3D Active Appearance Model for Aligning Faces in 2D Images

Chun-Wei Chen and Chieh-Chih Wang

***Abstract*— Perceiving human faces is one of the most important functions for human robot interaction. The active appearance model (AAM) is a statistical approach that models the shape and texture of a target object. According to a number of the existing works, AAM has a great success in modeling human faces. Unfortunately, the traditional AAM framework could fail when the face pose changes as only 2D information is used to model a 3D object. To overcome this limitation, we propose a 3D AAM framework in which a 3D shape model and an appearance model are used to model human faces. Instead of choosing a proper weighting constant to balance the contributions from appearance similarity and the constraint on consistent 2D shape with 3D shape in the existing work, our approach directly matches 2D visual faces with the 3D shape model. No balancing weighting between 2D shape and 3D shape is needed. In addition, only frontal faces are needed for training and non-frontal faces can be aligned successfully. The experimental results with 20 subjects demonstrate the effectiveness of the proposed approach.**

I. Introduction

As robots should work closely to human beings in the near future, human robot interaction (HRI) is critical and has attracted a great deal of attention. A robot should have enough understanding of the users in order to well perform HRI. As faces are one of the most expressive parts of human beings, face detection and recognition are very important topics for many applications. The existing face recognition algorithms and systems are summarized and evaluated in [1]. However, only accomplishing face detection and recognition may not be enough for some tasks such as facial expression analysis, gaze tracking and lip reading. To more delicately analyze human faces, detection and localization of human facial features such as eyes, nose, mouth, eyebrows, eyeballs and nostrils are critical.

The active appearance model (AAM) framework [2] provides an effective approach to localize human facial features on 2D visual images. AAM is a statistical model which is divided into two parts, a shape model [3] and an appearance model [4]. Matthews *et al.* [5] combined the AAM fitting algorithm with four different update rules. The one using the inverse compositional algorithm [6] can be run in real-time. Gross *et al.* [7] proposed a method to deal with occlusion. In [8], an application of generating realistic lip movement from speech using AAM was presented. Though human faces may have non-rigid deformations, AAM can model human faces properly as human faces have a consistent structure.

C.-W. Chen is with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan jwchen@robotics.csie.ntu.edu.tw

C.-C. Wang is with the Department of Computer Science and Information Engineering and the Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei, Taiwan bobwang@ntu.edu.tw

(a) Face alignment using 2D AAM

(b) Face alignment using 3D AAM

Fig. 1. A non-frontal face is aligned successfully with 3D AAM while 2D AAM failed.

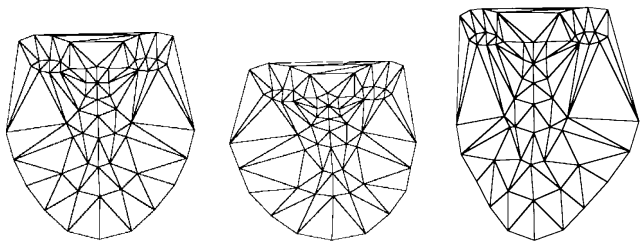
The traditional 2D AAM has a great success on human face alignment as long as only frontal images are matched. Due to head pose variations in the 3D world, 2D AAM may fail to converge as shown in Figure 1. To deal with this issue with the 2D AAM framework, one way to collect data in every possible pose which is infeasible in practice. Xiao *et al.* [9] proposed a 2D+3D AAM which exploits the 2D shape and 3D shape models simultaneously. The shape instance generated by 2D AAM is varied to be consistent with a possible 3D shape. This constraint is formulated as a part of the cost function. To combine this constraint into the original cost function, a balancing weighting is added. The value of this weighting constant is determined manually.

Based on the existing 2D AAM framework [5], we propose to directly use a 3D shape model with the appearance model. In our 3D AAM framework, only frontal face images are needed for learning. While the aligning is carried out, the 3D shape model generates faces in different poses by varying rotation and translation of a frontal face instance. Our main contribution is to extend the search space of 2D AAM to the 3D world and no balancing weighting between the 2D shape and 3D shape models is needed. The data collected from 20 subjects are used to verify the proposed approach. The ample experimental results show that our approach is effective to deal with the face pose variation issue.

The remainder of the paper is organized as follows. Section II reviews the traditional 2D AAM. The proposed 3D AAM framework and algorithm are described in Section III. The experimental results in Section IV demonstrate the proposed approach works under different conditions. Finally, conclusion and future work are in Section V.

II. 2D Active Appearance Model

In this section, the shape model and the appearance model of the traditional 2D AAM framework are briefly described. See [2] for more details.



(a) (b) (c)

Fig. 2. An example of the shape model. The shape can vary with the shape parameter. The left-most figure is the mean shape. The others are instances of shape with different shape parameters.

(a) (b) (c)

Fig. 3. Examples of varying appearance parameter λ and shape parameter $\mathbf{p}$ to represent different kinds of human looking. The left-most figure is mean appearance. The center and right-most figures are examples of varying appearance parameters.

