EXTENSION OF AAM WITH 3D SHAPE MODEL FOR FACIAL SHAPE TRACKING

Jaewon Sung, Daijin Kim

Pohang University of Science and Technology
Department of Computer Science and Engineering
San 31, Hyoja-Dong, Nam-Gu, Pohang, 790-784, Korea

ABSTRACT

This paper represents a 3D model-based object tracking algorithm to extract rigid and non-rigid motion of an object simultaneously . Previous AAM is extended by replacing 2D shape model with 3D shape model, and appropriate model fitting algorithm is derived. Proposed algorithm is applied to face tracking in video sequence to extract non-rigidly deforming facial shape.

1. INTRODUCTION

Face tracking is a fundamental technique that is required for face recognition, facial expression recognition and lip reading, model-based coding etc. in HCI (Human Computer Interface). Much research focuses on facial shape tracking from the viewpoint of facial expression recognition.

Xiao[1] modelled the face with cylindrical model. An iterative template matching algorithm is used to recover the 6 rigid motion parameters of the face from an image sequence. Li[2] gathered 3D shapes of a face, and learned shape space of face. The pose and shape parameters of a face are estimated using Kalman filter and genetic algorithm to obtain a frontal view image in successive image sequence. Salih[3] gathered 3D shapes of various facial expressions from a stereo image sequence to learn the shape space of facial expressions. The landmark points are tracked in a monocular view image sequence using optical flow algorithm constrained by learned shape space. This algorithm is robust to pose change, but this algorithm does not provide an initialization method. Essa[4] used a 3D shape model with muscles. In this model, the change of facial shape is driven by the activation of the attached muscles. The activation of each muscle is computed from optical flow information. This method does not consider the pose variation, and optical flow requires heavy computation. AAM (Active Appearance Model) [5] is widely used to track non-rigid objects including a face. In addition, an efficient model fitting algorithm is introduced by [6].

Thanks to .

Recently, AAM is extended to track 3D shape of non-rigid object. 2D shape model is constrained to follow the affine projection of 3D shape model [7]. The 3D shape model can be acquired from the tracking result of 2D AAM[8].

In this paper, we extended the previous AAM by replacing 2D shape model with 3D shape model. Therefore, the number of shape parameters is reduced to the dimension of 3D shape model.

The remainder of this paper is organized as follows. In section 2, the shape and appearance model for 3D extended AAM is described. In section 3, the fitting algorithm for the proposed model is derived. In section 4, the result of experiment is presented. In section 5, a conclusion and future works are drawn.

2. 3D EXTENDED AAM

2.1. Shape Model

The shape of a 3D object can be approximated by a 3D mesh with v vertices, which corresponds to the salient points on the surface of the object. The shape of an object is represented by a column vector with $3v \times 1$ elements as

$$\boldsymbol{s} = \left(s_x^1, s_y^1, s_z^1, \ldots, s_x^v, s_y^v, s_z^v\right)^T, \tag{1}$$

where s_x^j, s_y^j, s_z^j means the 3D coordinate of the j-th vertex in an object reference frame. If we apply PCA to a set of collected shapes of a non-rigid object, an arbitrary shape of the object can be synthesized by the linear combination of the shape parameters p_i multiplied to the basis vectors as

$$\boldsymbol{s} = \boldsymbol{s}_0 + \sum_{i=1}^{n} p_i \boldsymbol{s}_i, \tag{2}$$

where $\boldsymbol{s}_0$ means the mean shape.

We have to decide the location of origin and three directions for each axis to establish a 3D coordinate system for the object. We defined a coordinate system for faces as follows. Let a line spearing the center of the two eyes be l_e, and let a line which goes through the center of mouth and is perpendicular to the line l_e be l_m. Then the point of intersection is defined as the origin. We defined the line heading for left eye from the origin as x-axis and the line heading for the center of mouth from the origin as y-axis. The direction for z-axis can be determined from x and y axis.

2.2. Camera Model

We can observe only a 2D image that is the projection of the 3D nature of the object. To estimate the 3D shape from given image, we need to know the relation between the 3D nature in the world and its projection to the image plane. This relation can be regarded as the projection process of each vertex of the mesh.

