Head Pose Estimation Using Multi-scale Gaussian Derivatives

Varun Jain[1,2] and James L. Crowley[1,3]

[1] INRIA Grenoble Rhône-Alpes
[2] Université de Grenoble
[3] Grenoble INP
{varun.jain,james.crowley}@inria.fr

Abstract. In this paper we approach the problem of head pose estimation by combining Multi-scale Gaussian Derivatives with Support Vector Machines.

We evaluate the approach on the Pointing04 and CMU-PIE data sets and to estimate the pan and tilt of the head from facial images. We achieved a mean absolute error of 6.9 degrees for pan and 8.0 degrees for tilt on the Pointing04 data set.

1 Introduction

The problem of head pose estimation has been approached by the computer vision community in two ways: model based approaches and appearance based approaches. In model based approaches facial key points like eyes, eyebrows, nose, lips etc. have to be located and tracked and then the pose is estimated according to the relative position of these facial key points [1,2,3].

In holistic or appearance based approaches an image descriptor is used to represent the image and a feature vector is assembled using the descriptor values. Then a suitable machine learning technique is used for discrimination between different poses [4,5].

A major problem with model based approaches is that key point detection is a difficult task and tracking these key points is all the more likely to fail. With holistic approaches one has to choose an image descriptor and a machine learning technique from a wide array of options. In contrast to the model based approaches one needs training and testing data to make such approaches work but these approaches do not suffer from issues like facial key point detection and tracking failure.

Stiefelhagen in [5] used horizontal and vertical image derivatives of the first order and used neural networks for discrimination between different poses and applied this approach on the Pointing04 data set. A survey on head pose estimation methods [6] shows that Stiefelhagen achieved the best results so far on the Pointing04 data set.

We in this paper employ Multi-scale Gaussian Derivatives(MGD) and Support Vector Machines(SVM) for head pose estimation on the Pointing04 dataset [7] and show that our choice of descriptor gives better results than those obtained so far.

J.-K. Kämäräinen and M. Koskela (Eds.): SCIA 2013, LNCS 7944, pp. 319–328, 2013.

In the next section we briefly describe the problem of head pose estimation and the Pointing04 data set.

2 Head Pose Estimation and the Pointing04 Data Set

The problem of head pose estimation involves inferring the orientation of the head from static images or video. It is assumed that the human head has three degrees of freedom, in this paper we estimate only two degrees of freedom namely pan and tilt and the problem is treated as a multi-class classification problem.

To solve the problem of head pose estimation we need to choose an appropriate descriptor to extract features from the image and then a pattern recognition algorithm is required to discriminate between the different poses.

The approach that we present in the following sections was tested on the Pointing04 data set [7,8]. This data was collected by the PRIMA team at INRIA Grenoble Research Center where 15 people were asked to gaze successively at 93 markers that cover a half-sphere in front of the person. The head pose database consists of 15 sets of images. Each set contains of 2 series of 93 images of the same person at different poses. There are 15 people in the database, wearing glasses or not and having various skin colors. The pose, or head orientation is determined by 2 angles (pan,tilt), which varies from -90 degrees to +90 degrees.

Fig. 1. A small sequence from the Pointing04 dataset

We use two support vector machines for discriminating between different poses; one is trained for pan and the other for tilt.

The next section discusses Gaussian Derivatives briefly.