A. The 2D Shape Model

A shape consists of the fixed number of 2D points that describe the shape of the target object. Different shapes of the objects of the same category are labeled from training data. Rotation, translation and scale variations of these shapes are removed before learning. The Procrustes analysis [10] is then performed to align these shapes. Principal component analysis (PCA) is applied with the aligned shapes to build the shape model. After training, the mean shape $\mathbf{s}_0$ and the shape variation basis $\{\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_m\}$ can be used to reconstruct any shape $\mathbf{s}$ as the following form:

$$\mathbf{s} = \mathbf{s}_0 + \sum_{i=1}^{m} p_i \mathbf{s}_i, \tag{1}$$

where $\mathbf{p} = \{p_i\}_{i=1}^{m}$ is a real number set that called shape parameters. Figure 2 shows the shape instances resulted from different shape parameters.

B. The Appearance Model

Appearance is texture or intensity of the target object. Given the shape model, all the training images are transformed into their shape-free images. The mean shape serves as the comparison benchmark. Therefore, the mean shape is also called reference shape. All the texture on the images are mapped onto the mean shape. For each image, its appearance is represented as a vector in the same order and in the same dimension. This transformation is to make sure that all training and testing images are of the same dimension. In addition, the texture may change due to the lighting or camera settings. Therefore, normalization of the texture is necessary. After these procedures, PCA is performed on the training data to compute the appearance model.

The appearance model consists of the mean appearance, A_0, and the appearance variation basis, $\{A_i\}_{i=1}^{n}$. For each appearance parameter set $\lambda = \{\lambda_i\}_{i=1}^{n}$, a corresponding appearance is defined as

$$A = A_0 + \sum_{i=1}^{n} \lambda_i A_i. \tag{2}$$

Figure 3 shows the different appearance instances.

With the 2D shape and appearance model, the alignment process is to find the parameters of these two models for minimizing the difference between the test image and the face model.

III. 3D Active Appearance Model

In this section, the proposed 3D AAM framework is described in detail. The differences between 2D AAM and the proposed 3D AAM are addressed.

A. The 3D Shape Model

Instead of using the 2D shape model, the 3D shape model is applied to describe the geometry of human faces. A 3D morphable model (3DMM) [11] is a technique in the computer graphics literature to synthesize different looking of faces. 3DMM can generate different shapes and appearances of a human face and preserve the naturalness of the generated faces. 3DMM consists of the 3D shape and texture models. The 3D shape model in 3DMM is of dense landmarks in order to generate realistic faces. The 3D shape model in our framework follows the same principle but with sparser landmarks. Each landmark of the shape is extended from two dimensions to three dimensions. The mean shape and shape variation basis are modified as follows:

$$\mathbf{s}_i = \begin{pmatrix} \mathbf{s}_i^1 & \mathbf{s}_i^2 & \cdots & \mathbf{s}_i^N \end{pmatrix} = \begin{pmatrix} x_i^1 & x_i^2 & \cdots & x_i^N \\ y_i^1 & y_i^2 & \cdots & y_i^N \\ z_i^1 & z_i^2 & \cdots & z_i^N \end{pmatrix}. \tag{3}$$

With the 3D shape model, the pose of the target object is needed to determine the 2D shape shown in an image. Let the translation of the target object in the 3D world be $\mathbf{t} = (t_x, t_y, t_z)$ and the orientation of the target object be $\theta = (\alpha, \beta, \gamma)$ with respect to the camera coordinate system. Given a pose vector $\mathbf{q} = (\mathbf{t}, \theta)$, the 3D shape $\mathbf{s}$ can be transformed to a new location.

$$\mathbf{s}' = R(\theta)\,\mathbf{s} + \underbrace{\begin{pmatrix} t_x & t_x & \cdots & t_x \\ t_y & t_y & \cdots & t_y \\ t_z & t_z & \cdots & t_z \end{pmatrix}}_{N \text{ columns}}, \tag{4}$$

where $R(\theta)$ stands for the rotation matrix.

B. 3D Shape on 2D Image

Given the 3D location of a shape with respect to the camera coordinate system, the shape of the object can be easily projected onto the 2D image plane. The perspective camera model is applied in this work. A point $(X, Y, Z)^T$ in the 3D world is projected by function $\mathbf{P}$ according to the camera intrinsic parameters as follows:

$$\mathbf{P}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \frac{f}{Z}\begin{pmatrix} X \\ Y \end{pmatrix} + \begin{pmatrix} o_x \\ o_y \end{pmatrix}, \tag{5}$$

where f is the focal length and (o_x, o_y) is the principal point location on the image. The corresponding 2D shape consists of the landmarks projected by the function $\mathbf{P}$.

