Model Based Facial Pose Tracking Using a Particle Filter

Bogdan Kwolek
Rzeszów University of Technology
35-959 Rzeszów, Poland
bkwolek@prz.rzeszow.pl

Abstract

This paper presents a model-based technique for monocular tracking of the head pose using a non-calibrated camera. We use texture-mapped face images through the 3D head model as the data representation. The mapped data are compared to the model data via a similarity metric that expresses the likeness between the rendered and the reference images. The tracking is realized using a particle filter. In observation model we utilize rectangle features as the primary cue. The potential of our approach is demonstrated by tracking of the head pose on real videos.

1. Introduction

Accurate and reliable tracking of the three-dimensional head pose is an important problem. The usage of a three-dimensional head model provides the tracker with knowledge about the subject's structure, motions and appearance. Although there exist several methods to perform 3D head tracking [2][4][12][13][17], there is still a need to improve the accuracy and reliability of such systems.

A method developed by Horprasert [10] estimates the head pose through tracking only five salient facial points like eye corners and the nose top. In order to determine the head pose, Feng *et al*. [7] normalizes the head to a frontal pose and then tests it against the repository of frontal faces. The system developed by La Cascia *et al*. [13] works through mapping the face onto a cylindrical model and estimating the change in pose with regard to the incremental discrepancy in the re-rendered texture maps. Another example of 3D face modeling through parameterized meshes provides Ahlberg [1]. Basu *et al*. [2] uses optic flow to constrain the motion of non-rigid surface model. Rigid motion parameters of an ellipsoid model are estimated from the flow field using a standard optimization algorithm. The angular errors in this method could be as high as 20 degrees. Blanz *et al*. [3] developed a 3D morphable model, described with a linear combination of the texture and shape of multiple exemplars. This model could be fitted to a single image to obtain individual descriptors. Vacchetti *et al*. [14] utilizes key frames in a model based tracker to eliminate the jitter. Xiao *et al*. [16] demonstrates fitting both a 2D eigenspace appearance and 3D model. Darell *et al*. [4] employs a template-based method to estimate the head pose. The system uses the estimated head location to acquire an image of higher resolution, using a second active camera. This narrow field-of-view camera provides images with a resolution suitable for eigenspace-based pose estimation. The template-based methods require the presence of the same pixels over the entire image sequence. This limits the range of head poses which can be tracked.

This work is motivated by our desire to preserve eye contact in teleconferencing applications. We define tracking as the problem of determining the model parameters so that the tracked face re-projected to frontal pose best matches the reference face. The perspective projection of the 3D head representation onto the 2D viewing plane is used to generate the faces in the frontal pose. A texture taken from the considered face candidate is employed in rendering the face. Each sample of the particle filter represents the state of the potential face. The observation model of the particle filter utilizes the color rectangle features as the primary cue. The use of eigenspace-based face representation in such a tracking scheme is demonstrated too.

The entire tracking process is automatic, except for the initialization stage, which requires the person operating the system to manually align in the image plane a camera projection of the head to be tracked with the projection of the mesh representing the head model. Prior to the matching the subject is required to remain relative still in a neutral expression in order to shot an image with a frontal view of the face. Next, a locking of the facial appearance into the texture map takes place. Before the tracking the person observing the screen should align approximately his/her face onto the displayed mesh.

In the next section we describe our 3D head model. The components and details of image processing in model-based tracking are discussed in section 3. We then describe particle filtering. We present and discuss the experimental results in section 5. We draw conclusions in the last section.

2. The 3D head model

In order to determine the head pose, a model that incorporates knowledge about head structure, facial deformations, motions and appearance should be used.

The face model we utilize is a triangular mesh consisting $T = 128$ triangles, shown smoothly shaded in Fig. 1a. Such models are widely used in computer graphics because of acceleration from the modern graphics hardware. The model consists of a collection of $K = 71$ vertices $\mathbf{v}_0, ..., \mathbf{v}_{K-1}$ and a single texture image I. The texture image consists of pixels in the RGB space. The 3D triangles are defined by triples of vertex indices. Each vertex has a corresponding coordinate in the texture image. This means that a given vertex should always correspond to the same facial region. Each point in the reference image is associated with a point of the 3D mesh model. Therefore each triangular face of the 3D model has a corresponding 2D triangle in the reference image, see Fig. 1b. The transformation of the 3D representation of the head onto the 2D viewing plane of the tracking sequence is accomplished through a perspective projection. Under perspective projection the point $\mathbf{x} = \{x, y, z\}$ projects to the image point $\mathbf{x}_p = \frac{f}{z}\{x, y\}$, where f denotes the focal length.

