Pose Estimation using 3D View-Based Eigenspaces

Louis-Philippe Morency Patrik Sundberg Trevor Darrell

MIT Artificial Intelligence Laboratory
Cambridge, MA 02139

Abstract

In this paper we present a method for estimating the absolute pose of a rigid object based on intensity and depth view-based eigenspaces, built across multiple views of example objects of the same class. Given an initial frame of an object with unknown pose, we reconstruct a prior model for all views represented in the eigenspaces. For each new frame, we compute the pose-changes between every view of the reconstructed prior model and the new frame. The resulting pose-changes are then combined and used in a Kalman filter update. This approach for pose estimation is user-independent and the prior model can be initialized automatically from any view point of the view-based eigenspaces. To track more robustly over time, we present an extension of this pose estimation technique where we integrate our prior model approach with an adaptive differential tracker. We demonstrate the accuracy of our approach on face pose tracking using stereo cameras.

1. Introduction

Estimating the pose of a rigid object accurately and robustly for a wide range of motion is a classic problem in computer vision and has many useful applications. We are particularly interested in head pose tracking and its application in view-invariant face recognition, head gesture understanding, and conversational turn-taking cues.

In this paper we propose a method for estimating the absolute pose of an object from a known class, using intensity and depth view-based eigenspaces. Our approach consists of two steps: first we compute a prior model of the object given one initial frame and then this prior model is used to compute the absolute pose of each new frame. Here we focus our attention on human faces, although the methods are general enough to extend to many different classes of rigid objects. We built our depth and intensity view-based eigenspaces using Principal Component Analysis (PCA) for 28 different viewpoints surrounding the face of 14 people.

When presented with an intensity or depth image of a subject in an unknown pose, the system first finds the view with minimal reconstruction error, and then uses the corresponding PCA coefficients to reconstruct the image at all views. This is equivalent to finding the point on the multi-view depth and intensity manifold that most closely approximates the observed image at some view. The reconstructed 3D multi-view model is then used as a prior model for absolute-pose estimation. Rigid pose tracking is easiest when a 3-D shape and appearance model of the object is available.

Given the reconstructed prior model and a new frame showing the same subject, we estimate the new pose with a two step process. We first compute the relative pose between the new frame and each view in the prior model using an iterative view registration algorithm [13]. This computation uses intensity information as well as depth (if available), and amounts to estimating pose-change measurements between the new frame and every view in the prior model. As a final step, the pose measurements are integrated using a Kalman filter to produce a final estimate for the absolute pose [14]. This tracking framework efficiently computes the 6-DOF pose of the subject's head, and could be provided with 2D or stereo images as input, depending on the view registration algorithm used to do the relative pose computations.

As an extension of our approach, we integrated the reconstructed prior model in our existing Adaptive View-Based Appearance Model (AVAM) tracking framework [14]. This method creates a user-specific view-based model online during tracking. It can estimate the pose of an object accurately and with bounded drift, relative to the first frame. By integrating the prior model in the AVAM framework, we get a robust tracker able to initialize automatically and track object outside the pose space defined in our prior model.

Section 2 reviews previous work and how it relates to this paper, and Section 3 describes how we construct the view-based eigenspaces. We then in Section 4 present the algorithm to create a prior model of the person of interest given the initial image of an image sequence. Section 5 presents our technique for 6-DOF pose estimation using the view registration and Kalman filter framework. Section 5.1 describes the integrated framework with AVAM. Finally, in Section 6, we show results for a head tracking task with depth and intensity input from a commercial stereo camera,

and compare the accuracy of our pose estimation technique with that of another technique[18].

2. Previous Work

Pose estimation is possible from a single 2-D view–e.g, using color and coarse template matching [2, 16], pattern classifiers [15], or using graph matching techniques [11]–but techniques which can exploit a 3-D model are generally more accurate. 3-D representations model the appearance of objects more closely, and thus can lock on to a subject more tightly. Textured geometric 3D models [10, 1] have been used for tracking; because the prior 3D shape models for these systems do not adapt to the user, they tend to have limited tracking range.

Deformable 3D models fix this problem by adapting the shape of the model to the subject [9, 12, 4, 6]. These approaches maintain the 3D structure of the subject in a state vector which is updated as images are observed. These updates require that correspondences between features in the model and features in the image be known. Reliably computing these correspondences is difficult, and the complexity of the update grows quadratically with the number of 3D features, making the updates expensive [9]. Brand developed a 3-D morphable model which is able to track features while simultaneously estimating the underlying shape model [3]. In general, existing approaches to 3-D model estimation for tracking presume a single-viewpoint model of image appearance, which will not be valid for non-lambertian objects.

