ESTIMATING HEAD POSE WITH AN RGBD SENSOR: A COMPARISON OF APPEARANCE-BASED AND POSE-BASED LOCAL SUBSPACE METHODS

Donghun Kim, Johnny Park, and Avinash C. Kak

Robot Vision Lab, School of Electrical and Computer Engineering, Purdue University,
West Lafayette, IN 47907
{zava, jpark, kak}@purdue.edu

ABSTRACT

Estimating the head pose with RGBD data when the pose is allowed to vary over a large angle remains challenging. In this paper, we show that an appearance-based construction of a set of locally optimum subspaces provides a good (fast and accurate) solution to the problem. At training time, our algorithm partitions the set of all images obtained by applying pose transformations to the 3D point cloud for a frontal view into appearance based clusters and represents each cluster with a local PCA space. Given a test RGBD images, we first find the appearance cluster that it belongs to and, subsequently, we find its pose from the training image that is closest to the test image in that cluster. Our paper compares the appearance-based local-subspace method with the pose-based local-subspace approach and with a PCA-based global subspace method. This comparison establishes the superiority of the appearance-based local-subspace approach.

Index Terms— 3D pose estimation, 3D head pose, view-based subspace model, RGBD Sensor

1. INTRODUCTION

The work we present in this paper is for estimating the head pose from RGBD data, specifically the data produced by the popular Kinect sensor. This sensor has become popular for computer vision research because it does a good job of producing co-registered range-reflectance data in real time. A number of recent publications are based on the data produced by this sensor [1, 2, 3]. The head-pose calculations in these contributions are based on first processing the depth information (which can be an expensive operation unto itself) for the extraction of a prominent landmark, such as the tip of the nose, and then orienting the rest of the data so that it is in the same configuration as the features on a human face.

The work we report in this paper presents an alternative framework that requires no specific landmarks to be extracted from the depth information. That, together with the use of a set of locally optimum subspaces, means that our algorithm can work much faster without any special hardware such as GPUs. Since our approach does not seek out a specific landmark, that allows our algorithm to work properly over a wider range of pose variations.

(a) (b) (c) (d) (e)

Fig. 1. Generated training samples: (a) Template, (b) $\Delta\theta_x = 30^\circ$, (c) $\Delta\theta_y = -30^\circ$, (d) $\Delta\theta_z = 30^\circ$, (e) $(\Delta\theta_x, \Delta\theta_y, \Delta\theta_z) = (30^\circ, -30^\circ, 30^\circ)$

The locally optimum subspaces in our framework are used to represent the appearance-based clusters of a large number of images constructed by applying pose transformations to a frontal-view RGBD image. Another way of creating locally optimum subspaces would be to cluster the images thus obtained over different ranges of the parameters in the pose space. We refer to the former as *appearance-based local-subspace* method and the latter as *pose-based local-subspace* method. Our results will demonstrate the superiority of the former. Our results will also demonstrate the superiority of the former over using a single global subspace, in the manner advocated by [4, 5], to represent all of the training images. We should also mention that the pose-based local-subspace method is similar to the contributions reported in [6, 7, 8, 9].

The remainder of this paper is organized as follows: Section 2 presents the details of our head-pose calculation framework: In Section 2.1, we start with an overview of the algorithm and describe its training and testing phases. Section 2.2 shows how we generate the training images at different poses from a single frontal view for a human subject. Section 2.3 presents the steps that go into, first, clustering of the training images, and, next, into the construction of a set of local subspaces from the clusters. Subsequently, Section 2.4 describes how the 3D pose of a query image can be estimated through hierarchical local subspace analysis. Experimental validation of the proposed framework is presented in Section 3.

To appear in *International Conference on Image Processing(ICIP), Sep. 15-18, 2013, Melbourne, Australia*

2. THE METHOD

2.1. Overview of the Algorithm

In this section we will provide a high-level summary of our algorithm for head pose estimation.

During the training phase, a single RGBD image of a head is recorded for the frontal pose followed by background subtraction. This RGBD data is transformed into a 3D RGB point cloud using the camera calibration parameters. We then apply a large number of pose transformations on the 3D point cloud and, for each transformation, a corresponding 2D image is calculated by projecting the point cloud onto the camera plane. In Section 2.2 we will point out how we deal with the problems caused by variable depth resolution of the RGBD sensor in this step.

The N number of training images obtained in this manner and their associated pose parameters can be represented by $\{(\mathbf{x}_1, \mathbf{p}_1), \cdots, (\mathbf{x}_N, \mathbf{p}_N)\}$. We can think of each of these synthetically generated images and its 3D pose as a training sample in a high n-dimensional space. Our next goal is to partition this data on the basis of the appearance of the images, then construct a locally optimum subspace for each cluster in the partition.