The 3d coordinate of the j-th vertex in object reference frame $\boldsymbol{s}^j$ can be transformed to the coordinate in camera reference frame $\boldsymbol{X}^j$ using rotation matrix R and translation vector T as

$$\boldsymbol{X}^j = \left[x^j y^j z^j\right]^T = R\boldsymbol{s}^j + T. \tag{3}$$

The rigid motion (rotation R, translation T) can be represented by 6 parameters using twist representation[9],

$$\boldsymbol{q} = \left[w_x w_y w_z t_x t_y t_z\right]^T. \tag{4}$$

Three parameters w_x, w_y, w_z represents the amount of rotation along three axes, and t_x, t_y, t_z represents the location of the origin of object reference frame. These 6 parameters will be called as pose parameter. Rotation matrix R can be computed from Rodrigues formula.

The location of the j-th vertex $\boldsymbol{u}^j = [u^j \ v^j]^T$ in the image plane can be computed as

$$u^j(\boldsymbol{c}) = \frac{x^j}{z^j} f_\alpha + u_0 = \frac{\boldsymbol{r}_1^T \boldsymbol{s}^j + t_x}{\boldsymbol{r}_3^T \boldsymbol{s}^j + t_z} f_\alpha + u_0 \tag{5}$$

$$v^j(\boldsymbol{c}) = \frac{y^j}{z^j} f_\beta + v_0 = \frac{\boldsymbol{r}_2^T \boldsymbol{s}^j + t_y}{\boldsymbol{r}_3^T \boldsymbol{s}^j + t_z} f_\beta + v_0, \tag{6}$$

where $\boldsymbol{r}_i$ means the i-th row vector of the rotation matrix R and $\boldsymbol{c}$ is a concatenated vector of $\boldsymbol{p}$ and $\boldsymbol{q}$.

2.3. Appearance Model

To extract appearance data independent to the shape variation, we used Li's method to extract *shape-pose-free texture* [2]. The mean shape $\boldsymbol{s}_0$ at a fixed location T_0 with frontal view $R_0 = I_3$ is projected to image plane. Then the projected points on the image plane construct a 2D mesh $\boldsymbol{m}_0$. For an input image, we determine the location of predefined vertices on the input image, and these locations construct a 2D mesh $\boldsymbol{m}$ for this input image. Then the input image is warped to *shape-pose-free texture* template according to the correspondence between the two meshes $\boldsymbol{m}_0$and $\boldsymbol{m}$.

Again, applying PCA to the set of gathered *shape-pose-free textures*, we can build a linear model for appearance as

$$\boldsymbol{A} = \boldsymbol{A}_0 + \sum_{k=1}^{m} \alpha_k \boldsymbol{A}_k, \tag{7}$$

where $\boldsymbol{A}_0$ is mean appearance and α_k are appearance parameters.

3. FITTING ALGORITHM

AAM is a generative model that can synthesize the appearance of the object with the given parameters. To interpret an image with AAM, we must find the parameters that minimize the difference between synthesized image and an input image.

Matthews and Baker [6] showed that IC-LK algorithm[10] which is modified version of LK algorithm[11] to speed up the convergence process can be applied to the model fitting algorithm of AAM. To apply the IC-LK algorithm to 3D extended AAM, we have to define warping function, inverse of warping and composition of warping function. In this section, we briefly review IC-LK algorithm and modify this algorithm to adapt to 3D extended AAM.

3.1. IC-LK algorithm

In LK algorithm, the error is defined as the difference of intensity values between template image $\boldsymbol{A}_0$and input image as

$$E(\Delta \boldsymbol{c}) = \left[\boldsymbol{A}_0(\boldsymbol{x}) - I(W(\boldsymbol{x}; \boldsymbol{c_0} + \boldsymbol{\Delta c}))\right]^2, \tag{8}$$

where $\boldsymbol{c}$ is the parameter vector of the warping function. The LK algorithm iteratively finds the warping parameters that minimizes the error. The update rule for LK algorithm is

$$\boldsymbol{\Delta c} = H \sum_{\boldsymbol{x} \in \boldsymbol{A}_0} \left[\nabla I \frac{\partial W}{\partial \boldsymbol{c}}\right]^T \left[\boldsymbol{A}_0(\boldsymbol{x}) - I\big(W(\boldsymbol{x}; \boldsymbol{c_0})\big)\right] \tag{9}$$