A view-based approach to 3-D modeling and tracking has several advantages over mesh or volumetric shape models. The relative pose of constituent range observations can be adjusted dynamically during model formation. It can easily represent varying levels of detail on an object, and it directly captures non-lambertian appearance on the surface of an object [14].

Below, we describe a pose estimation algorithm using view-based eigenspaces with depth and intensity components. View-based models for object recognition using eigenspaces were described in [17], which constructed a separate PCA model for sets of images at given views. [5] developed a multi-view active appearance model that described shape and texture variation across views; object appearance was matched using the closest view and pose inferred with a linear projection of model coefficients.

Recently the reconstruction distance to a set of eigenspaces at different views was used to interpolate head pose; the relationship between a set of approximate correlation scores and object pose can be learned from training examples [18]. Our approach differs from this work in that we learn a joint eigenspace across views, intensity, and depth images, and that we use the model only to reconstruct a multi-view model close to the observed object. We do not use the correlation scores or reconstruction error from each view to infer pose; instead we compute the pose-changes between every view of the reconstructed prior model and the new frame. The resulting pose-changes are then combined and used in a Kalman filter update.

The Adaptive View-based Appearence Model described in [14] is a relative-pose tracker which combines differential tracking with keyframe-based tracking. Adaptive view-based models can be acquired online during the tracking. The uncertainty in the pose of a keyframe shrinks overtime as the keyframe is revisited. Hence, the uncertainty in the pose estimate of new frame registered against a keyframe is bounded. The resulting tracker has bounded drift and can be used to track heads undergoing large motion for a long time. By creating the adaptive appearance model online, the tracker gives accurate relative pose but doesn't have any mechanism to estimate absolute pose [14].

3. 3D View-based Eigenspaces

We wish to learn a multi-view depth and intensity model which is user-independent and can be used to initialize pose tracking . We also want a view-based model that can reconstruct a multi-view manifold of the object given only one view. Ideally, we could recreate the depth manifold given only an intensity image as input.

To achieve these goals, we define our view-based eigenspaces model $\mathcal{P}$ as:

$$\mathcal{P} = \{\bar{I}, \mathcal{V}_I, \bar{Z}, \mathcal{V}_Z\}$$

where $\bar{I}$ and $\bar{Z}$ are the mean intensity and depth for all the views; $\mathcal{V}_I$ and $\mathcal{V}_Z$ are the intensity and depth eigenspaces of our model. To navigate in our model, we define windows $\mathcal{P}_i$ in the eigenvector matrices for each view i of our model:

$$\mathcal{P}_i = \{\bar{I}_i, V_{I_i}, \bar{Z}_i, V_{Z_i}, \varepsilon_i\}$$

where $\bar{I}_i$ and $\bar{Z}_i$ are the mean intensity and depth images for this view, ε_i is the pose of that view and V_{I_i} and V_{Z_i} are windows in the eigenspace matrices. Note that V_{I_i} and V_{Z_i} are not eigenspaces since we defined our eigenspaces $\mathcal{V}_I$ and $\mathcal{V}_Z$ over all the views. In our case, poses of rigid body are represented as $\varepsilon = [\ T^x \quad T^y \quad T^z \quad \Omega^x \quad \Omega^y \quad \Omega^z\ ]$, a 6 dimensional vector consisting of the translation and the instantaneous rotation.

3.1. Eigenspaces Acquisition

We want to generate a user-independent view-based model that can render every view of the object given a correct match with one of the views. The view-based eigenspaces $\mathcal{P}$ can be learned from multiple adaptive view-based appearance models $\{\mathcal{M}^1, \mathcal{M}^2, ..., \mathcal{M}^n\}$. Following the definition

stated in [14], an adaptive view-based appearance model is defined as

$$\mathcal{M} = \{\{I_i, Z_i, \varepsilon_i\}, \Lambda\}$$

where $\{I_i, Z_i\}$ are the intensity and depth images at each view i, ε_i is the pose of each key frame modelled with a Gaussian distribution, and Λ is the covariance matrix over all random variables ε_i.