The training data is partitioned using a simple K-Means clustering algorithm, and the center and the covariance matrix of each of the resulting clusters are calculated. A locally optimum PCA subspace is then constructed for each cluster. Subsequently, the training images are projected into subspaces—the subspace for each image is chosen on the basis of minimum reconstruction error—as we describe in Section 2.3.

For a test RGBD image, we first find the best subspace to use on the basis of the minimum reconstruction error. We then search in that subspace for the training image that is closest to the test image. The 3D pose of the test image is then calculated from a lookup table using the nearest training image.

2.2. Generation of Training Samples from a Single RGBD Image

In this section, we will describe how the training images are created from a single RGBD image of a human subject during the training phase. Figure 1 shows some examples of such training images. This section will also discuss the construction of the 3D RGB point cloud model.

The 3D position (X, Y, Z) associated with an RGBD "pixel" at the raster coordinates (x, y) is calculated using the formulas:

$$X = \frac{Z_D}{f_c}(x - u_x),\ Y = -\frac{Z_D}{f_c}(y - u_y),\ Z = Z_D, \quad (1)$$

where Z_D is the depth value recorded by the sensor, f_c the focal length, and u_x and u_y the center coordinates of the image plane. Given 3D points obtained in this manner, we first carry out background subtraction by using GMM-EM (Gaussian Mixture Model with Expectation Maximization). To illustrate, we shown an RGBD image in Figure 2(a) and its associated depth map in (b). The 3D point cloud is simply a collection of 6-dimensional vectors of the form:

(a) (b) (c) (d)

Fig. 2. (a) A sample RGBD image; (b) Normalized depth image; (c) Image obtained after transformation of point cloud with $(\Delta\theta_x, \Delta\theta_y, \Delta\theta_z) = (-30^\circ, 25^\circ, -20^\circ)$ and its projection on the camera plane but with no interpolation; (d) Result obtained with 2.5D interpolation.

$$\mathcal{M} = \{X_{3D},\, V_{RGB}\}, \quad (2)$$

where X_{3D} contains the three spatial coordinates and V_{RGB} the three color values.

Given a single RGBD image of the frontal pose, we generate N training images by first applying N pose transformations to its point cloud obtained as described above, then projecting the resulting point clouds back on the camera image plane. For the pose parameter vector $\mathbf{p}$, this process is described by

$$I_t = \mathcal{T}\left(K\,[I \,|\, 0^T] G(\mathbf{p})\, X_{3D}\right), \quad (3)$$

where K is the intrinsic camera calibration matrix. The notation $\mathcal{T}(\cdot)$ stands for the conversion from the vectorized image with RGB values to the 2D image on the camera image plane. Finally $G(\cdot)$ is the 3D transformation including the translation parameters $t = [\ t_x \quad t_y \quad t_z\ ]^T$ and the Euler rotation matrix R computed from the rotation parameters $\theta = [\ \theta_{rx} \quad \theta_{ry} \quad \theta_{rz}\ ]$ as

$$G(\mathbf{p}) = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}, \quad (4)$$

where $\mathbf{p} = [\ \theta_{rx} \quad \theta_{ry} \quad \theta_{rz} \quad t_x \quad t_y \quad t_z\ ]$. When a pose-transformed point cloud is projected back onto the camera image plane, one often ends up with "holes" on account of the variable depth resolution of an RGBD sensor. This effect is shown in Figure 2(c). We get rid of such holes by applying bilinear interpolation to neighboring points using the constraint that the points used for bilinear interpolation possess roughly the same depth values. This technique provides both the depth value and the RGB data at what could otherwise be a hole. Figure 2(d) shows a projection when such interpolation is a part of the projection operator.

2.3. Appearance-Based Subspace Modeling

When training data is distributed in a complex manner (as

is the case when the data resides on a manifold) in whatever space one is using for its representation, using a single optimal subspace for all of the training data becomes infeasible. In such cases, one can take recourse to nonlinear approaches, such as those in [10, 11, 12], or use a set of locally optimum subspaces, as for example advocated in [13, 14, 15, 16]. We ourselves believe in the power of locally optimum subspaces. However, the challenge then is how to best build view-based local subspaces. Our own approach is along the lines of the work presented by Kambhatla and Leen [17].