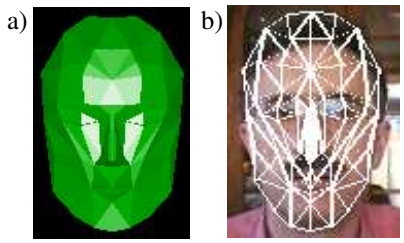

Figure 1. The face model in a default configuration (a), the wireframe fitted to the face (b).

Combining the model of the shape and model of the textures results in a 3D face representation which can be rendered. This allows us to generate images depicting faces in the requested pose and therefore to evaluate quickly the quality of aligning the face to the mesh, see Fig. 2. This has proven to be useful during an initialization of the tracker.

Figure 2. Reference face re-rendered in pose 60, 70 and 90 degrees.

3. Image processing

In the following subsections, the usage of image cues in estimating the head pose is presented. First we describe the use of color rectangle features. The integral images [15] allow the features to be extracted fast. The eigenspace-based pose estimation is evaluated next. To evaluate the robustness of such cues we perform a deterministic search for the best pose. We need to estimate the position $\{x, y, z\}$ and the Euler angles $\{\alpha, \beta, \gamma\}$ describing the head orientation in the coordinate system coupled with the camera.

3.1. Color rectangle features

The drawback of color-based tracking techniques is that are usually not very accurate. In this subsection we demonstrate that a rectangle pixel-sum can be used in model-based estimation of the head pose. Such a feature can provide sufficient discrimination capabilities in many tracking scenarios. In contrary, pixel-based features have often sharp minima in the observation model. The use of pixel-based texture models in head pose estimation is demonstrated in [13]. The work [8] uses relatively small number of the rectangle features to perform a tracking, since there was no efficient method to compute such features. An integral image-based tracker in which rectangle features are computed fast is presented in work [9].

The integral image at location x, y is defined as the sum of all pixels above and to the left of x, y. This can be expressed as follows:

$$ii(x, y) = \sum_{j=0}^{x} \sum_{k=0}^{y} I(j, k), \tag{1}$$

where $ii(x, y)$ is the value of the integral image at (x, y), and $I(j, k)$ represents the original image value. This formula can be rewritten into the following recurrent equations:

$$\begin{aligned} r(x, y) &= r(x, y-1) + I(x, y) \\ ii(x, y) &= ii(x-1, y) + r(x, y), \end{aligned} \tag{2}$$

where $r(x, y)$ is called the cumulative row sum, $r(x-1) = 0$, $ii(-1, y) = 0$, and $ii(x, -1) = 0$. On the basis of these recurrent equations the integral image can be computed in a single pass over the input image.

The pixel-sum in a rectangular image area is defined as follows:

$$p_s(x, y, w, h) = \sum_{j=x}^{x+w-1} \sum_{k=y}^{y+h-1} I(j, k), \tag{3}$$

where w and h are the width and the height of the rectangular area, and x, y determine the location of the upper right corner of the rectangle, see also Fig. 3. It can be demonstrated that this pixel-sum can be computed with only four lookups,

two subtractions and one addition, according to the following formula:

$$p_s = ii(x+w-1, y+h-1 + ii(x-1, y-1) - ii(x-1, y+h-1) - ii(x+w-1, y-1). \quad (4)$$

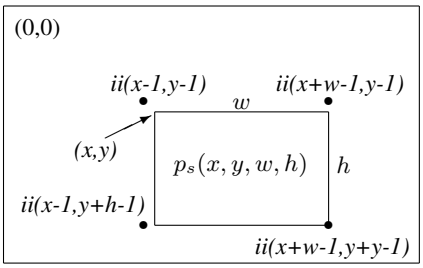

Figure 3. Calculation of the pixel-sum in a rectangle using the integral image.