For each model $\mathcal{M}^j$, we concatenate the intensity and depth images of all views in two vectors I^j and Z^j:

$$\begin{aligned} I^j &= \left[\begin{array}{cccc} I_1^j & I_2^j & \cdots & I_m^j \end{array}\right] \\ Z^j &= \left[\begin{array}{cccc} Z_1^j & Z_2^j & \cdots & Z_m^j \end{array}\right] \end{aligned}$$

where I_i^j and Z_i^j are segmented intensity and depth images from the appearance model $\mathcal{M}^j$ at pose ε_i, and m is the number of views. All the segmented images have the same size and are stored in one-dimensional vectors. We can compute the average vectors:

$$\bar{I} = \tfrac{1}{n}\textstyle\sum_{j=1}^{n} I^j \quad \bar{Z} = \tfrac{1}{n}\textstyle\sum_{j=1}^{n} Z^j \tag{1}$$

and then stack all the normalized intensity and depth vectors into two matrices:

$$\begin{aligned} \mathcal{I} &= \left[\begin{array}{ccc} \left(I^1 - \bar{I}\right) & \left(I^2 - \bar{I}\right) & \cdots \end{array}\right]^T \\ \mathcal{Z} &= \left[\begin{array}{ccc} \left(Z^1 - \bar{Z}\right) & \left(Z^2 - \bar{I}\right) & \cdots \end{array}\right]^T \end{aligned}$$

Since we want to be able to reconstruct the intensity and depth images from only one intensity image, we must use the same set of weights for the intensity and the depth eigenvectors. To achieve that, we apply SVD decomposition on $\mathcal{I} = U_{\mathcal{I}} D_{\mathcal{I}} \mathcal{V}_{\mathcal{I}}^T$ and compute the corresponding depth eigenvectors by applying the same weights:

$$\mathcal{V}_{\mathcal{Z}}^T = D_{\mathcal{I}}^{-1} U_{\mathcal{I}}^{-1} \mathcal{Z}$$

This approach allows us to create a prior model with both intensity and depth even when no stereo information is available. Although the resulting depth basis vectors of $\mathcal{V}_{\mathcal{Z}}$ are not optimal, there is a strong correlation between the intensity and depth images that provides justification for this approach.

Figure 1 shows the mean face of our view-based eigenspaces built using adaptive view-based models of 14 people. Each adaptive model contains 28 views of one person: 7 views along the X axis by 4 views along the Y axis. All adjacent views are separated by 10°. By looking closely at the depth images, we can see that the chin is closer when looking at 20° up. When looking on the side, we can see a small bump representing the nose. Such subtle details can be important during tracking.

Figure 2 shows the first three intensity eigenvectors displayed for the 7 horizontal views. We can see in the second eigenvector the variations for the nose and the eye shadow. The third eigenvector presents some lip variation.

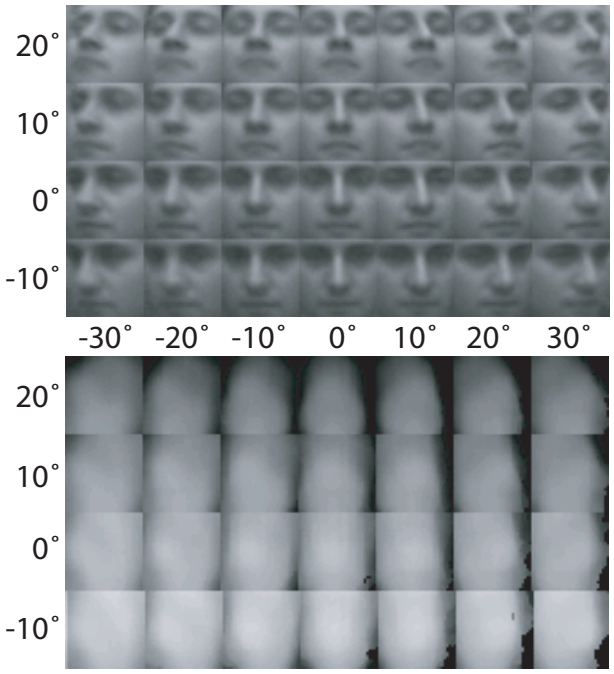

Figure 1: Top: 28 views of the average intensity manifold. Bottom: 28 views of the average depth manifold (white means closer).