Our interest lies two different ways to create locally optimum subspaces, the first we refer to as *pose-based local-subspace modeling*, as the second as *appearance-based local-subspace modeling*. Constructing a pose-based local-subspace model for the training data is straightforward and really needs no further elaboration. In pose-based local-subspace modeling, we simply create a K partition of the pose parameters and construct local subspace for all the training images in each partition.

In appearance-based local-subspace modeling, on the other hand, we obviously need a method to first cluster all of the training data with respect to the appearance of the images. For that purpose, we use Lloyd's K-Means algorithm to cluster all the training images into K disjoint clusters $\{\mathbf{R}^{(1)}, \cdots, \mathbf{R}^{(K)}\}$, with the corresponding subspaces being $\{\mathbf{S}^{(i)}, \cdots, \mathbf{S}^{K}\}$. The images in the individual clusters obey:

$$\mathbf{R}^{(i)} = \{\mathbf{x} \,|\, d(\mathbf{x}, \mathbf{S}^{(i)}) \leq d(\mathbf{x}, \mathbf{S}^{(j)}) \;:\; all\; j \neq i\}, \quad (5)$$

where the i^{th} subspace $\mathbf{S}^{(i)} =< \mathbf{r}^{(i)}, \mathbf{U}^{(i)} >$ denoted by the centroid $\mathbf{r}^{(i)}$ and a matrix $\mathbf{U}^{(i)}$ with eigenvectors of the covariance matrix $\mathbf{C}^{(i)}$, and $d(\mathbf{x}, \mathbf{S}^{(i)})$ is the reconstruction distance between a training image $\mathbf{x}$ and the i^{th} subsapce. This distance is given by

$$\begin{aligned} d(\mathbf{x}, \mathbf{S}^{(i)}) &= (\mathbf{x} - \mathbf{r}^{(i)})^T \mathbf{U}^{(i)T} \mathbf{U}^{(i)} (\mathbf{x} - \mathbf{r}^{(i)}) \quad (6) \\ &= (\mathbf{x} - \mathbf{r}^{(i)})^T \mathbf{P}^{(i)} (\mathbf{x} - \mathbf{r}^{(i)}). \quad (7) \end{aligned}$$

where $\mathbf{U}^{(i)}$ is the $m \times n$ matrix whose rows are the training eigenvectors of the covariance matrix $\mathbf{C}^{(i)}$. The projection matrix $\mathbf{P}^{(i)}$ is orthogonal to the local m dimensional subspaces.

According to the Lloyd Algorithm, the reference vectors $\mathbf{r}^{(i)}$ are to be placed at the generalized centroid of the region $\mathbf{R}^{(i)}$. The generalized centroid is defined by

$$\mathbf{r}^{(i)} = \arg\min_{r} \frac{1}{N_i} \sum_{\mathbf{x} \in \mathbf{R}^{(i)}} (\mathbf{x} - \mathbf{r})^T \mathbf{P}^{(i)} (\mathbf{x} - \mathbf{r}), \quad (8)$$

where N_i is the number of data samples in $\mathbf{R}^{(i)}$. As shown in [17], the generalized centroid $\mathbf{r}^{(i)}$ of region $\mathbf{R}^{(i)}$ is the same to the mean of data in $\mathbf{R}^{(i)}$.

We then compute the local covariance matrices

$$\mathbf{C}^{(i)} = \frac{1}{N_i} \sum_{\mathbf{x} \in \mathbf{R}^i} (\mathbf{x} - \mathbf{r}^{(i)})(\mathbf{x} - \mathbf{r}^{(i)})^T, \text{ for } i = 1, \cdots, K \quad (9)$$

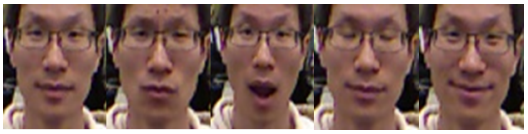

Fig. 3. Five templates used to generate testing samples

and build the subspaces spanned by their own eigenvectors $\mathbf{e}_j^{(i)}$ for $i = 1, \cdots, K$ and $j = 1, \cdots, m$.

Finally, the process of partitioning and finding the centroid is iterated until the relative change in the total reconstruction error is below some threshold. For fair comparison between subspaces, we used the same dimensionality m to calculate the reconstruction distance. The initial seeds for K-Means are supplied by sampling just the yaw parameter at K points. In order to place greater emphasis on the pose-space points that are at the two ends of the yaw-angle range, a sigmoidal function is used to distribute the sampling points nonlinearly.