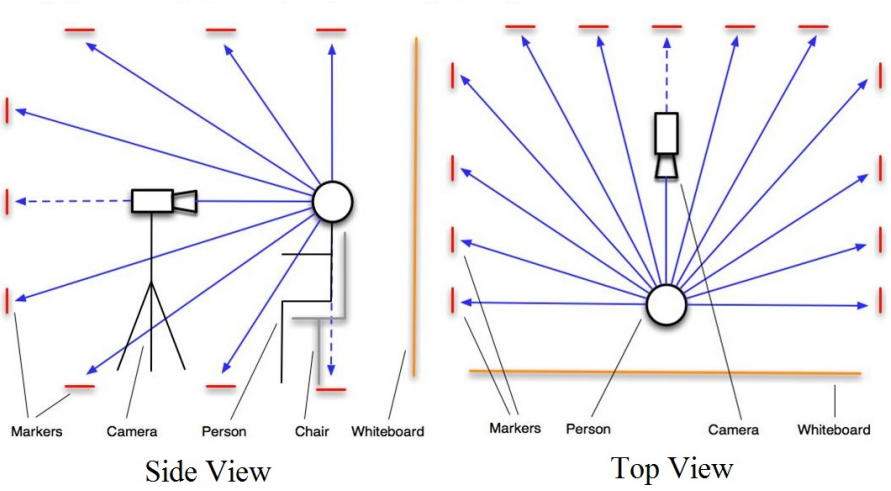

Fig. 2. How the Pointing04 database was collected

3 Gaussian Derivatives

Gaussian derivatives can efficiently describe the neighborhood appearance of an image for recognition and matching[9]. This can be done by calculating several orders of Gaussian derivatives normalized in scale and orientation at every pixel.

The basic Gaussian function is defined as:

$$G(x, y; \sigma) = e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

Here σ is the scale factor or variance and defines the spatial support. This function measures the intensity of the neighborhood and does not contribute to the identification of the neighborhood and can be omitted. The first order derivatives are of the form:

$$G_x(x, y; \sigma) = \frac{\partial G(x, y; \sigma)}{\partial x} = -\frac{x}{\sigma^2} G(x, y; \sigma) \tag{2}$$

$$G_y(x, y; \sigma) = \frac{\partial G(x, y; \sigma)}{\partial y} = -\frac{y}{\sigma^2} G(x, y; \sigma) \tag{3}$$

First order derivatives give information about the gradient (intensity and direction). The second order derivatives are given by:

$$G_{xx}(x, y; \sigma) = \frac{\partial^2 G(x, y; \sigma)}{\partial x^2} = (\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2}) G(x, y; \sigma) \tag{4}$$

$$G_{yy}(x, y; \sigma) = \frac{\partial^2 G(x, y; \sigma)}{\partial y^2} = (\frac{y^2}{\sigma^4} - \frac{1}{\sigma^2}) G(x, y; \sigma) \tag{5}$$

$$G_{xy}(x, y; \sigma) = \frac{\partial^2 G(x, y; \sigma)}{\partial x \partial y} = \frac{xy}{\sigma^4} G(x, y; \sigma) \tag{6}$$

Second order derivatives provide us with information about image features such as bars, blobs and corners. Higher order derivatives are only useful if the second order derivatives are strong otherwise they just contain image noise.

Normalizing Gaussian derivatives in scale is not a trivial task. Several methods have come up in the past addressing this problem. It was suggested by Lindeberg in [10] that Gaussian derivatives be calculated across scales to get scale invariant features and then Lowe in [11] defined the intrinsic or characteristic scale as the value of the scale parameter at which the Laplacian provides a local maximum. The computational cost of directly searching the scale axis for this characteristic scale can be prohibitively expensive. A cost-effective method for computing Multi-scale Gaussian derivatives is described in the following section. The inverse-tangent of the ratio of first order derivatives at any image point is considered to be the direction of the gradient. It has been shown that Gaussian derivatives are steerable [12] and by using appropriate trigonometric ratios the Gaussian derivatives can be rotated in the desired direction.

4 Half-Octave Gaussian Pyramid (Multi-scale Gaussian Derivatives)

This algorithm has been discussed in detail in [13] and an integer coefficient version of the same can constructed using repeated convolutions of the binomial kernel (1, 2, 1).

The algorithm involves repeated convolutions with a Gaussian kernel in a cascaded configuration where the process is speeded up by approximating a Gaussian filter with separable binomial filters as shown below:

$$G(x, y; \sqrt{2}) = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \tag{7}$$

A key feature of this algorithm is that for different levels of the pyramid the difference of adjacent image pixels in the row and column directions are equivalent to convolution with Gaussian derivatives.