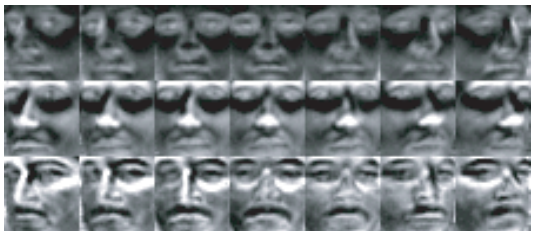

Figure 2: The first three intensity eigenvectors (rows) partially displayed for the 7 horizontal views (colomns).

4. Prior Model Reconstruction

The purpose of prior model reconstruction is to generate a set of views for use in the pose estimation module. Given a single example frame near one of the views in our model $\mathcal{P}$, we want to reconstruct all the other available views of our model. In our case, given one image, we can recreate the 27 other views including the depth images.

The new unsegmented frame $\{I_t, Z_t\}$ is preprocessed to find a region of interest for the object. This can be done using motion detection, background subtraction, flesh color detection or a simple face detector [19].

For each view i of our model and for each subregion $\{I'_t, Z'_t\}$ of the same size as the views in $\mathcal{P}$ inside the region of interest, we find the vector $\vec{w}_i$ that minimizes

$$E_i = |I'_t - \bar{I}_i - \vec{w}_i \cdot \mathcal{V}_{I_i}|^2, \tag{2}$$

The minimization is straightforward using linear least squares.

From the eigenvector weights $\vec{w}_i$, we first reconstruct the intensity and depth images

$$I_{R_i} = \bar{I}_i + \vec{w}_i \cdot \mathcal{V}_{I_i} \quad (3)$$

$$Z_{R_i} = \bar{Z}_i + \vec{w}_i \cdot \mathcal{V}_{Z_i} \quad (4)$$

The reprojection step is done for every view i. After the reconstruction of all intensity and depth images I_{R_i} and Z_{R_i}, we search for the best projection minimizing the correlation function:

$$\frac{(I'_t - \bar{I}'_t) \cdot (I_{R_i} - \bar{I}_{R_i})}{|I'_t - \bar{I}'_t||I_{R_i} - \bar{I}_{R_i}|} + \lambda \frac{(Z'_t - \bar{Z}'_t) \cdot (Z_{R_i} - \bar{Z}_{R_i})}{|Z'_t - \bar{Z}'_t||Z_{R_i} - \bar{Z}_{R_i}|}$$

where λ is constant to compensate for the difference between intensity measurements(brightness levels) and the depth measurements(mm). If the depth image is not available then λ is set to 0.

From the correlation function, we get a correlation score c_i for each view and each subregion $\{I'_t, Z'_t\}$. The lowest correlation c_i^* over all views and all subregions corresponds to the best match. Using the weights $\vec{w}^*$ of the best match, we again reconstruct the intensity and depth images of the object in all the views using eq. (3) and (4). The output of the matching algorithm is the set of reconstructed frames $\{I_i^*, Z_i^*\}$, the pose ε_i^* of the best view-based eigenspace and the associated correlation score c_i^*. We can define a prior view-based appearance model

$$\mathcal{M}_P = \{\{I_{P_i}, Z_{P_i}, \varepsilon_{P_i}\}, \Lambda_P\}$$

where the images and poses are copied directly from reconstructed frames and associated poses, and the covariance matrix Λ_P is initialized as the identity matrix times a small constant.

Figure 3 shows reconstructions for 2 different people. Each reconstruction was done using view-based eigenspaces that exclude the person reconstructed. The top half shows a reconstruction where the example image is oriented near the view 0° around X axis and 0° around the Y axis. The reconstructed views displayed in the figure are the horizontal view. The second reconstruction uses an example image at 20° around X axis and 20° around the Y axis. The reconstructed views are at -10° around the Y axis.

Figure 3: Model reconstruction from a frontal view (top half) and a rotated view (bottom half). Both reconstructions (bottom rows) are compared with ground truth (top rows).

5. 6-DOF Absolute Pose Estimation

In this section we present our technique to estimate the absolute pose of a rigid object using 3D view-based eigenspaces as a prior model. Figure 4 presents an overview of our pose estimation algorithm. Our approach is separated in two steps: first we compute a prior model of the subject given one initial frame and then this prior model is used with each new frame to compute the absolute pose.

During the initialization stage, each new frame $\{I_t, Z_t\}$ is projected into the view-based eigenspace model as described in section 4. If the best correlation score c_i^* is larger than a threshold k, the prior model is created.