To minimize the influence of different lighting conditions we normalized the input images to unit variance. Computing the standard deviation of pixel values in the rectangle requires only eight lookups and few operations. The standard deviation is given by: $\sigma = \sqrt{\mu^2 - \frac{1}{N}\sum x^2}$, where N is the number of pixels inside the rectangle, μ can be determined using the integral image, and $\sum x^2$ can be computed using a squared integral image.

The subject to be tracked is represented by mean colors of the pixels within rectangles R_i. The similarity between the reference face and the considered face candidate was measured using the normalized cross-correlation between the corresponding rectangle features. The normalized cross-correlation for color c is given by:

$$e^{(c)} = \frac{\sum_{\{x,y\}\in W}(I^{(c)}_{x+i,y+j} - \overline{I}^{(c)}_{i,j})(T^{(c)}_{x,y} - \overline{T}^{(c)})}{\sqrt{\sum_{\{x,y\}\in W}(I^{(c)}_{x+i,y+j} - \overline{I}^{(c)}_{i,j})^2 (T^{(c)}_{x,y} - \overline{T}^{(c)})^2}}, \quad (5)$$

where $I_{x,y}$ denotes the mean color pixel reflecting the value of the rectangle feature from the input image, $T_{x,y}$ denotes the mean color pixel representing the value of the rectangle feature from the reference image, and W is the number of non-background pixels in the reference image.

Assuming the independence of color pixel values, the maximum likelihood estimation is equivalent to determining the model pose that maximizes the error $\rho = e^{(r)}e^{(g)}e^{(b)}$, where r, g, b denote colors in the utilized color space. A considered pose is the estimated pose of the subject if after rendering to the frontal view best matches the reference face.

The search space $S = (x, y, z, \alpha, \beta, \gamma)$ is the set of all states within some range of estimated state in the last iteration. Assuming that the face in the frontal pose is symmetric, we generate on the basis of the rendered image two images. Taking into account the reflected symmetry, the first image is generated of the basis of the left side of the face, whereas the second one is created on the basis of the right side of the face. We then choose the image which better matches with the reference face. This manipulation was realized because of some undesirable effects accompanying the re-rendering. In particular, the invisible parts of the face, which after the transformation of the face to the frontal pose can not be re-rendered correctly, are detected easily.

Some images from the experiments are presented in Fig. 4. In this experiment a person moves and rotates his head in front of the wooden desks. The rectangle features were computed in interiors of the overlapping sub-images of size 5x5 pixels.

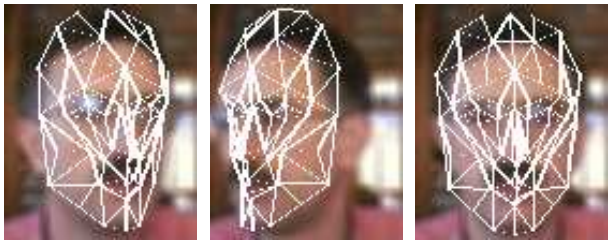

Figure 4. Estimation of the head pose using color rectangle features. Frames #6, #35, #50.

Figure 5. depicts the integral images which were obtained after re-rendering the face to frontal pose. These images have been extracted during a search for the best match in the frame #6, see also Fig. 4. The image (a) illustrates a rendered view of the face in case of over-rotating (about 10 deg), the image (b) demonstrates the rendered view in case of under-rotating (about -10 deg), whereas the next image demonstrates the pose of the face, which has been re-projected according to the estimated head pose. The last image in the sequence demonstrates the synthetic face that has been re-projected according to the estimated head pose in the frame #50. In this frame the face has been rotated about a small angle in order to transform it to the frontal pose and thus a relatively small degradation of the synthesized face can be observed.

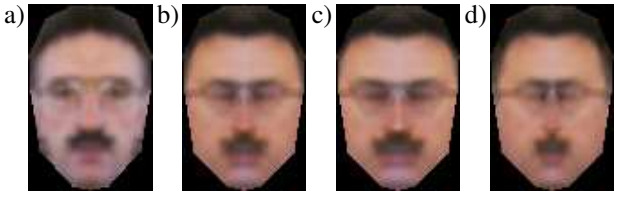

Figure 5. Re-rendering the face to the frontal pose in frames #6 (the first three images) and #50 (the last image). Over-rotated face (a), under-rotated face (b), re-projected face according to the estimated head pose (c), face rotated about a relatively small angle (d).