C. Warp Function

After establishing mapping from 3D shape to 2D shape, the next problem is to map the texture of the 2D shape on the image into shape-free image. As the 2D shape consists of limited number of landmarks, the texture mapping is defined piecewisely through the mesh formed by the landmarks of the shape. The mapping between two corresponding meshes is addressed first. Assume that a point $\mathbf{x} = (x, y)^T$ is inside the mesh formed by the landmarks $\mathbf{x}_i = (x_i, y_i)^T$, $\mathbf{x}_j = (x_j, y_j)^T$, and $\mathbf{x}_k = (x_k, y_k)^T$ in the reference shape. It is feasible to find unique $\phi_{\mathbf{x}}$ and $\omega_{\mathbf{x}}$ such that

$$\mathbf{x} = \mathbf{x}_i + \phi_{\mathbf{x}}\left(\mathbf{x}_j - \mathbf{x}_i\right) + \omega_{\mathbf{x}}\left(\mathbf{x}_k - \mathbf{x}_i\right). \tag{6}$$

The mapping to shape-free image then can be written as

$$\mathbf{W}\left(\mathbf{x}; \mathbf{p}, \mathbf{q}\right) = \mathbf{x}'_i + \phi_{\mathbf{x}}\left(\mathbf{x}'_j - \mathbf{x}'_i\right) + \omega_{\mathbf{x}}\left(\mathbf{x}'_k - \mathbf{x}'_i\right), \tag{7}$$

where $\mathbf{x}'_i$, $\mathbf{x}'_j$, and $\mathbf{x}'_k$ are the corresponding landmarks, on the 2D shape, of i-th, j-th, and k-th landmarks from a 3D shape. Let $\mathbf{I}$ denote an image and $\mathbf{I}(\mathbf{x})$ be the pixel value at point $\mathbf{x}$. Then the texture on the current shape can be mapped onto the reference shape using function composition, i.e. $\mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p}, \mathbf{q}\right)\right)$. Accordingly, the appearance model is constructed using the approach addressed in Section II-B.

D. Fitting a 3D AAM to an Image

Thus far, we have constructed the 3D shape model and the appearance model. The next step is to align a 3D AAM to a test image. The fitting process can be treated as a problem of minimizing the norm of the difference vector between the model appearance and testing appearance, i.e.

$$\min_{\lambda, \mathbf{p}, \mathbf{q}} \sum_{\mathbf{x} \in \mathbf{s}_0} \left(\mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p}, \mathbf{q}\right)\right) - \mathbf{A}_0\left(\mathbf{x}\right) - \sum_{i=1}^{n} \lambda_i \mathbf{A}_i\left(\mathbf{x}\right) \right)^2. \tag{8}$$

We now describe the stages to linearize this nonlinear cost function and the least-square solution.

1) Linearization of the Cost Function: To linearize the cost function, we assume that λ, $\mathbf{p}$, and $\mathbf{q}$ are initial guesses, and solve for the update amount $\Delta\lambda$, $\Delta\mathbf{p}$, and $\Delta\mathbf{q}$ iteratively, i.e. minimize

$$\sum_{\mathbf{x} \in \mathbf{s}_0} \left(\mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p} + \Delta\mathbf{p}, \mathbf{q} + \Delta\mathbf{q}\right)\right) - \mathbf{A}_0\left(\mathbf{x}\right) - \sum_{i=1}^{n} \left(\lambda_i + \Delta\lambda_i\right) \mathbf{A}_i\left(\mathbf{x}\right) \right)^2 \tag{9}$$

with respect to $\Delta\lambda$, $\Delta\mathbf{p}$, and $\Delta\mathbf{q}$ and update the parameters through adding directly. The cost function in Equation 9 can be linearized with respect to $\mathbf{p}$ and $\mathbf{q}$ using the first order approximation as shown below:

$$\sum_{\mathbf{x} \in \mathbf{s}_0} \left(\nabla\mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p}, \mathbf{q}\right)\right) \frac{\partial \mathbf{W}}{\partial\left(\mathbf{p}, \mathbf{q}\right)}\left(\mathbf{x}; \mathbf{p}, \mathbf{q}\right) \begin{pmatrix} \Delta\mathbf{p} \\ \Delta\mathbf{q} \end{pmatrix} + \mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{p}, \mathbf{q}\right)\right) - \mathbf{A}_0\left(\mathbf{x}\right) - \sum_{i=1}^{n} \left(\lambda_i + \Delta\lambda_i\right) \mathbf{A}_i\left(\mathbf{x}\right) \right)^2. \tag{10}$$