When depth information is available in the new frame, we can register the frames using a hybrid error function which combines robustness of the ICP (Iterative Closest Point) algorithm and the precision of the normal flow constraint (NFC) [13]. When only 2D images are available an iterative approach like [6] may be used to give an appropriate set of pose-change measurement. We model a pose-change measurement δ_s^t as having come from $\delta_s^t = \varepsilon_t - \varepsilon_s + \omega$ where ε_s is the pose estimate associated with the view s of our prior model and ω is Gaussian.

To estimate the pose ε_t of the new frame based on the pose-change measurements, we use the Kalman filter formulation described in [14] where the state vector $\mathcal{X}$ is populated with the pose variables $\{\varepsilon_t, \varepsilon_{\mathcal{P}_1}, \varepsilon_{\mathcal{P}_2}, \ldots\}$ and the observation vector $\mathcal{Y}$ is populated with the pose-change measurements $\{\delta_{P_1}^t, \delta_{P_2}^t, \ldots\}$. The covariance between the components of $\mathcal{X}$ is denoted by $\Lambda_{\mathcal{X}}$.

The Kalman filter update computes a prior for $p(\mathcal{X}_t|\mathcal{Y}_{1..t-1})$ by propagating $p(\mathcal{X}_{t-1}|\mathcal{Y}_{1..t-1})$ one step forward using a dynamic model. Each pose-change measurement $y_s^t \in \mathcal{Y}$ between the current frame and a base frame of $\mathcal{X}$ is modelled as having come from:

$$y_s^t = C\mathcal{X} + \omega,$$

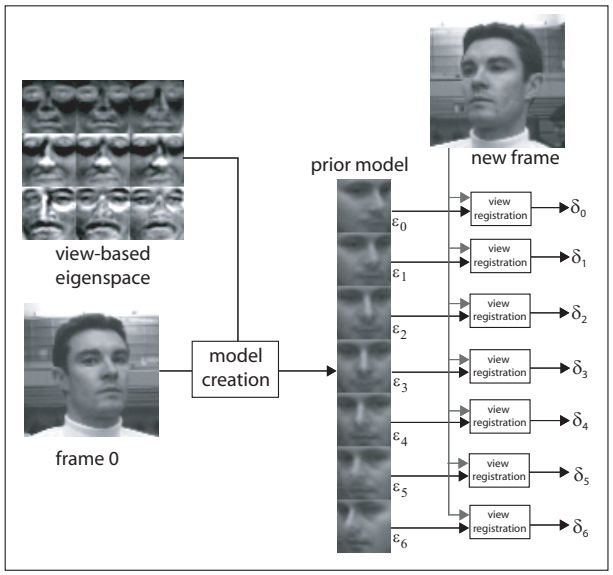

Figure 4: Overview of the prior model and its usage. Section 3 describes how the view-based eigenspaces are created, Section 4 describes the model creation step that is done once at the beginning of each sequence, and section 5 describes view registration.

$$C \;=\; \left[\; I \quad 0 \quad \cdots \quad -I \quad \cdots \quad 0 \;\right],$$

where ω is Gaussian. Each pose-change measurement y_s^t is used to update all poses using the Kalman Filter state update:

$$[\Lambda_{\mathcal{X}_t}]^{-1} = \left[\Lambda_{\mathcal{X}_{t-1}}\right]^{-1} + C^\top \Lambda_{y_s^t}^{-1} C \tag{5}$$

$$\mathcal{X}_t \;=\; \Lambda_{\mathcal{X}_t}\left(\left[\Lambda_{\mathcal{X}_{t-1}}\right]^{-1}\mathcal{X}_{t-1} + C^\top \Lambda_{y_s^t}^{-1} y_s^t\right) \tag{6}$$

We define our observations variables δ_s^t as a pose-change measurement between the new frame and a base frame in $\mathcal{X}$.

5.1. Integration with AVAM

In this section we present an extension of the 6-DOF absolute pose estimator where we integrate the reconstructed prior model inside an Adaptive View-based Appearance Model (AVAM) tracking framwork. In the AVAM framework, user-specific keyframes are added in the model during the tracking. One of the main advantages of the AVAM framework is that pose estimation of the new frame $\{I_t, Z_t\}$ and pose adjustments of the view-based model $\mathcal{M}$ are performed simultaneously. The original AVAM described in [14] is a relative-pose tracker which combines differential tracking with keyframe-based tracking. By integrating the prior model with the AVAM framework, we obtain an absolute-pose tracker with accurate pose estimates and bounded-drift.