3.2. Model-based analysis via synthesis

Low-dimensional linear subspace models are often utilized to model the illumination bases [13] and for modeling the appearance [4]. In this subsection we demonstrate how linear subspaces can be used in the analysis through a model-based synthesis of images.

Low-dimensional linear subspace models utilize the principal components of face space reflecting the statistical properties of facial appearance. Our algorithm for estimating the head pose on the basis of linear subspace models operates on gray images. During initialization of the tracker we collect training faces $\mathbf{\Gamma_1}, \mathbf{\Gamma_2}, ..., \mathbf{\Gamma_M}$. The average face for this collection is computed as $\mathbf{\Psi} = \frac{1}{M}\sum_{m=1}^{M}\mathbf{\Gamma}_M$. Next, this image is subtracted from each training image and a new set of vectors $\mathbf{\Phi}_m = \mathbf{\Gamma}_m - \mathbf{\Psi}$ is created. Since the number of the face images is much smaller than the dimension of face space, there only exists $M-1$ nontrivial eigenvectors with the remaining ones associated with zero or negative eigenvalues. The matrix $\mathbf{B} = [\mathbf{\Phi}_1\mathbf{\Phi}_2, ..., \mathbf{\Phi}_M]$ is used to create the covariance matrix of $\mathbf{B}^T\mathbf{B}$. The eigenvectors $\mathbf{g}_m$ and eigenvalues λ_m of such a covariance matrix are determined finally. The eigenvalues of the covariance matrix indicate how much of the variance in the image is captured by the corresponding eigenvector. The higher the eigenvalue, the more characteristic features of a face does the corresponding eigenvector describe. The first $M^{'} = 6$ normalized eigenvectors, sorted by decreasing eigenvalue represent linear subspace which is further utilized in reconstruction of the face. The reconstruction is realized as follows:

$$\hat{\mathbf{\Gamma}} = \mathbf{\Psi} + \mathbf{G}\mathbf{G}^T(\mathbf{\Gamma} - \mathbf{\Psi}), \tag{6}$$

where the orthogonal columns of $\mathbf{G}$ are the eigenvectors.

The first six eigenfaces obtained in a typical initialization of the tracker are shown in Fig. 6. The eigenfaces depicted in the upper row were generated using a re-rendered face to frontal pose, whereas the bottom sequence of eigenfaces has been obtained with no re-rendered face. The second and fifth images in the upper row indicate some structures from the re-rendered image, for example see a vertical line at the nose which is present both in Fig. 5a and at the discussed images. The distorted bottom part of the face via the re-rendering can be observed in the depicted eigenfaces too. The eigenbasis was generated using mirrored face images.

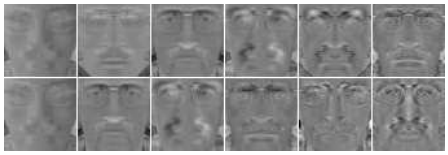

Figure 6. The first six eigenvectors. Eigenfaces generated with re-rendered face (top), using no re-rendered face (bottom).

In order to determine the pose the considered face candidate was re-rendered to the frontal pose and then reconstructed using the eigenfaces. The residual image was extracted using the current and the reconstructed image, see Fig. 7. To eliminate the influence of the background a suitable image mask was utilized during image comparison.

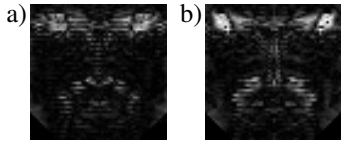

Figure 7. Residual images obtained in searching for the best pose in frame #6. Residual image for the re-projected face according to the estimated pose (a), residual image for re-projected face with over-rotation about 10 deg (b).

Figure 8. shows the images illustrating the estimated head poses, which were obtained using the discussed method. Comparing the depicted estimates of the pose with the estimates shown in Fig. 4. we can notice that a something larger error has been obtained for frame #6 and a something smaller error for the frame #35. Other experiments demonstrated that the utilized cues can be complementary to one another in various real situations.

One of the main disadvantages of the method based on eigenfaces is the necessity for the use of re-rendered images at the time of training. We observed that without such re-rendered images the method does not work. This makes the reuse of this method for different subject types something challenging. However, this subsection demonstrates that only one re-rendered image of a face profile can even be sufficient for model-based analysis through synthesis.