2.4. 3D Pose Estimation by Local Subspace Analysis

In this section we describe how to estimate the 3D pose for a given query image using hierarchical local subspace analysis. With the set of local subspaces, $\{\mathbf{S}_1, \cdots, \mathbf{S}_K\}$, for a query image $\mathbf{x}$, we first find the best subspace to use by minimizing the reconstruction distance of Eq.6: as

$$q = \arg\min_{i} d(\mathbf{x}, \mathbf{S}^{(i)}), \text{ for } i = 1, \cdots, K. \quad (10)$$

Let the training samples $\mathbf{x}_i^{(q)}$ in the q^{th} subspace thus ascertained have their local-subspace representations given by the vectors $\mathbf{y}_i^{(q)} = \mathbf{U}^{(q)}(\mathbf{x}_i^{(q)} - \mathbf{r}^{(q)})$ for $i = 1, \cdots, N_q$ where N_q is the number of samples in the q^{th} subspace. Subsequently, we search in the local subspace for that training image which is closest to the query image $\mathbf{x}$. This is accomplished through the minimization:

$$l^* = \arg\min_{i} \left\| \mathbf{y}_i^{(q)} - \mathbf{U}^{(q)} \left(\mathbf{x} - \mathbf{r}^{(q)} \right) \right\|^2, \text{ for } i = 1, \cdots, N_q. \quad (11)$$

The 3D pose returned for the query image $\mathbf{x}$ is the pose associated with the nearest training sample image as represented in the local subspace through its coefficient vector $\mathbf{y}_{l^*}^{(q)}$.

3. EXPERIMENTAL VALIDATION

The protocol we have used for the experimental validation of the head pose calculation algorithm consists of the following steps:

- For the training phase, record a single frontal RGBD image of a human subject. Apply pose transformations to the 3D point cloud of the recorded image within the following angular ranges: [-75, 75] for yaw, [-90, 90]

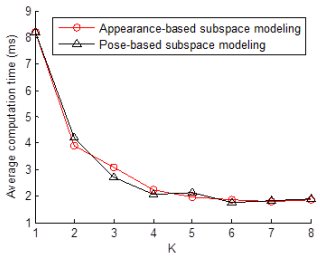

Fig. 4. Average computation time for the pose-based and the appearance-based subspace modeling methods. $K = 1$ corresponds to the global subspace approach and $K = 2, ..., 8$ the local subspace approach.

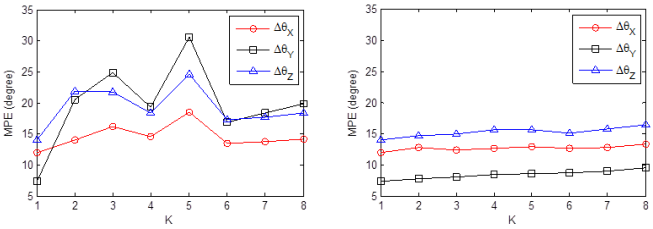

Fig. 5. The mean pose estimation error for: (a) Pose-based subspace modeling method; and (b) Appearance-based subspace modeling method.

for pitch, and [-75,75] for roll, at intervals of 15 degrees with respect to each of the three pose angles. This yields a total of 1573 training images with the size of 45 by 40 through just their intensity values. (n=1800)

- Construct a set of locally optimum subspaces from this training data in the manner described in Section 2 for different values of the user-specified "number of clusters" parameter K. **Note that when $K = 1$, we have a single global subspace to use for representing all the training images and for calculating the pose of a given test image.** Therefore, $K = 1$ corresponds to how the subspace method has traditionally been used in appearance-based object recognition. This brings to an end the training phase of the algorithm.
- For the testing phase, record a set of frontal images for the same human subject but with possibly different facial expressions. For the results we will report in this section, this set of frontal RGBD images is shown in Figure 3.
- For each of the five RGBD image in the test set shown in Figure 3, apply randomly selected pose transformations to its 3D point cloud to create 1000 images. This gives a total of 5000 images with random poses for testing the algorithm. It is important to note that, in addition to the pose parameters associated with these 5000 images being random, they were constructed from the five RGBD images recorded independently of the image used for generating the subspaces. Each testing image is also represented by a 45×40 matrix of gray-scale intensity values.
- Calculate the pose for each of the 5000 images in the test set using the local subspaces derived from the training data. Compute the error between the estimated pose and its ground-truth value.

