Asian Journal of Information Technology 5(5) : 460-464, 2006

A Novel Approach for Semi-automatic Construction of 3d Facial Model from Single Orthogonal Front Image of a Human Face

Rafizul Haque, S.M., Rajib Biswas and Abu Bakar Siddique
Department of Computer Science and Engineering, Discipline Khulna University, Khulna, Bangladesh

Abstract: 3D human face models have been widely used in applications such as facial animation, facial expression recognition, face recognition, etc. Most of the techniques for constructing 3D facial model from face image use two orthogonal views of human face. This paper describes an easy method to generate 3D facial model from a single orthogonal front view of a face. The user has to manually detect the position of both eyes by marking the pupil in the single front image of the given face. Then using the distance between the two eyes and the specified ratio of the generic facial geometry, face boundary and other facial features are detected. Then the facial features and the generic 3D face model are used to generate the 3D facial model.

Key words: 3D Face modeling, automatic texture mapping, general face geometry

INTRODUCTION

Almost every animator may respond that, the most difficult character to model and animate are humans. 3D human face models have been widely used in applications such as facial animation, facial expression recognition, human action recognition, face recognition. Modeling human faces provides a potential solution to identify faces with variations in illumination, pose and facial expression. There has been a lot of work on face modeling, head modeling and morphing from images of a person. Most of the techniques for constructing 3D facial model from face image use two orthogonal views. If only the construction of a person's three-dimensional face model is concerned, a generic three-dimensional face model and image of a person's face in single orthogonal front view are required. Then the captured face image from the orthogonally viewed image is mapped onto the generic face model so that the person's three-dimensional face model can be automatically constructed.

PREVIOUS WORKS

Ming-Sing Su[1] has proposed a method to construct three dimensional face models from the person's two orthogonal views. This approach is a no hairstyle interference method, which constructs the person's face model and hair model independently. The fine facial structures of the image of the person, i.e. eyes, nose and lips, as well as the person's outline in two orthogonal views are obtained. A feature based generic three-dimensional baldhead model is then used to adjust its predefined features to fit the person's fine structures and outline morphologically in two respective views. The independently obtained three-dimensional hair model is put on the morphed baldhead model to be the person's three-dimensional geometrical face model. The person's realistic face model is finally obtained by using the texture mapping of the person's image in two orthogonal views onto the person's three-dimensional face model. Here in this method, the hair is to be extracted and generic hair model has been used. This proposal has one more limitation, which is in the background used in the image. The background has to be blue for the input image.

In the study of Islam[2], they have proposed a face detection algorithm for color images in presence of varying lightning condition with the entire image then generates facial features based on facial geometry and at last, generates face boundaries based on information collected from facial features. This method only searches the facial feature points like nose, lips and eyes are searched in only a specified area. This method needs the user to mark the face boundary though it is proposed as automatic technique.

In the study of Islam[3], they have presented an algorithm for constructing 3D face model from 2D face images. The algorithm analyzes a pair of face images-the frontal and the right side views of a face- and constructs the 3D model of that face with its real color value. Basically, the algorithm extracts the facial features and obtains the length and width of the face from the frontal face image and attains the height from the profile image. Using the length, width and height of the face, its 3D model is built. Here, in this method every snap has to be

Correspoding Author: Rafizul Haque, S.M., Department of Computer Science and Engineering, Discipline Khulna University, Khulna, Bangladesh

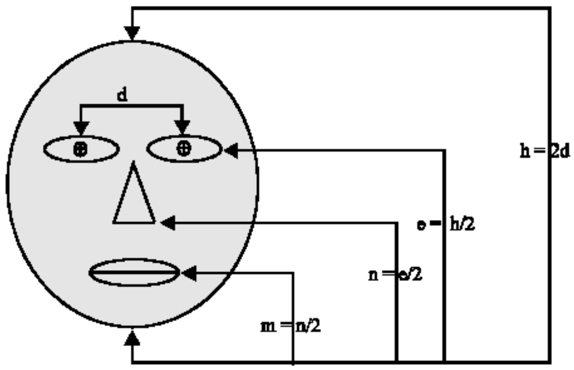
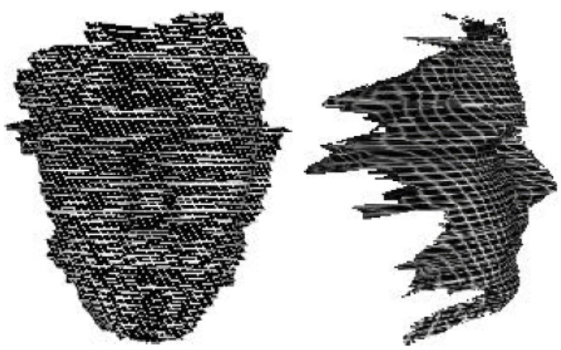