The pyramid is very easy to access, derivative values can be determined for every image position by using bilinear interpolation and derivatives between scale values can be computed using quadratic interpolation between adjacent levels of the pyramid. The following sets of equations explain how different order of derivatives can be calculated using difference of adjacent image pixels in the row and column directions:

$$\frac{\partial p(x, y, k)}{\partial x} = p * G_x(x, y; 2^k \sigma_0) \approx p(x + 1, y, k) - p(x - 1, y, k) \tag{8}$$

$$\frac{\partial p(x, y, k)}{\partial y} = p * G_y(x, y; 2^k \sigma_0) \approx p(x, y + 1, k) - p(x, y - 1, k) \tag{9}$$

$$\frac{\partial^2 p(x,y,k)}{\partial x^2} = p * G_{xx}(x,y;2^k\sigma_0)$$
$$\approx p(x+1,y,k) - 2p(x,y,k) + p(x-1,y,k) \quad (10)$$

$$\frac{\partial^2 p(x,y,k)}{\partial y^2} = p * G_{yy}(x,y;2^k\sigma_0)$$
$$\approx p(x,y+1,k) - 2p(x,y,k) + p(x,y-1,k) \quad (11)$$

$$\frac{\partial^2 p(x,y,k)}{\partial x \partial y} = p * G_{xy}(x,y;2^k\sigma_0)$$
$$\approx p(x+1,y+1,k) - p(x+1,y-1,k)$$
$$- p(x-1,y+1,k) + p(x-1,y-1,k) \quad (12)$$

In the above equations at the k_{th} level of the pyramid the support is defined by $\sigma_k = 2^k\sigma_0$ and the image at the same level is defined by $p(x,y,k)$.

The next section is about dimensionality reduction using Principal Component Analysis(PCA) and why we need it.

5 Principal Component Analysis

In our experiments the part of the image containing the face was normalized to 24 X 36 pixels this size of 24 X 36 pixels for the normalized region was chosen after extensive experimentation where normalized images of 24 X 36 pixels gave better results at head pose estimation as compared to other sizes. Then several types of derivatives were calculated at 2 levels of scale however the region within 4 pixels of the image boundary is ignored because of boundary effect.

Principal component analysis was then used to omit correlated dimensions by transforming the original dimensions into new dimensions which are a linear sum of the original dimensions but are linearly uncorrelated. Then these new dimensions are ranked according to the variance i.e. the dimension which accounts for the most variability in the data gets the first rank and so on [14].

PCA is done by eigenvalue decomposition of the data correlation matrix after normalizing the data for each dimension. PCA provides you with scores and loadings. The scores are the transformed values corresponding to your data point and loadings are the coefficients your original variable should be multiplied with to get the score.

We found that just 12 dimensions out of 4480 dimensions can account for most of the variability in data and give us an acceptable accuracy. PCA was used only for dimensionality reduction, Support Vector Machines were then used to discriminate between different poses. In section seven we compare the execution time of the SVM with and without using PCA.

6 Support Vector Machines

Support Vector Machines (SVM) belong to a family of generalized linear classifiers and can be interpreted as an extension of the perceptron [15].

After using several types of kernels we settled on the radial basis kernel as it provided us with the maximum accuracy, represented by the following equation:

$$K(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}} \tag{13}$$

The SVM employed was a soft margin SVM, soft margin SVMs are used when the classes are not separable even after transforming the data to a higher dimension. The condition for the optimal hyper-plane can be relaxed by including an extra term ξ [16]:

$$y_i(X_i^T W + b) \geq 1 - \xi_i, (i = 1, ..., m) \tag{14}$$

For minimum error,ξ_i should be minimized as well as $||W||$, and the objective function becomes:

$$\begin{aligned} &minimize \quad W^T W + C \sum_{i=1}^{m} \xi_i^k \\ &subject\,to \;\; y_i(X_i^T W + b) \geq 1 - \xi_i, \;\; and \;\; \xi_i \geq 0; (i = 1, ..., m) \end{aligned} \tag{15}$$

Here C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error. $1/\gamma$ or σ is the width of the radial basis kernel. The C-penalty parameter was chosen using cross validation. For the data in hand the value $C = 100$ and $\sigma = 11$ lead to the maximum correlation coefficient between the predicted pan angles and the actual pan angles. Similarly $C = 140$ and $\sigma = 6$ lead to the maximum correlation coefficient between the predicted tilt angles and the actual tilt angles.

