	0	0	0
4	0	0	0.1	0.63	0.2	0.03	0	0.05
5	0	0.03	0.03	0.13	0.68	0.15	0	0
6	0	0.03	0	0.03	0.3	0.3	0.35	0
7	0.05	0.05	0	0	0.05	0.08	0.45	0.33
8	0.1	0.05	0	0	0	0	0.1	0.75

(f) **Random Trees - Proposed**

Fig. 9 Confusion matrix of SVM classifier and Random Trees classifier using data from scene 1 with Mahalanobis distance as the image selection method. Each class number is defined in same way as in Figure 1. **Generic** result is based on using generic dataset as training data. **Imbalanced** result is based on scene-specific dataset without oversampling method. **Proposed** result is based on using the mean image as selection method.