2) Minimization of the Cost Function: The problem can be solved using the least-square approach after the orders of some terms are rearranged. The term $\sum_{i=1}^{n} \left(\lambda_i + \Delta\lambda_i\right) \mathbf{A}_i\left(\mathbf{x}\right)$ is split into two terms, $\sum_{i=1}^{n} \lambda_i \mathbf{A}_i\left(\mathbf{x}\right)$ and $\sum_{i=1}^{n} \Delta\lambda_i \mathbf{A}_i\left(\mathbf{x}\right)$, and the latter term is then coupled with the vector $(\mathbf{p}, \mathbf{q})^T$. Equation 10 now becomes

$$\sum_{\mathbf{x} \in \mathbf{s}_0} \left(\begin{pmatrix} \nabla\mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{r}\right)\right) \frac{\partial \mathbf{W}}{\partial \mathbf{r}}\left(\mathbf{x}; \mathbf{r}\right) & -\mathbf{A}_1\left(\mathbf{x}\right) & \cdots & -\mathbf{A}_n\left(\mathbf{x}\right) \end{pmatrix} \begin{pmatrix} \Delta\mathbf{r} \\ \Delta\lambda \end{pmatrix} + \mathbf{I}\left(\mathbf{W}\left(\mathbf{x}; \mathbf{r}\right)\right) - \mathbf{A}_0\left(\mathbf{x}\right) - \sum_{i=1}^{n} \lambda_i \mathbf{A}_i\left(\mathbf{x}\right) \right)^2,$$

where $\mathbf{r} = (\mathbf{p}, \mathbf{q})^T$ and $\Delta\mathbf{r} = (\Delta\mathbf{p}, \Delta\mathbf{q})^T$. After some substitutions, the function above can be minimized using the least-square solution.

3) Warp Jacobian: In the previous derivation, the Jacobian of the warp function $\mathbf{W}$ is needed.For each affine transformation, it depends only on the following points.

1) The landmarks, $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{x}_k$, of the mesh in the reference shape that the input $\mathbf{x}$ lies in.
2) The corresponding landmarks, $\mathbf{x}'_i$, $\mathbf{x}'_j$, and $\mathbf{x}'_k$ in the current shape which is determined by parameters $\mathbf{p}$ and $\mathbf{q}$, of $\mathbf{x}_i$, $\mathbf{x}_j$, and $\mathbf{x}_k$.

Let's write the Jacobian of the warp function more explicitly.

$$\frac{\partial \mathbf{W}}{\partial \mathbf{r}}\left(\mathbf{x}; \mathbf{r}\right) = \frac{\partial\left(\mathbf{x}'_i + \phi_{\mathbf{x}}\left(\mathbf{x}'_j - \mathbf{x}'_i\right) + \omega_{\mathbf{x}}\left(\mathbf{x}'_k - \mathbf{x}'_i\right)\right)}{\partial \mathbf{r}} \tag{11}$$

Applying the chain rule to the term above, it can be rewritten as

$$\frac{\partial \mathbf{W}}{\partial \mathbf{r}} = \frac{\partial \mathbf{W}}{\partial \mathbf{x}'_i}\frac{\partial \mathbf{x}'_i}{\partial \mathbf{r}} + \frac{\partial \mathbf{W}}{\partial \mathbf{x}'_j}\frac{\partial \mathbf{x}'_j}{\partial \mathbf{r}} + \frac{\partial \mathbf{W}}{\partial \mathbf{x}'_k}\frac{\partial \mathbf{x}'_k}{\partial \mathbf{r}} \tag{12}$$

with

$$\frac{\partial \mathbf{W}}{\partial \mathbf{x}'_i} = 1 - \phi_{\mathbf{x}} - \omega_{\mathbf{x}} \tag{13a}$$

$$\frac{\partial \mathbf{W}}{\partial \mathbf{x}'_j} = \phi_{\mathbf{x}} \tag{13b}$$

$$\frac{\partial \mathbf{W}}{\partial \mathbf{x}'_k} = \omega_{\mathbf{x}}. \tag{13c}$$

It shows that the value of the Jacobian of the warp function at point $\mathbf{x}$ is the linear combination of the Jacobians of the landmarks, i.e. $\frac{\partial \mathbf{x}'_i}{\partial \mathbf{r}}$, $\frac{\partial \mathbf{x}'_j}{\partial \mathbf{r}}$ and $\frac{\partial \mathbf{x}'_k}{\partial \mathbf{r}}$. The point $\mathbf{x}'_i$ stands for the position of the i-th landmark in the current 2D shape. Recall that the current 3D shape can be retrieved from the current model parameter $\mathbf{p}$ and current pose parameter $\mathbf{q}$ as shown in Equations 1 and 4. In addition, the position of the i-th landmark in the current 2D shape, $\mathbf{x}'_i$, can be computed through applying the projection $\mathbf{P}$ to the i-th column of $\mathbf{s}'$. Thus, the 2D landmark $\mathbf{x}'_i$ can be written in the following form

$$\mathbf{x}'_i = \mathbf{P}\left(\mathbf{X}'_i\right), \tag{14}$$

where $\mathbf{X}'_i$ is the i-th column of $\mathbf{s}'$ (or the position of the i-th landmark in the current 3D shape $\mathbf{s}'$).