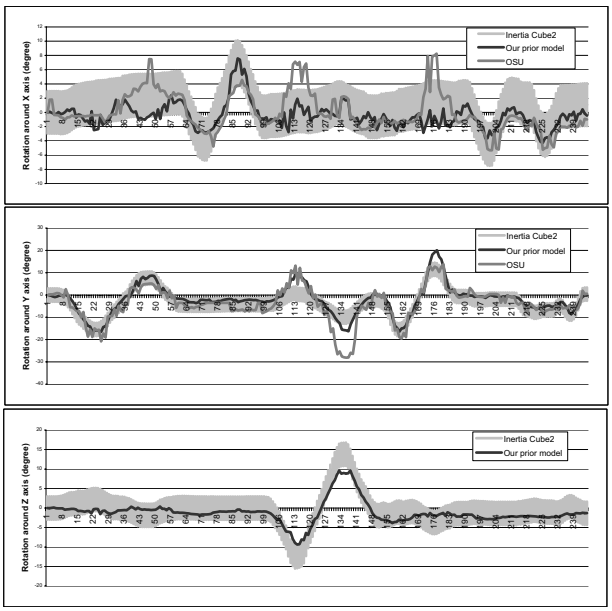

Figure 5: Comparison of the pose estimation results of the best reconstruction (4 eigenvectors) with the InertiaCube^2 and with our reimplementation of the OSU pose estimator. The ground truth estimates from the Inertial Cube^2 are shown with 3° error bars.

Since the AVAM framework also uses a Kalman filter to update the poses, we can extend the Kalman filter state vector to include the key frame pose variables and the previous frame pose variable:

$$\mathcal{X} = \left[\; \varepsilon_t \quad \varepsilon_{t-1} \quad \varepsilon_{\mathcal{M}_1} \quad \varepsilon_{\mathcal{M}_2} \quad \cdots \quad \varepsilon_{\mathcal{P}_1} \quad \varepsilon_{\mathcal{P}_2} \quad \cdots \;\right]^\top$$

By integrating the prior model in the AVAM framework, we get a robust tracker able to initialize automatically and estimate absolute pose of the object.

6. Experiments

We designed our experiments to demonstrate the accuracy of our approach on estimating the relative and absolute pose of a user's head in 3D. All the experiments were done using a Videre Design stereo camera [7].

We constructed the view-based eigenspaces using 14 view-based appearance models acquired with the original tracker described in [14]. Each participant was aligned facing the camera, and then asked to rotate his head along the X and Y axes. We configured the tracker to tesselate the rotation space at 10 degree intervals. In this fashion, a set of 28 intensity and depth image pairs was created for each participant. We then manually cropped the faces to 32x32 pixels, while keeping alignment across participants. Figure 1 shows the average face in all 28 orientations, and Figure 2 shows 3 of the 13 eigenvectors that we use to create prior models.

To analyze our algorithm quantitatively, we compared the pose estimates of our system with those of an InterSense InertiaCube^2 sensor [8]. The InertiaCube^2 is an inertial 3-DOF orientation tracking system, which we mounted on the inside of a construction hat that was worn by a test subject during tracking. The sensor works by measuring the direction of gravity and the Earth's magnetic field, and is driftless along the X and Z axes. However, the Y axis (pointing up) can suffer from errors due to drifting. InterSense reports a dynamic accuracy of 3°RMS.

6.1. 6-DOF Absolute Pose Detection

To analyze the accuracy of our view-based eigenspace model for pose estimation, we recorded 2 sequences with ground truth poses using an InertiaCube^2 sensor. Sequence 1 contains 245 frames and sequence 2 contains 155 frames. Because the purpose of the experiment was to evaluate the accuracy of the pose estimation algorithms, poses in both sequences are constrained to the rotation space of the prior model. The pose estimation algorithm described in section 5.1 is applied independently for each frame.

For comparison, we also reimplemented a 2-dimensional version of the OSU system [18]. Our implementation of the OSU estimator used 7 different eigenspaces in the horizontal direction, at poses between -30 and 30 degrees with equal spacing, and 4 different eigenspaces in the vertical direction, between -20 and 10 degrees. This is the same range spanned by the eigenspaces we used in the other part of the paper. However, using this method pose estimation is done independently for the two degrees of freedom.