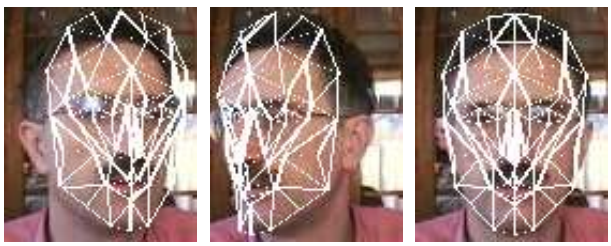

Figure 8. Estimation of the head pose using statistical facial texture model. Frames #6, #35, #50.

The estimated rotation angle of the head is shown in Fig. 9. Results of tracking using rectangle features and eigenfaces are plotted. The pre-recorded sequence was acquired at 10 fps and 320x240 resolution.

4. Particle filter

One of the goals of visual tracking is to estimate the unknown state from a set of noisy observations arriving in a

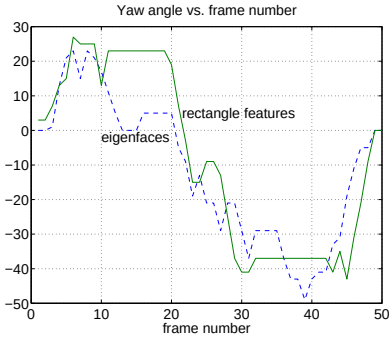

Figure 9. Yaw angle vs. frame number

sequential fashion. Recently, sequential Monte Carlo methods [5], also known as particle filters, have become increasingly popular stochastic approaches for approximating posterior distributions as they are neither limited to linear systems nor require Gaussian noise.

Two important components of each particle filter are motion model $p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ describing the state transition and observation model $p(\mathbf{z}_t \mid \mathbf{x}_t)$ describing the likelihood that a state $\mathbf{x}_t$ causes the observation $\mathbf{z}_t$. The particle filter approximates the posterior distribution $p(\mathbf{x}_t \mid \mathbf{z}_{1:t})$ by a set of weighted samples. Each sample represents the hypothetical state of the object. Starting with a weighted particle set $P = \left\{(\mathbf{x}_{t-1}^{(n)}, \pi_{t-1}^{(n)}) \mid n = 1...N\right\}$ approximately distributed according to $p(\mathbf{x}_{t-1} \mid \mathbf{z}_{1:t-1})$ the particle filter operates through predicting new particles from a proposal distribution. To give a new particle representation $P = \left\{(\mathbf{x}_t^{(n)}, \pi_t^{(n)}) \mid n = 1...N\right\}$ of the posterior density $p(\mathbf{x}_t \mid \mathbf{z}_{1:t})$ the weights of particles are set to $\pi_t^{(n)} \propto \pi_{t-1}^{(n)} p(\mathbf{z}_t \mid \mathbf{x}_t^{(n)}) p(\mathbf{x}_t^{(n)} \mid \mathbf{x}_{t-1}^{(n)}) / q(\mathbf{x}_t^{(n)} \mid \mathbf{x}_{t-1}^{(n)}, \mathbf{z}_t)$. When the proposal distribution from which particles are drawn is chosen as the distribution conditioning the state at the previous time step, the importance function reduces to $q(\mathbf{x}_t^{(n)} | \mathbf{x}_{t-1}^{(n)}, \mathbf{z}_t) = p(\mathbf{x}_t^{(n)} | \mathbf{x}_{t-1}^{(n)})$ and in consequence the weighting function takes the form $\pi_t^{(n)} \propto p(\mathbf{z}_t \mid \mathbf{x}_t^{(n)})$. This simplification leads to CONDENSATION [11], a variant of the particle filter, which is commonly applied in computer vision. From time to time the particles should be re-sampled according to their weights to avoid degeneracy [6].

One way to model the transition of the state is using a random walk which can be described by

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta, \tag{7}$$

where $\eta \sim N(0, \nu^2)$, and ν^2 is typically learned from training sequences. Such a choice was motivated by observation that the frame to frame pose difference in our previous test sequence was not to large, see also Fig. 9. The dimension of the state vector is 6.