To evaluate the Jacobian of $\mathbf{x}'_i$ in Equation 12, the chain rule is applied. The Jacobian is

$$\frac{\partial \mathbf{x}'_i}{\partial \mathbf{r}} = \frac{\partial \mathbf{P}\left(\mathbf{X}'_i\right)}{\partial \mathbf{r}} \tag{15a}$$

$$= \nabla \mathbf{P}\left(\mathbf{x}\right)|_{\mathbf{x}=\mathbf{X}'_i} \frac{\partial \mathbf{X}'_i}{\partial \mathbf{r}}. \tag{15b}$$

The first part is straightforward.

$$\nabla \mathbf{P}\left(X, Y, Z\right) = \begin{pmatrix} \frac{f}{Z} & 0 & -\frac{f}{Z^2}X \\ 0 & \frac{f}{Z} & -\frac{f}{Z^2}Y \end{pmatrix}. \tag{16}$$

The second part is divided into three parts which are $\frac{\partial \mathbf{X}'_i}{\partial \mathbf{p}}$, $\frac{\partial \mathbf{X}'_i}{\partial \theta}$ and $\frac{\partial \mathbf{X}'_i}{\partial \mathbf{t}}$.

The cost function is then minimized by computing and updating model parameters and pose parameters iteratively. The scheme of optimization is illustrated by Algorithm 1.

Note that the proposed method can generate 3D shape instances in the 3D world. The method in [9] is based on 2D AAM and couples the constraint that preserves the consistency between 3D shapes and 2D shapes with the cost function. To preserve this consistency, a constant K should be chosen to balance the weighting between the original cost function and the consistency. In our work, no constant is needed since we use the 3D shape model only rather than use the 2D and 3D shape model together.

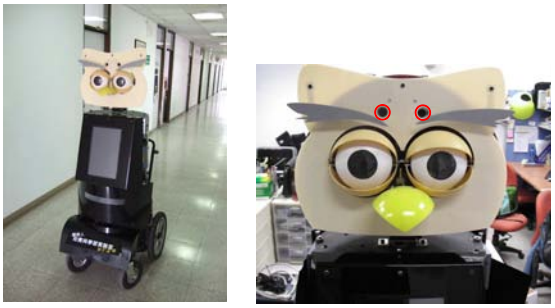
IV. Experimental Results

In this section, the experiment details and results are described.

Algorithm 1 Fitting a 3D AAM to an image

Require: Initial estimates of model parameters,$\mathbf{p}_0$ and λ_0, and pose parameter ,$\mathbf{q}_0$.

1: Set parameters $\mathbf{p} \leftarrow \mathbf{p}_0$, $\lambda \leftarrow \lambda_0$, and $\mathbf{q} \leftarrow \mathbf{q}_0$
2: Calculate gradient image $\nabla\mathbf{I}$ of input image $\mathbf{I}$.
3: **while** not converge yet **do**
4: Warp $\mathbf{I}$ and $\nabla\mathbf{I}$ through $\mathbf{W}\left(\mathbf{x};\mathbf{p},\mathbf{q}\right)$ to compute $\mathbf{I}\left(\mathbf{W}\left(\mathbf{x};\mathbf{p},\mathbf{q}\right)\right)$ and $\nabla\mathbf{I}\left(\mathbf{W}\left(\mathbf{x};\mathbf{p},\mathbf{q}\right)\right)$
5: Calculate the error vector $\mathbf{b} \leftarrow \mathbf{I}\left(\mathbf{W}\left(\cdot;\mathbf{r}\right)\right) - \mathbf{A}_0 - \sum_{i=1}^{n} \lambda_i \mathbf{A}_i$
6: Calculate the Jacobian $\frac{\partial \mathbf{W}}{\partial(\mathbf{p},\mathbf{q})}\left(\mathbf{x};\mathbf{p},\mathbf{q}\right)$of the warp function $\mathbf{W}$
7: Use Least-square approach to find $\Delta\mathbf{p}$, $\Delta\mathbf{q}$, and $\Delta\lambda$
8: Update parameter: $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$
9: Update parameter: $\mathbf{q} \leftarrow \mathbf{q} + \Delta\mathbf{q}$
10: Update parameter: $\lambda \leftarrow \lambda + \Delta\lambda$
11: **end while**



(a) (b)

Fig. 4. The NTU-PAL2 robot. The cameras are mounted at the locations depicted by the red circles in the right figure.