To build the view-based local subspaces, we compared the conventional pose-based subspace model and our appearance-based subspace model on the computational time and the accuracy. Figure 4 illustrates the average elapsed time for the

Results	IS	Subspace (n= 1800, N_{Train}= 1573, N_{Test}= 5000)							
K	1	1	2	3	4	5	6	7	8
Dim.	1800	391	231	165	133	113	100	91	84
MPE (°)	11.2	11.1	11.7	11.7	12.2	12.3	12.1	12.5	13.0
SPE (°)	15.5	15.2	16.1	16.6	16.8	17.0	17.0	17.8	18.4
Time (ms)	24.8	8.18	3.90	3.08	2.24	1.98	1.85	1.81	1.86

Table 1. The experimental results of the appearance-based subspace modeling (IS - Image space)

appearance-based subspace modeling and the pose-based subspace modeling. As the number of subspaces increases, the search time is radically reduced—about a three-fold reduction at $K = 4$ for on both methods.

On the accuracy of estimating 3D poses, we evaluate the mean of average error on each pose angle. Figure 5(a) shows mean error for the pose-based subspace modeling method and Figure 5(b) the mean error for the appearance-based method. From these two graphs, one can first see that in general the appearance-based method performs better than the pose-based method. Furthermore, in the appearance-based method, the local subspace approach (i.e., $K = 2, ..., 8$) yields comparable pose estimation results compared to the global subspace approach (i.e., $K = 1$) while the computation speed is 3-4 times faster. Table 1 presents the local dimensionality, the average computation time, the mean pose estimation error (MPE), and the standard deviation of the pose estimation error (SPE) for different values of K which is the number of clusters for the partitioning of all of the images. The column whose heading is IS is for the case when all the images are used directly — without resorting to subspace representation.

4. CONCLUSION

Our results demonstrate the superiority of the appearance-based local-subspace modeling approach for head-pose estimation. Being hierarchical, it returns the head-pose value much faster than using a single globally optimum subspace for all of the training data. Future work is to build a more robust person-independant framework for the same.

5. REFERENCES

[1] G. Fanelli, T. Weise, J. Gall, and L. Van Gool, "Real time head pose estimation from consumer depth cameras," *Pattern Recognition*, pp. 101–110, 2011.

[2] P. Padeleris, X. Zabulis, and A. Argyros, "Head pose estimation on depth data based on particle swarm optimization," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 42–49.

[3] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool, "Random forests for real time 3d face analysis," *International Journal of Computer Vision*, pp. 1–22, 2012.

[4] H. Murase and S. Nayar, "Visual learning and recognition of 3-d objects from appearance," *International journal of computer vision*, vol. 14, no. 1, pp. 5–24, 1995.

[5] H. Borotschnig, L. Paletta, M. Prantl, A. Pinz *et al.*, "Active object recognition in parametric eigenspace," in *British Machine Vision Conference*, vol. 2, 1998, pp. 629–638.

[6] S. Srinivasan and K. Boyer, "Head pose estimation using view based eigenspaces," in *16th IEEE International Conference on Pattern Recognition. on*, vol. 4. IEEE, 2002, pp. 302–305.

[7] L. Morency, P. Sundberg, and T. Darrell, "Pose estimation using 3d view-based eigenspaces," in *IEEE International Workshop on Analysis and Modeling of Faces and Gestures*, 2003, pp. 45–52.

[8] G. Shakhnarovich and B. Moghaddam, "Face recognition in subspaces," *Handbook of Face Recognition*, pp. 19–49, 2011.

[9] S. Li, X. Peng, X. Hou, H. Zhang, and Q. Cheng, "Multi-view face pose estimation based on supervised isa learning," in *the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, 2002, pp. 100–105.

[10] B. Raytchev, I. Yoda, and K. Sakaue, "Head pose estimation by nonlinear manifold learning," in *the 17th International Conference on Pattern Recognition*, vol. 4, 2004, pp. 462–466.

[11] V. Balasubramanian, J. Ye, and S. Panchanathan, "Biased manifold embedding: A framework for person-independent head pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–7.

[12] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[13] Y. Hu and T. Huang, "Subspace learning for human head pose estimation," in *IEEE International Conference on Multimedia and Expo*, 2008, pp. 1585–1588.

[14] L. Morency, J. Whitehill, and J. Movellan, "Generalized adaptive view-based appearance model: Integrated framework for monocular head pose estimation," in *the 8th IEEE International Conference on Automatic Face & Gesture Recognition*, 2008, pp. 1–8.

[15] K.-C. Lee, J. Ho, M.-H. Yang, and D. Kriegman, "Video-based face recognition using probabilistic appearance manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2003, pp. 313–320.

[16] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1994, pp. 84–91.

[17] N. Kambhatla and T. Leen, "Dimension reduction by local principal component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1493–1516, 1997.