$$H = \left\{ \sum_{\boldsymbol{x} \in \boldsymbol{A}_0} \left(\nabla I \frac{\partial W}{\partial \boldsymbol{c}}\right)^T \left(\nabla I \frac{\partial W}{\partial \boldsymbol{c}}\right) \right\}^{-1}, \tag{10}$$

where $\partial W / \partial \boldsymbol{c}$ is the Jacobian of warping function at a current estimate of the warping parameters $\boldsymbol{c}_0$, and ∇I is the gradient of input image at $W(\boldsymbol{x}; \boldsymbol{c})$. Jacobian $\partial W / \partial \boldsymbol{c}$ and pseudo Hessian H must be updated at every iteration.

In IC-LK algorithm, the error is defined as

$$E(\Delta \boldsymbol{c}) = \left[\boldsymbol{A}_0(W(\boldsymbol{x}; \boldsymbol{\Delta c})) - I(W(\boldsymbol{x}; \boldsymbol{c}))\right]^2, \tag{11}$$

and the warping function is updated as

$$W(\boldsymbol{x}; \boldsymbol{c}) = W(\boldsymbol{x}; \boldsymbol{c_0}) \circ W(\boldsymbol{x}; \boldsymbol{\Delta c})^{-1} \tag{12}$$

$$\Delta \boldsymbol{c} = H \sum_{\boldsymbol{x}\in \boldsymbol{A}_0} \left[\nabla \boldsymbol{A}_0 \frac{\partial W}{\partial \boldsymbol{c}}\right]^T \left[I\big(W(\boldsymbol{x};\boldsymbol{c_0})\big) - \boldsymbol{A}_0(\boldsymbol{x})\right]. \quad (13)$$

At this point, the Jacobian and the Hessian become constant because the Jacobian is always computed at $\boldsymbol{c} = 0$. The gradient of template is constant also. These changes reduce the computation in the iteration.

3.2. Image Warping

This section describes how to synthesize an image corresponding to given parameters with 3D extended AAM. First, the appearance is synthesized on a shape-pose-free texture template with appearance parameters. Next, equation (5) and (6) determine the position of each vertex on the image plane. These vertices define a 2D mesh $\boldsymbol{m}$. The appearance image synthesized on shape-pose-free texture template is piece-wise warped to the mesh $\boldsymbol{m}$ on the image plane. The warping function for a point $\boldsymbol{x}$ in a triangle $(\boldsymbol{x}_i^0, \boldsymbol{x}_j^0, \boldsymbol{x}_k^0)$ is determined by the destination triangle $(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k)$ as

$$W(\boldsymbol{x};\boldsymbol{c}) = \boldsymbol{x}_i + a(\boldsymbol{x}_j - \boldsymbol{x}_i) + b(\boldsymbol{x}_k - \boldsymbol{x}_i), \quad (14)$$

where a, b are coefficients determining the position $\boldsymbol{x}$ in a source triangle.

3.3. Warp Jacobian

The warping function in equation (5) and (6) is determined through the projected location of the vertices. The Jacobian is computed as

$$\frac{\partial W}{\partial \boldsymbol{c}} = \sum_{j=1}^{v} \left[\frac{\partial W}{\partial \boldsymbol{u}^j} \left(\frac{\partial \boldsymbol{u}^j}{\partial \boldsymbol{p}} \frac{\partial \boldsymbol{u}^j}{\partial \boldsymbol{q}}\right)\right]. \quad (15)$$

Jacobian $\frac{\partial W}{\partial \boldsymbol{u}^j}$ is computed from equation (14) and $\frac{\partial u^j}{\partial \boldsymbol{p}}, \frac{\partial v^j}{\partial \boldsymbol{p}}$ is computed as

$$\frac{\partial u^j}{\partial p_i} = \frac{(\boldsymbol{r}_1^T \boldsymbol{s}_i^j)(\boldsymbol{r}_3^T \boldsymbol{s}^j + t_z) - (\boldsymbol{r}_1^T \boldsymbol{s}^j + t_x)(\boldsymbol{r}_3^T \boldsymbol{s}_i^j)}{(\boldsymbol{r}_3^T \boldsymbol{s}^j + t_z)^2} \cdot f_\alpha \quad (16)$$