A. Data Collection

Our data are collected using the cameras mounted on the NTU-PAL2 robot as shown in Figure 4. 62 landmarks are annotated in every training image. A template 3D face shape $\mathbf{S}_T$ is applied to transform 2D data to 3D data.

$$\mathbf{S}_T = \begin{pmatrix} x_1 & x_2 & \cdots & x_N \\ y_1 & y_2 & \cdots & y_N \\ z_1 & z_2 & \cdots & z_N \end{pmatrix} \tag{17}$$

Let S be the annotated 2D shape with

$$\mathsf{S} = \begin{pmatrix} \hat{x}_1 & \hat{x}_2 & \cdots & \hat{x}_N \\ \hat{y}_1 & \hat{y}_2 & \cdots & \hat{y}_N \end{pmatrix}. \tag{18}$$

Then the 3D data S_t is generated using the following formula and Figure 5 shows how it is generated.

$$S_t = \begin{pmatrix} \hat{x}_1 & \hat{x}_2 & \cdots & \hat{x}_N \\ \hat{y}_1 & \hat{y}_2 & \cdots & \hat{y}_N \\ cz_1 & cz_2 & \cdots & cz_N \end{pmatrix}, \tag{19}$$

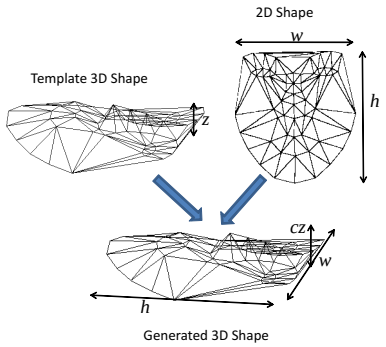


Fig. 5. The figure shows how a 3D shape generated from the 2D shape and a template 3D shape.

where $c = \min\left\{\frac{\max\{\hat{x}_i\}-\min\{\hat{x}_i\}}{\max\{x_i\}-\min\{x_i\}}, \frac{\max\{\hat{y}_i\}-\min\{\hat{y}_i\}}{\max\{y_i\}-\min\{y_i\}}\right\}_{i=1}^{N}$.

Note that the use of template 3D face is due to lack of true 3D face data. The more different template 3D faces are given the higher deformability of 3D shape model will be assured.

B. Software and Hardware

The implementation of the proposed 3D AAM is based on an existing 2D AAM-API[1] which is an open source C++ implementation of the 2D AAM framework. More details of the AAM-API can be found in [12]. The experiments are performed on an AMD Athlon 64 X2 dual core 1.99GHz machine.

C. Experiments

Our experiment consists of four parts. The first part is a person-specific experiment. In this part, all the training data and test data are collected from the same person. The test data varies only in yaw. The second part is to test the performance of the proposed algorithm on the general cases, i.e. the training and test data include more than one subject. The experiment in this part also shows the ability to align unseen subjects. The test data includes the faces vary in yaw, roll or pitch only. The third part is same as the second part but all the 10 test subjects wear glasses. The last part is to fit faces with variations both in rotations and translations. In each case, only frontal face images are used to train the proposed 3D AAM, and training images will not appear in the test dataset. After training, the mean appearance consists of about 5000 pixels.

The face rectangle in the test image is labeled manually before matching. The rough translation can be estimated according to the face rectangle along with the 3D mean shape. As the proposed algorithm applies Taylor's first order approximation, the initial guess should be close to the ground truth. In our implementation, we list the possible orientations by dividing the range of the orientation about each axis into 4 regions equally. Therefore, there are 5 possible rotation angles for each axis and there will be

[1]http://www2.imm.dtu.dk/~aam/

(a) Success

(b) Failure

Fig. 6. Examples of success and failure instances.

	Detection Rate	Iterations
Subject1	85%	30.47
Subject2	83.33%	18.6
Subject3	94.44%	24.29

TABLE I

PERFORMANCE EVALUATION OF THE SINGLE-PERSON CASE.

125 possible orientations. Assume the possible angle is from -30° to 30°, then $(\theta_x, \theta_y, \theta_z)$ is a valid orientation configuration if $\theta_x, \theta_y, \theta_z \in \{-30^\circ, -15^\circ, 0^\circ, 15^\circ, 30^\circ\}$. For every orientation configuration, 10 translation configurations are uniformly sampled within a region on the image for more accurate position and scale. The face on the image can move within a 20-pixels-by-20-pixels square and the scale can be changed by multiplying the depth of the face by a factor. To determine which configuration is the initial guess, distance from feature space(or DFFS)[13] is used. DFFS is defined as the norm of the difference between the texture and its interpretation in the appearance model. The configuration with the texture attains the lowest DFFS is then chosen as the initial guess.

The fitting algorithm is applied to align face in the test image. The fitting stops when the following condition holds.

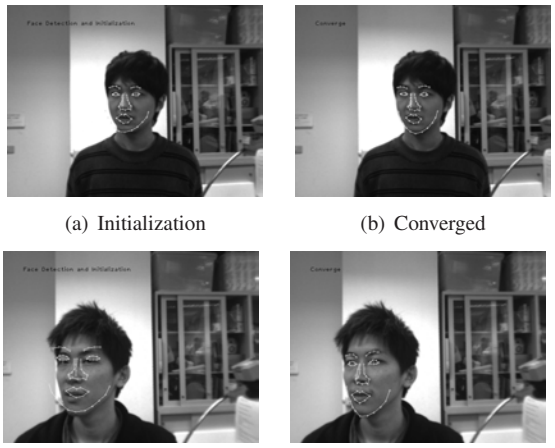
$$\left|E_{\text{i}} - E_{\text{i-1}}\right| \leq kE_{\text{i-1}}, \tag{20}$$

where E stands for error and k is constant 10^{-5}. The user judge the result by whether he/she is satisfied with the result. 10 people is asked to repeat the judgement respectively. The final detection rates is the average of the individual detection rates. Figure 6 shows examples of success and failure.