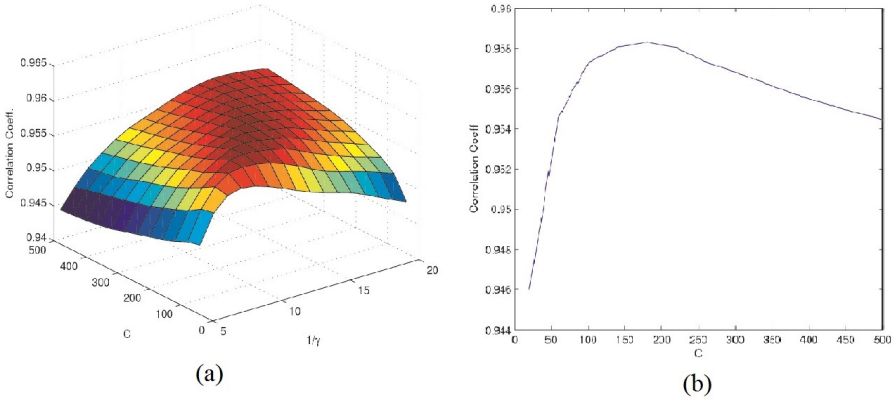

Fig. 3. (a) Graph of Correlation Coeff. vs. C-parameter and $1/\gamma$ for pan and (b) Graph of Correlation Coeff. vs. C-parameter at $1/\gamma = 11$ for pan

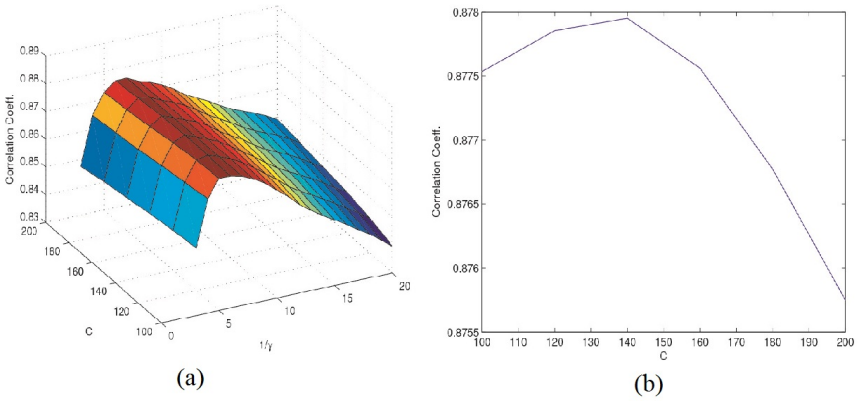

Fig. 4. (a) Graph of Correlation Coeff. vs. C-parameter and $1/\gamma$ for tilt and (b) Graph of Correlation Coeff. vs. C-parameter at $1/\gamma = 6$ for tilt

7 Results

We used 80 percent of the data for training, 10 percent for cross-validation and the rest for testing. Face detection was then performed on the images in the dataset using the OpenCV face detector [17]. Following that a half-octave gaussian pyramid was constructed over a normalized imagette of the face which is of the size 24X36 pixels.

The data was split several times and the accuracy calculated for every split and finally the average was calculated. The results of the Mean absolute error(MEA) are shown in the table below and they are better than the state of the art reported in [6].