$$\frac{\partial v^j}{\partial p_i} = \frac{(\boldsymbol{r}_2^T \boldsymbol{s}_i^j)(\boldsymbol{r}_3^T \boldsymbol{s}^j + t_z) - (\boldsymbol{r}_1^T \boldsymbol{s}^j + t_y)(\boldsymbol{r}_3^T \boldsymbol{s}_i^j)}{(\boldsymbol{r}_3^T \boldsymbol{s}^j + t_z)^2} \cdot f_\beta \quad (17)$$

$$\frac{\partial \boldsymbol{u}^j}{\partial \boldsymbol{q}} = \begin{bmatrix} -xy & x^2+z^2 & -yz & z & 0 & -x \\ -(y^2+z^2) & xy & xz & 0 & z & -y \end{bmatrix} \begin{bmatrix} f_\alpha & 0 \\ 0 & f_\beta \end{bmatrix} \frac{1}{z^2}. \quad (18)$$

The values x, y, z are the coordinates of each vertex of the mean shape placed at R_0, T_0.

3.4. Composition of warping

The input image is backward warped to be compared with a template image in every iteration. $\Delta \boldsymbol{c}$ at equation (12) represents the amount that the vertices in the template image should move when compared to the backward warped input image. To update the warping function, we must compose current warping function with an inverted warping function corresponding to $\Delta \boldsymbol{c}$. The inverse of the warping function is simply computed by inverting the sign of warping parameters $W(\boldsymbol{x};\boldsymbol{\Delta c}^{-1}) = W(\boldsymbol{x};-\boldsymbol{\Delta c})$. New position of each vertex $\hat{\boldsymbol{u}}^j$ on image plane is computed by composing the current warping function and the inverted warping function. The remaining problem is to compute the new parameter $\boldsymbol{c}$ of composed warping function. If we know the current values of parameters $\boldsymbol{p_0}, \boldsymbol{q_0}$, the new position of j-th vertex can be approximated as

$$\hat{\boldsymbol{u}}^j = \boldsymbol{u}^j(\boldsymbol{c_0} + \Delta\hat{\boldsymbol{c}}). \quad (19)$$

This equation can be rewritten as linear equation system using first order Taylor expansion as

$$\hat{\boldsymbol{u}}^j - \boldsymbol{u}^j(\boldsymbol{c_0}) \approx \Delta\hat{\boldsymbol{u}}^j = \frac{\partial \boldsymbol{u}^j}{\partial \boldsymbol{c}} \Delta\hat{\boldsymbol{c}}. \quad (20)$$

In this equation, the movement of each vertex $\Delta\hat{\boldsymbol{u}}^j$ is computed by substracting current position from estimated new position and the concatenation of two Jacobian $\left[\frac{\partial \boldsymbol{u}^j}{\partial \boldsymbol{p}} \; \frac{\partial \boldsymbol{u}^j}{\partial \boldsymbol{q}}\right]$ is computed from the equations (16), (17) and (18). At this time, these equations must be computed at the current pose parameter.

4. EXPERIMENTAL RESULTS

We recorded a test movie with a stereo vision camera. This movie contains 103 frames, and each frame consists of gray level stereo images with 640x480 resolution.

4.1. Data collection

We defined 41 landmarks on a face and extracted 3D shape data from selected frames. 7 frames (1, 15, 30, 45, 60, 75, 90'th frames) are selected in the sequence. The landmarks are manually located on the right image, and the corresponding points are determined by finding the best matching points on the left image. Then, 3D structure is reconstructed using stereo vision technique. 7 shape data and 14 appearance data are collected to learn 3D extended AAM. We used 5 shape parameters, 7 appearance parameters, and 6 pose parameters.

4.2. Tracking result

Tracking begins with rough initialization. 3D locations of two eyes, and center of the mouth are extracted from first stereo frame, and the pose parameters of mean shape $\boldsymbol{s}_0$ are estimated from the three points. Shape and appearance parameters are set to zero. We used right image sequence for tracking experiments.