Fig. 1: General face geometry



(a) (b)

Fig. 2: (a) Front and (b) side view of the 3D mesh

(a) (b)

Fig. 3: (a) a front image (b) the resultant 3D face, of a person

Fig. 4: Some other views of the resultant 3D face

taken from the same distance. The front image and the profile image of the person must have to be taken from the same distance; otherwise the measurement of the feature points will be wrong which is being calculated from the two images. This method needs the user to mark the facial boundary.

Woon-Sook Lee[4] has proposed an efficient method to reconstruct 3D facial model for animation from two orthogonal pictures taken from front and side views or from range data obtained from any available resources. It is based on extracting features on a face in a semiautomatic way and modifying a generic model with detected feature points. Then the fine modifications follow if range data is available. Automatic texture mapping is employed using a composed image from the two images.

Woon-Sook Lee[5] describes a combined method of facial reconstruction and morphing between two heads, showing the extensive usage of feature points detected from pictures. Firstly, presents an efficient method to generate a 3D head for animation from picture data and then a simple method to do 3Dshape interpolation and 2D morphing based on triangulation. The basic idea is to generate an individualized head modified from a generic model using orthogonal picture input, then process automatic texture mapping with texture image generation by combining orthogonal pictures and coordinate generation by projection from a resulted head in front, right and left views, which results a nice triangulation on texture image. Then an intermediate shape can be obtained from interpolation between two different persons. Generating an intermediate image and new texture coordinate processes the morphing between 2D images. Texture coordinates are interpolated linearly and the texture image is created using Barycentric coordinates for each pixel in each triangle given from a 3D head. Here, the feature points are extracted in semi automatic way, but how the head model is morphed is not described. These methods do not imply that the head model is morphed automatically.

Zicheng Liu[6] presents a system which automatically generates a 3D face model from a single frontal image of a face. The system consists of two components. The first component is the feature detection and the second component is the model fitting. They used existing feature detection software for the first component. After detecting the face features, we fit a 3D face model by using a linear space of face geometries and assuming an orthogonal projection. Even though the depths of the resulting face models are usually not accurate, the models look recognizable as long as the view angle is not too far from the front. Here, this method uses existing feature detection software.

PROPOSED METHOD

Here, in this study, a new method is proposed. The proposed method is based on the easiest concept using one orthogonal front image. The feature points are detected using the location of the two eyes. The proposed method needs the user to select the location point of the eyes. From the distance of the eyes and the general facial geometry[2], the face boundary and other facial features are located. Now using these feature points and the generalized 3D facial model of the human face the face is mapped using some basic concepts.

Steps in texture mapping: To use texture mapping, these steps are performed:

- Specify the texture.
- Indicate how the texture is to be applied to each pixel.
- Enable texture mapping.
- Draw the scene, supplying both texture and geometric coordinates.

The data describing a texture can consist of one, two, three, or four elements per texel, representing anything from a modulation constant to an (R, G, B, A) quadruple. Using an advanced technique, called *mipmapping*, a single texture can be specified at many different resolutions. Also, the specification of the map can include boundary values to use when the object's texture coordinates get outside the valid range. Boundary values allow pasting together multiple texture maps smoothly, thereby increasing the effective size of the largest available texture.

There are three possible functions for computing the final RGBA value from the fragment color and the texture-image data. One possibility is to simply use the texture color as the final color; this is the *decal* mode, in which the texture is painted on top of the fragment, just as a decal would be applied. Another method is to use the texture to *modulate*, or scale, the fragment's color; this technique is useful for combining the effects of lighting with texturing. Finally, a constant color can be blended with that of the fragment, based on the texture value.

Image acquisition: The image has to be acquired orthogonal to the front face. As the image can be resized so the distance of taking snaps is not an important factor here. The images can be taken either in digital camera or manual camera. If the image is captured by digital camera then it can directly be taken as the input image. But if any manual camera then the picture takes the picture is to be scanned to take as an input image.