In the OSU pose estimation framework, the first step is to find the face in the image. This is accomplished by computing correlation scores with the mean face for each of the orientations in the model for every possible location in a large region of interest around the face. We then normalize the detected face to have zero mean and unit variance, and project it onto each eigenspace. The eigenspace that can represent the largest fraction of the energy of the input face determines a first coarse estimate of the pose. Finally, an incremental estimate of the pose is computed. The incremental estimates for the horizontal and vertical directions are given by

$$\Delta\theta_h = \frac{r_{h,23} - 0.989}{0.073} \tag{7}$$

$$\Delta\theta_v = \frac{r_{v,23} - 0.945}{0.066}, \tag{8}$$

where $r_{h,23}$ and $r_{v,23}$ are the ratios of the energy captured by the second and third best candidates for the coarse pose estimation in the horizontal and vertical directions, respectively. The numerical constants were computed using linear least squares from a training sequence tagged with ground truth. The incremental estimate is always taken to be in the direction of the second best coarse pose estimate, and is capped at 5.0 degrees.

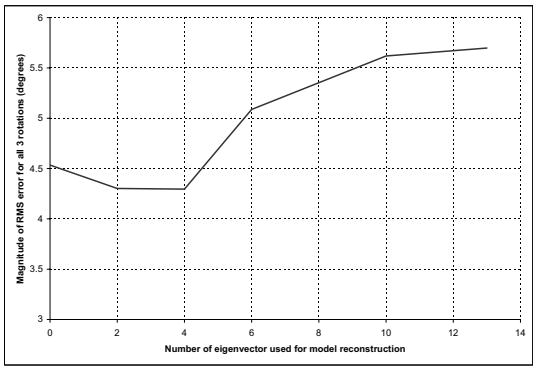

Figure 6: Variation of the RMS error of pose estimation on sequence 1 as we change the number of eigenvectors used to create the prior model.

Our prior model	**X**	**Y**	**Z**
Sequence 1	1.62°	2.55°	1.67°
Sequence 2	1.04°	3.91°	1.44°
OSU	**X**	**Y**	**Z**
Sequence 1	2.30°	4.46°	
Sequence 2	1.74°	3.01°	

Table 1: RMS error in degrees for pose estimation comparing our prior model pose estimation (top) and the OSU pose estimation(bottom). The OSU pose estimator doesn't return any estimate for the rotation around the Z axis.

Figure 5 shows the pose variations recorded from the ground truth sensor compared with the OSU pose estimator and our prior model pose estimator. Figure 6 shows the magnitude of the RMS error for the 3 rotations when varying the number of eigenvectors used for the reconstruction of the prior model. We found that, in our case, 4 eigenvectors was a good trade-off between model expressiveness and model over-fitting.

Table 1 presents a comparison of the RMS error for the pose estimation using our prior model and the OSU pose estimator. The InertiaCube^2 is not drift-free around the Y axis. The average RMS error of the prior model pose estimator for all 3 axis is 3.88° which is close to the accuracy of the InertiaCube sensor (3°). The OSU pose estimator doesn't model rotations around the Z axis hence gives no estimates. If we fix the OSU Z-axis output to 0, the RMS error around the Z axis is approximately 3.9° for both sequences. Note that our prior model pose estimator is able to handle rotation around the Z axis even though the training data for our prior model did not include frames with rotation around the Z axis.

6.2. Integrated AVAM & Prior Model

One of the main advantages of the user-independent view-based eigenspace model when inserted in our Kalman filter update is that we can estimate absolute poses. This is a considerable advantage compared to tracking techniques are initialized manually with ad-hoc techniques.

To demonstrate the performance of the pose estimator, we recorded a sequence of approximately 2 mins at 7Hz for a total of 800 frames. The user moved freely from left to right, front to back, rotated his head left and right, top to bottom, and also tilted his head. The purpose of this sequence is to show how our integrated technique can give an accurate estimate of the absolute pose in an unconstrained environment. In the video sequence, which can be found at http://www.ai.mit.edu/projects/watson/, we represent the estimate of the absolute pose by a cube around the head of the user. The thickness of the cube is inversely proportional to the variance of the absolute pose estimate. The red squares below the cube represent the number of base frames used to compute the estimate. This video shows the results of our approach which integrates differential tracking, adaptive view-based appearance model and view-based eigenspace prior model.