D. Results

1) The Person-Specific Case: In the first part of experiments, the 3D AAM tried to fit to test images that have different poses or slightly different facial expressions. 3 models for 3 subjects are built respectively. The results are shown in Figure 7. The performance is shown in Table I. The detection rate refers to the percentage of the number of the success in the experiment.

2) The Multi-Person Case: In the second part of the experiment, a 20 person dataset is used to verify the proposed algorithm. One frontal face for each subject is used for training. These 20 frontal face images are divided into 5 disjoint groups. Five 3D AAMs are trained respectively by



(a) Initialization (b) Converged

(c) Initialization (d) Converged

Fig. 7. the results of the person-specific case

	Range	0°-10°	0°-20°	0°-30°	0°-40°	0°-50°
Multi-Person	Yaw	93.91%	91.02%	80.22%	71.54%	67.47%
	Roll	90.43%	89.36%	79.87%	77.67%	75.30%
	Pitch	94.07%	72.41%	62.50%	55.57%	50.99%
Glasses	Yaw	78.33%	80.00%	73.18%	65.20%	65.20%
	Roll	66.67%	72.17%	68.21%	67.23%	66.00%
	Pitch	63.33%	46.25%	36.28%	31.20%	31.20%

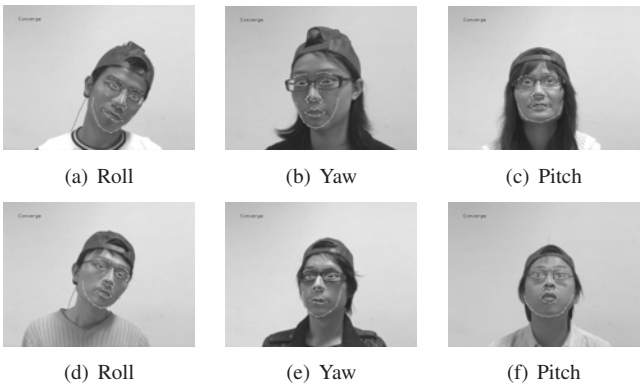
TABLE II

PERFORMANCE EVALUATIONS OF THE MULTI-PERSON AND GLASSES CASES.

(a) Roll (b) Yaw (c) Pitch

(d) Roll (e) Yaw (f) Pitch

Fig. 8. The results of the multi-person case.



(a) Roll (b) Yaw (c) Pitch

(d) Roll (e) Yaw (f) Pitch

Fig. 9. The results of glasses case.

leaving one group out. In other words, 16 subjects appear in the training stage for each 3D AAM. The reason for training 5 models is to see if our approach can work on unseen subjects. In the test stage, 15 images (5 for yaw variation, 5 for roll variation, and 5 for pitch variation) for each subject are selected to evaluate the proposed algorithm, and training images will not appear in these 15 images. In this dataset, all subjects wear a IMU on their heads to record the ground truth. The test dataset has 300 images. The test images are fitted by the algorithm under the five models respectively. The detection rate is the average of the detection rates from the 5 models. The performance is shown in Table II and the alignment results can be found in Figures 8.

The detection rates of the multi-person case are above 80% within 30° rotation in yaw and roll. The detection rates of the yaw and pitch variation cases descend could be due to self-occlusion.

3) Glasses: In this part, the 3D AAMs trained in Section IV-D.2 are used to test the images collected from 10 subjects who wear glasses. 15 images(5 for yaw variation, 5 for roll variation, and 5 for pitch variation) for each subject are selected to evaluate the algorithm. Note that all the subjects in the training images do not wear glasses. The performance can be found in Table II and the alignment results can be found in Figures 9 and 10.

It is shown that the detection rates of the subject wearing glasses are lower than the cases without glasses. The texture of the face region changes due to glasses itself. In some images, the eyes are completely occluded by the light reflection. Some results which are possibly affected by this situation are shown in Figure 10.