In our approach the particle filter utilizes the observation models built on re-projection errors between the representation of synthesized face images in the frontal pose and the representation of reference faces. Generally, the particle filter works well when the conditional densities $p(\mathbf{z}_t \mid \mathbf{x}_t)$ are reasonably flat.

Using the normalized cross-correlation based error ρ we defined the observation model for the rectangle features as $p(\mathbf{z}^C \mid \mathbf{x}) = (\sqrt{2\pi}\sigma)^{-1} e^{-\frac{1-\rho}{2\sigma^2}}$. Using such weighting we favor the head pose candidates whose color appearances are similar to the appearances of the reference face. The second ingredient of the observation model reflecting the similarity of reconstructed facial textures was weighted in a similar manner $p(\mathbf{z}^T \mid \mathbf{x}) = (\sqrt{2\pi}\sigma)^{-1} e^{-\frac{1-\phi}{2\sigma^2}}$, where $1 - \phi$ denotes the normalized sum of residual pixels. Assuming that the observations are conditionally independent given the state we get the observation model $p(\mathbf{z}_t \mid \mathbf{x}_t) = p(\mathbf{z}_t^C \mid \mathbf{x}_t) p(\mathbf{z}_t^T \mid \mathbf{x}_t)$.

The reference image representing the tracked head has been accommodated over time. The actualization of the image has been realized according to the following equation $I_t = (1 - \gamma) I_{t-1} + \gamma R(\mathbf{x}_{t-1})$, where γ is accommodation rate, I_t denotes the reference image, $R(\mathbf{x}_{t-1})$ is the innovation consisting of the image re-rendered according to the estimated state. The accommodation rate depends on the head pose and it assumes larger values for frontal poses of the head. The normal of the considered triangular patch has been used to modify the accommodation rate of each pixel.

5. Experimental results

The initial poses of the head which are needed for the capture of texture maps are obtained using a graphical tool in which the mesh modeling the face could be translated along all six degrees of freedom. In addition some vertices of the mesh could be adjusted to obtain a person specific model of the head. During the initialization of the tracker the subject should be at rest and approximately facing the camera. Experiments has shown that the tracker is robust to small displacements and rotations (i.e. several pixels and degrees) of the head to be tracked. The initialization process generally takes less than one minute. A typical laptop computer equipped with 2.5 GHz Pentium IV is utilized to run the software prepared in C/C++ and operating at images of size 320x240. The system uses DirectX/OpenGL libraries and some procedures employ the computational resources available in the utilized graphic card. The testing sequences were taken using a Sony EVI D31 camera.

To demonstrate the abilities of our approach to head pose tracking we performed various experiments. Some images from testing sequence that are depicted in Fig. 10. show the ability of the tracker to estimate the head pose. All of these head motions appear to be tracked with an accepted accuracy. The width of the face in the images averages out 70 pixels.

The tracker runs with 400 particles at frame rates of 5-6 Hz. By dealing with multiple cues the presented approach can estimate the head pose reliably. We conducted experi-

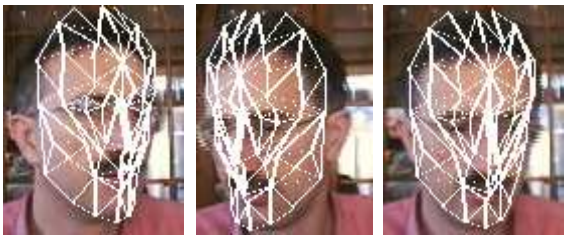

Figure 10. Estimation of the head pose using particle filter. Frames #19, #99, #115.

ments on the dataset [13] and compared the yaw and pitch estimated by our tracker to the ground truth. Figure 11. shows the comparison of the angles for one sequence from the mentioned dataset. We see that the tracker can estimate the pose quite accurately in most frames. The system is robust to occlusions of a small part of the face by eg. finger.

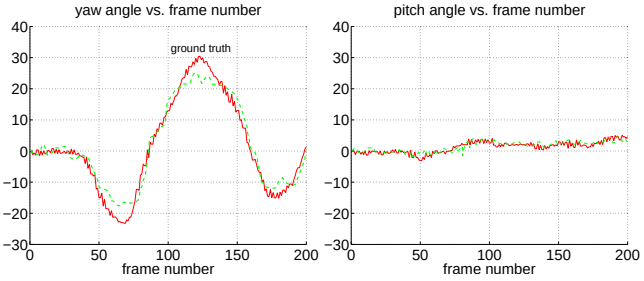

Figure 11. Head pose vs. frame number. The red/solid curve depicts the ground truth, the green/dashed depicts the estimates.