Our mean absolute errors of 6.9, 8.0 degrees for pan and tilt respectively are much lower than the best error achieved so far by Stiefelhagen [5] which was 9.5, 9.7 degrees for pan and tilt respectively. The accuracy achieved for the discrete poses: 64.51, 62.72* for pan is much higher than the accuracy reported by Stiefelhagen: 52, 66.3. Our accuracy for tilt is less than the accuracy achieved in [5] because the authors of that paper considered only 7 out of the 9 poses for tilt in the Pointing04 data set.

Table 1. Our MEA as compared with the state-of-the-art

MEA	pan	tilt
our approach	6.9	8.0
state-of-the-art	9.5	9.7

Table 2. Our accuracy over discrete poses as compared with the state-of-the-art

Accuracy%	pan	tilt
our approach	64.51	62.72
state-of-the-art	52	66.3

For continuous poses the correlation coefficients for pan and tilt were found to be 0.95, 0.87 for pan and tilt respectively showing that the proposed system can work well even for continuous poses even though it is trained on a dataset containing only discrete poses. Table 3 and 4 show the confusion matrices for pan and tilt respectively.

Table 3. Confusion Matrix for Pan, true values are in the first column, predicted values in the first row

	-90	-75	-60	-45	-30	-15	0	15	30	45	60	75	90
-90	23	1	0	1	0	0	0	0	0	0	0	0	0
-75	5	17	3	0	0	0	0	0	0	0	0	0	0
-60	0	3	10	1	0	0	0	0	0	0	0	0	0
-45	0	0	0	16	1	1	0	0	0	0	0	0	0
-30	0	0	0	4	11	2	0	0	0	0	0	0	0
-15	0	0	0	0	1	10	6	0	0	0	0	0	0
0	0	0	0	0	0	4	24	2	1	0	0	0	0
15	0	0	0	0	0	0	5	11	3	0	1	0	0
30	0	0	0	0	0	0	0	3	23	2	0	0	0
45	0	0	0	0	0	0	0	0	4	9	4	3	0
60	0	0	0	0	0	0	0	0	2	6	7	10	1
75	0	0	0	0	0	0	0	0	0	1	4	5	4
90	0	0	0	0	0	0	0	0	0	0	2	8	14

Table 4. Confusion Matrix for Tilt, true values are in the first column, predicted values in the first row

	-90	-60	-30	-15	0	15	30	60	90
-90	3	0	0	0	0	0	0	0	0
-60	0	38	7	0	0	0	0	0	0
-30	0	9	13	4	0	1	0	1	0
-15	0	0	8	17	9	1	0	0	0
0	0	0	2	8	21	11	1	0	0
15	0	0	1	0	7	19	12	0	0
30	0	0	0	0	0	10	23	5	0
60	0	0	0	1	0	0	5	37	0
90	0	0	0	0	0	0	1	0	4

Table 5 shows the prediction times with and without using PCA and we can see that the PCA speeds up the prediction time by a factor of around 200.

Table 5. Comparison of prediction time with and without using PCA

	SVM with PCA	SVM without PCA
Prediction time(sec)	0.108	20.17

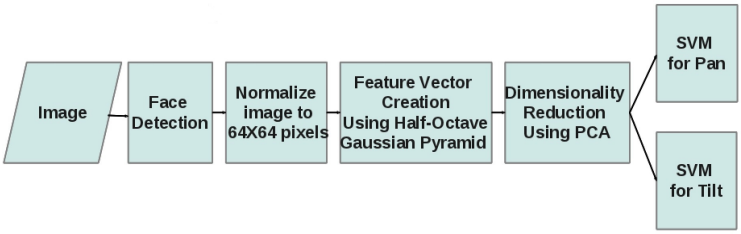

Fig. 5. Schematic of our approach

After training the SVM's on the Pointing04 dataset we test them on the CMU-PIE dataset [18]. Although the CMU-PIE dataset is not labeled for pose and hence does not let you perform a mathematical analysis, we could see that the predicted values of our SVM's were in agreement with the general orientation of the head in the data set.

Two representative images form the PIE dataset are given below along with the results obtained from our SVM's.