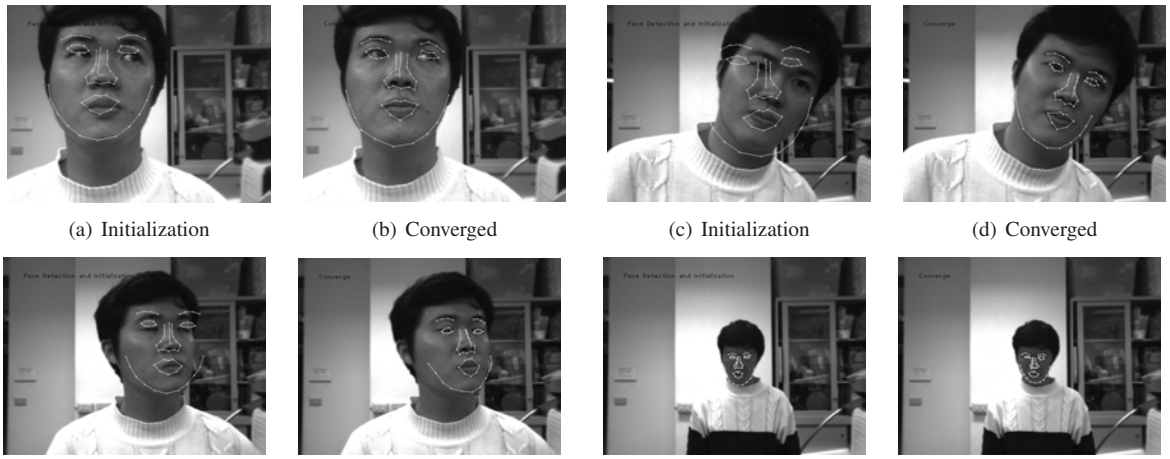
4) 3D Pose Variations: In this experiment, the cases with variations in pitch, roll, and translation are tested. The results are shown in Figure 11. These experimental results show the effectiveness of the proposed algorithms to deal with face 3D pose variations.

E. Computational Complexity

It takes about 0.69 second per iteration in our implementation. The average time (per iteration) of 2D AAM implemented in AAM-API is 19ms. The jacobian matrix in the 2D AAM is an estimate matrix learned at the training stage while it is calculated for every iteration in our framework. Calculating the jacobian matrix is a step needs a great deal of computation. In our experiment, it takes 0.61 second to calculate the warp jacobian.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed the 3D AAM framework for aligning 3D faces in 2D images. Our main contribution is to extend the search space of AAM to the 3D world and no balancing weighting between 2D and 3D shapes is needed. Moreover, only frontal faces are collected for training as faces in different poses can be generated by transforming the



(a) Initialization (b) Converged (c) Initialization (d) Converged

(e) Initialization (f) Converged (g) Initialization (h) Converged

Fig. 11. The results of face 3D variations in yaw, roll, pitch and translation.

(a) (b)

(c) (d)

Fig. 10. The fail examples of glasses case.

3D shape model. The experimental results have demonstrated that the proposed approaches are effective and capable to handle face pose variations.

The proposed approach may fail when the view is so oblique in which occlusion occurs. The occlusion issues should be of our interest. Instant response from a robot is a critical requirement for HRI. The inverse compositional algorithm [5][6] provides a way to speed up the registration process and we would like to see if the similar procedures could lead our approaches to achieve real-time performance. We would also like to improve the existing face recognition, face expressions analysis and lip reading algorithms with the proposed 3D AAM algorithm in the near future.

ACKNOWLEDGMENT

This work was partially supported by grants from Taiwan NSC (#96-2628-E-002-251-MY3, #96-2218-E-002-008, #96-2218-E-002-035); Excellent Research Projects of National Taiwan University (#95R0062-AE00-05); Taiwan DOIT TDPA Program (#95-EC-17-A-04-S1-054); Taiwan ITRI and Intel.

REFERENCES

[1] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, 2003.

[2] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 681–685, 2001.

[3] T. F. Cootes, D. H. Cooper, C. J. Taylor, and J. Graham, "Trainable method of parametric shape description," *Image Vision Comput.*, vol. 10, no. 5, pp. 289–294, 1992.

[4] A. Lanitis, C. J. Taylor, and T. F. Cootes, "Automatic tracking, coding and reconstruction of human faces using flexible appearance models," *IEEE Electronic Letters*, vol. 30, pp. 1578–1579, 1994.

[5] I. Matthews and S. Baker, "Active appearance models revisited," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 135 – 164, November 2004.

[6] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221 – 255, March 2004.

[7] R. Gross, I. Matthews, and S. Baker, "Active appearance models with occlusion," *Image and Vision Computing*, vol. 24, no. 6, pp. 593–604, 2006.

[8] G. Englebienne, T. F. Cootes, and M. Rattray, "A probabilistic model for generating realistic lip movements from speech," in *Neural Information Processing Systems*, 2008.

[9] J. Xiao, S. Baker, I. Matthews, and T. Kanade, "Real-time combined 2d+3d active appearance models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, June 2004, pp. 535 – 542.

[10] C. Goodall, "Procrustes methods in the statistical analysis of shape," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 53, no. 2, pp. 285–339, 1991.

[11] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 187–194.

[12] M. B. Stegmann, "The AAM-API: An open source active appearance model implementation," in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2003, 6th Int. Conference, Montréal, Canada*, ser. LNCS 2879. Springer, nov 2003, pp. 951–952.

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