6. Conclusions

In this paper we presented a framework for estimating the head pose using a particle filter. We have shown that the observation model of the particle filter approximating the probability distributions in model-based analysis through synthesis can be constructed on the basis of rectangle image features. Experiments with real video sequences illustrate how such rectangle features can be used together with statistical facial texture models to improve the tracking capabilities of applications devoted to estimating the head pose. In deterministic search-based tracking we compared the usefulness of the mentioned above cues. Experiments showed that rectangular features are able to support the estimating of the head pose. These features make it general enough to be useful for many teleconference as well as human-machine applications. The performance of the particle filter-based head pose estimation has been demonstrated via experiments. Further work need to be performed to speed-up the algorithm through using the computational power offered by modern graphics hardware. Issues related to reduction the number of particles should also be taken into account.

7. Acknowledgement

This work has been supported by Polish Ministry of Education and Science under the grant 3T11C05730.

References

[1] J. Ahlberg and R. Forchheimer. Face tracking for model-based coding and face animation. *Int. J. on Imaging Systems and Technology*, 13(1):8–22, 2003.

[2] S. Basu, I. Essa, and A. Pentland. Motion regularization for model-based head tracking. In *Proc. of the Int. Conf. on Patt. Rec.*, pages 611–616, Vienna, Austria, 1996.

[3] V. Blanz and T. Vetter. Face recognition based on fitting a 3d morphable model. *IEEE Tr. on PAMI.*, 25(9):1–12, 2003.

[4] T. Darell, B. Moghaddam, and A. Pentland. Active face tracking and pose estimation in an interactive room. In *Int. Conf. on Comp. Vis. and Patt. Rec.*, pages 62–72, 1996.

[5] A. Doucet, N. Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[6] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

[7] G. Feng and P. Yuen. Recognition of head and shoulder face image using virtual frontal view image. *IEEE Tr. Syst. Man Cybern. and Syst. Humans*, 30(6):871–882, 2000.

[8] P. Fieguth and D. Terzopoulos. Color-based tracking of heads and other mobile objects at video frame rates. In *Proc. of the IEEE Int. Conf. on Comp. Vis. and Patt. Rec.*, pages 21–27, Hilton Head Island, 1997.

[9] B. Han, C. Yang, R. Duraiswami, and L. Davis. Bayesian filtering and integral image for visual tracking. In *6th Int. Workshop on Image Analysis for Multimedia Interactive Services*, Montreux, Switzerland, 2005.

[10] T. Horprasert. Computing 3d head orientation from a monocular image. In *Int. Conf. on Aut. Face and Gesture Rec.*, pages 27–32, 1996.

[11] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conf. on Computer Vision*, pages 343–356, Cambridge, 1996.

[12] N. Krahnstoever and R. Sharma. Appearance management and cue fusion for 3d model-based tracking. In *IEEE Int. Conf. on Comp. Vis. and Patt. Rec.*, pages 249–254, 2003.

[13] M. La Cascia, S. Sclaroff, and S. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models. *IEEE Tr. on PAMI*, 22(4):322–336, 2000.

[14] L. Vacchetti, V. Lepetit, and P. Fua. Stable real-time 3d tracking using online and offline information. *IEEE Tr. on Patt. Anal. and Machine Intell.*, 26(10):1385–1391, 2004.

[15] P. Viola and M. Jones. Robust real-time face detection. *Int. J. of Computer Vision*, 52(2):137–154, 2004.

[16] J. Xiao, S. Baker, I. Matthews, and T. Kanade. Real-time combined 2d+3d active appearance models. In *IEEE Int. Conf. on Comp. Vis. and Patt. Rec.*, pages 535–542, 2004.

[17] Z. Zivkovic and F. Heijden. A stabilized adaptive appearance changes model for 3d head tracking. In *ICCV Workshop on Rec., Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pages 175–182, Vancouver, Canada, 2001.