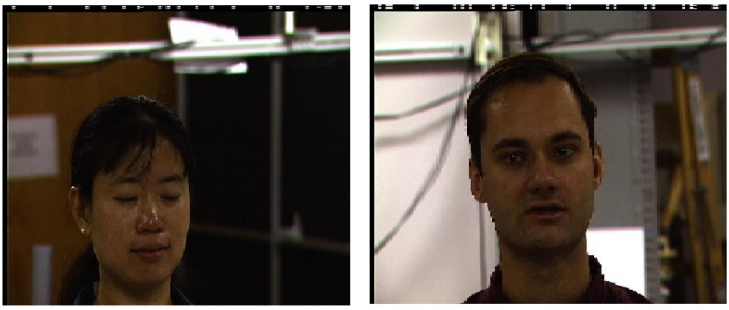

Fig. 6. (a)Pan=-15, Tilt=-15 and (b)Pan=0, Tilt=0 were predicted using our approach

8 Conclusion

We have shown that our approach works better than the state-of-the-art on the Pointing04 dataset. Multi-scale Gaussian Derivatives can be used effectively for image representation and the problem of high-dimensionality can be resolved using PCA.

The approach can be easily extended to continuous head pose estimation by using a suitable video database to train the system and a face tracking algorithm.

The developed system is suitable for hand held devices like tablet computers since it does not require much memory or processing power unlike model based systems.

References

1. Gee, A., Cipolla, R.: Fast visual tracking by temporal census. Image and Vision Computing 14(2), 105–114 (1996)
2. Horprasert, T., Yacoob, Y., Davis, L.: Computing 3-d head orientation from a monocular image sequence. In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, pp. 242–247 (1996)
3. Wang, J.-G., Sung, E.: Em enhancement of 3d head pose estimated by point at infinity. Image and Vision Computing 25(12), 1864–1874 (2007)
4. Niyogi, S., Freeman, W.: Example-based head tracking. In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, pp. 374–378 (1996)
5. Stiefelhagen, R.: Estimating head pose with neural networks results on the pointing04 icpr workshop evaluation data. In: Proceedings of ICPR Workshop Visual Observation of Deictic Gestures (2004)
6. Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation in computer vision: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4), 607–626 (2009)
7. Gourier, N., Hall, D., Crowley, J.L.: Estimating face orientation from robust detection of salient facial features. In: Proceedings of POINTING 2004 International Workshop on Visual Observation of Deictic Gestures (2004)
8. Head Pose Image Database, `http://www-prima.inrialpes.fr/perso/Gourier/Faces/HPDatabase.html`
9. Jain, V., Crowley, J.: Smile detection using multi-scale gaussian derivatives. In: Proceedings of the 12th WSEAS International Conference on Signal Processing, Robotics and Automation, pp. 149–154 (2013)
10. Lindeberg, T.: Scale-Space Theory in Computer Vision. Kluwer Academic Press (1994)
11. Lowe, D.G.: Object recognition from local scale-invariant features. In: International Conference on Computer Vision, pp. 1150–1157 (1999)
12. Freeman, W.T., Adelson, E.H.: The design and use of steerable filters. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(9), 891–906 (1991)
13. Crowley, J.L., Riff, O.: Springer Lecture Notes on Computer Science. In: Griffin, L.D., Lillholm, M. (eds.) Scale-Space 2003. LNCS, vol. 2695, pp. 584–598. Springer, Heidelberg (2003)
14. Jolliffe, I.T.: Principal Component Analysis. Springer (2002)
15. Vapnik, V.N.: Statistical Learning Theory. Wiley, NY (1998)
16. Cortes, C., Vapnik, V.N.: Support-Vector Networks. In: Machine Learning, vol. 20, pp. 273–297. Springer (1995)
17. Viola, P., Jones, M.J.: Robust real-time face detection. International Journal of Computer Vision 20(17), 137–154 (2004)
18. Sim, T., Baker, S., Bsat, M.: The cmu pose, illumination, and expression (pie) database. In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, pp. 46–51 (2002)