Table 2 presents the tracking results for different configurations of the tracker. The first row represents results using only the differential tracker. Differential tracking drifts after a certain amount of time, leading to high RMS error. The second row represents the pose estimation algorithm described in Section 5. Since the movements in this sequence were not constrained to the rotation space of our prior model, the performance of this approach is poor. The third row shows the performance of the original differential tracker and adaptive model. This tracking technique has shown to yield good results when estimating relative pose but without a good prior model this technique does poorly in terms of absolute pose estimation. The fourth row presents the results of the differential tracker with the prior model pose estimator. This gives better results then the prior model alone since the differential tracker can give a good pose estimate even outside the rotation space of the prior model. Also, the differential tracker acts as a dynamic model in the Kalman filter which helps to smooth the estimates of the prior model pose estimator. Finally, the last row presents the results of the integrated pose estimation and tracking. In that configuration, the prior model estimates the absolute pose during the initialization period which helps the accuracy of the online adaptive appearance model.

Technique	**X**	**Y**	**Z**	**Total**
Diff	10.80°	21.96°	14.74°	28.56°
Prior	5.12°	10.26°	3.47°	11.98°
Diff+Adapt	6.48°	8.15°	2.23°	10.65°
Diff+Prior	2.64°	5.39°	2.43°	6.48°
Diff+Adapt+Prior	2.46°	5.00°	2.46°	6.09°

Table 2: RMS error for each tracking technique.

7. Summary and Conclusions

We described a new technique for pose estimation based on view-based models. View-based models capture a rich representation of object shape and surface appearance across a wide range of pose change. We learned a multi-view, depth and intensity eigenspace model which provides a set of prior keyframes for pose estimation and tracking customized to a new user. A Kalman filter framework combines pose estimates using both prior and adaptive keyframes according to estimates of uncertainty for each. The use of our pose estimation technique greatly reduces the absolute error in view-based tracking, which was previously limited by the coarse pose estimate of generic face detectors used for initialization. We demonstrate the accuracy of our integrated approach on face pose tracking using stereo cameras.

References

[1] S. Basu, I.A. Essa, and A.P. Pentland. Motion regularization for model-based head tracking. In *Proceedings. International Conference on Pattern Recognition*, 1996.

[2] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 232–237, 1998.

[3] M. Brand and R. Bhotika. Flexible flow for 3d nonrigid tracking and shape recovery. In *Proc. Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 315–322, 2001.

[4] A. Chiuso, P. Favaro, H. Jin, and S. Soatto. Structure from motion causally integrated over time. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(4):523–535, April 2002.

[5] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(6):681–684, June 2001.

[6] D. DeCarlo and D. Metaxas. Adjusting shape parameters using model-based optical flow residuals. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(6):814–823, June 2002.

[7] Videre Design. *MEGA-D Megapixel Digital Stereo Head*. http://www.ai.sri.com/ konolige/svs/, 2000.

[8] InterSense Inc. *InertiaCube*^{2}. http://www.intersense.com.

[9] T. Jebara and A. Pentland. Parametrized structure from motion for 3D adaptive feedback tracking of faces. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[10] M. LaCascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on

registration of textured-mapped 3D models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(4):322–336, April 2000.

[11] M. Lades, J.C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R.P. Wurtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on Computers*, 42(3):300–311, March 1993.

[12] P. F. McLauchlan. A batch/recursive algorithm for 3D scene reconstruction. *Conf. Computer Vision and Pattern Recognition*, 2:738–743, 2000.

[13] L.-P. Morency and T. Darrell. Stereo tracking using icp and normal flow. In *Proceedings International Conference on Pattern Recognition*, 2002.

[14] L.-P. Morency, A. Rahimi, and T. Darrell. Adaptive view-based appearance model. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 2003.

[15] J. Ng and S. Gong. Composite support vector machines for detection of faces across views and pose estimation. *Image and Vision Computing*, 20:359–368, 2002.

[16] N. Oliver, A. Pentland, and F. Berard. Lafter: Lips and face real time tracker. In *Computer Vision and Patt. Recog.*, 1997.

[17] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Seattle, WA, June 1994.

[18] S. Srinivasan and K.L. Boyer. Head pose estimation using view based eigenspaces. In *Proceedings. 16th International Conference on Pattern Recognition*, pages 302–305, 2002.

[19] P. Viola and M. Jones. Robust real-time face detection. In *Proceedings International Conference on Computer Vision*, page II: 747, 2001.