Because we do not have the ground truth data (the true values of shape, appearance, and pose parameters) for this experiment at present, we checked the tracked result whether the predefined vertices are coherently tracking the point in the image sequence. The test movie contains 4 facial expressions in sequential order: normal, laughing, frowning, and surprised. Figure 1 shows some examples of tracked image sequence. Bar graphs on the left part of the image visualize the current values of 5 shape parameters. Figure 2 shows the change of 5 shape parameters throughout the entire sequence.

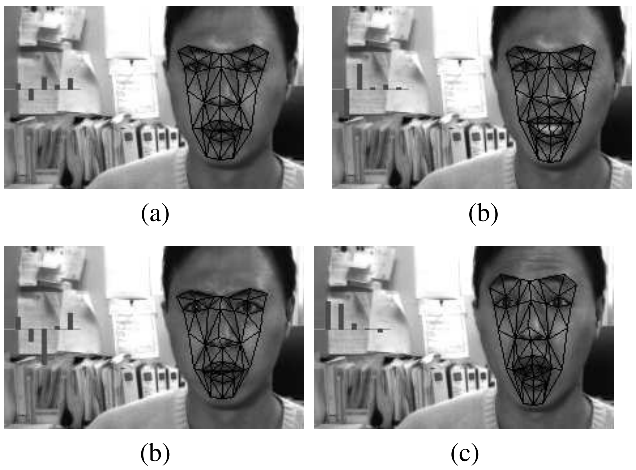

Fig. 1. Examples of tracked image sequence.

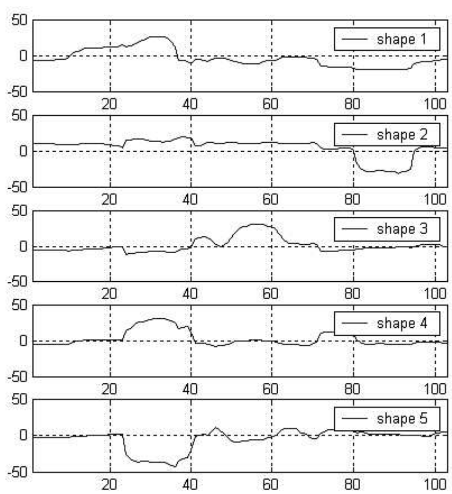

Fig. 2. Tracking result of 5 shape parameters

5. CONCLUSION

In this paper, we developed a facial shape tracking system using 3D extended AAM. 3D shape model is used instead of a prior 2D shape model. and perspective camera model is combined to the warping function. Model fitting algorithm is derived by modifying the IC-LK algorithm. Non-rigid deformation of facial shape is extracted using developed face tracking system. We will adopt more experiments to test the performance of this system.

6. REFERENCES

[1] Jing Xiao Takeo Kanade, "Robust full-motion recovery of head by dynamic templates and registration techniques," in *AFGR*. IEEE, May 2002.

[2] Yongmin Li Shaogang Gong, Heather Liddell, "Modelling faces dynamically across views and over time," in *ICCV*, 2001.

[3] Gokturk Jean-Yves Bouguest Carlo Tomasi, Bernd Girod, "Model-based face tracking for view-independent facial expression recognition," in *AFGR*. IEEE, 2002.

[4] I. Essa A. Pentland, "Coding, analysis, interpretation and recognition of facial expressions," *PAMI*, pp. 757–763, 1997.

[5] T.F. Cootes G.J. Edwards, C.J. Taylor, "Active appearance models," *PAMI*, vol. 23, no. 6, pp. 681–685, 2001.

[6] I. Matthews S. Baker, "Active appearance models revisited," Cmu-ri-tr-03-01, CMU, Apr 2003.

[7] I. Matthews S. Baker and J. Xiao T. Kanade, "Real-time combined 2d+3d active appearance models," in *CVPR*, Jan. 2004.

[8] J. Chai and J. Xiao T. Kanade, "A closed-form solution to non-rigid shape and motion recovery," in *ECCV*, 2004.

[9] C. Bregler J. Malik, "Tracking people with twists and exponential maps," in *CVPR*, 1998, pp. 8–15.

[10] Simon Baker Iain Matthews, "Lucas-kanade 20 years on : A unifying framework," *IJCV*, 2004.

[11] B.D. Lucas T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int. Joint Conf. Artificial Intelligence*, 1981, pp. 674–679.