Detecting the position of two eyes and distance between two eyes: The contribution of this study is to propose and describe a new algorithm for detecting the facial features and to determine the 3D facial model. This method needs the user to mark the eye point. Let the user has marked the left eye at point (L_x, L_y) and the right eye at the point (R_x, R_y). As the coordinates of the eye position are almost same thus we do not consider L_y and R_y values for measuring the distance between the two eyes. To determine the distance between the two eyes we just subtract L_x from R_x. We consider the distance between the two eyes is represented by d'.

$$d' = R_x - L_x$$

From the distance of the two eyes the face boundary is detected. Other facial features are detected using this distance between two eyes.

Image resizing:
d = desired distance between two eyes.
d′ = distance between two eyes of the input image

$$\text{Scaling ratio, } r = \frac{d}{d'}$$

$P_i\ (X_i, Y_i)$ is any point from the given image and the corresponding point from the scaled image will be $P_{is}(X_{is}, Y_{is})$, where,

$$X_{is} = r \times X_i,$$
$$Y_{is} = r \times Y_i$$

Cropping the Image: $P_o\ (X_o, Y_o)$ is the coordinate in the cropped image using,

Creating 3d facial model: The eye points and the facial boundary are detected they are combined with the generic three-dimensional face model. We have a generic face model created using rectangular mesh. The feature points are matched with the feature point locations of the image. Then they are merged using pixel interpolation.

Texture mapping: The mapping from texture space to screen space is split into two phases. First is the surface parameterization that maps texture space to object space, followed by the standard modeling and viewing transformations that map object space to screen space, typically with a perspective projection. These two mappings are composed to find the overall 2D texture space to 2D screen space mapping and the intermediate

3D space is often forgotten. This simplification suggests ties of texture mapping with image warping and geometric distortion.

Scanning order: There are three general approaches to drawing a texture-mapped surface: a scan in screen space, a scan in texture space and two-pass methods. The three algorithms are outlined below:

```
Screen Scanning:
    for y
      for x
                compute u (x, y) and v (x, y)
                copy TEX [u, v] to SCR [x, y]

Texture Scanning:
    for v
      for u
                compute x (u, v) and y (u, v)
                copy TEX [u, v] to SCR [x, y]

Two-Pass:
    for v
      for u
                compute x (u, v)
                copy TEX [u, v] to TEMP [x, v]
    for x
      for v
                compute y (x, v)
                copy TEMP [x, v] to SCR [x, y]
```

Here, TEX is the texture array, SCR is the screen array and TEMP is a temporary array. It can be noted that, copying pixels involves filtering.

Screen order, sometimes called inverse mapping, is the most common method. For each pixel in screen space, the pre-image of the pixel in texture space is found and this area is filtered. This method is preferable when the screen must be written sequentially the mapping is readily invertible and the texture is random access.

Texture order may seem simpler than screen order, since inverting the mapping is unnecessary in this case, but doing texture order correctly requires subtlety. Unfortunately, uniform sampling of texture space does not guarantee uniform sampling of screen space except for affine mappings. Thus, for non-affine mappings texture subdivision must often be done adaptively; otherwise, holes or overlaps will result in screen space. Scanning the texture is preferable when the texture to screen mapping is difficult to invert, or when the texture image must be read sequentially and will not fit in random access memory.

Two-pass methods decompose a 2D mapping into two 1D mappings, the first pass applied to the rows of an image and the second pass applied to the columns. These methods work particularly well for affine and perspective mappings, where the warps for each pass are linear or rational linear functions. Because the mapping and filter are 1D they are amenable to stream processing techniques such as pipelining. Two-pass methods are preferable when the source image cannot be accessed randomly but it has rapid row and column access and a buffer for the intermediate image is available.

Perspective projection: A naive method for texture mapping in perspective is to linearly interpolate the texture coordinates *u* and *v* along the sides of the polygon and across each scanline is done. However, linear interpolation will never give the proper effect of nonlinear foreshortening. It is not rotationally invariant and the error is obvious in animation. One solution is to subdivide each polygon into many small polygons. The correct solution, however, is to replace linear interpolation with the true formula, which requires a division at each pixel. Perspective mapping of a planar texture can be expressed using homogeneous matrix notation[7]:

$$[xw \;\; yw \;\; w] = [u \;\; v \;\; 1]\begin{bmatrix} A & D & G \\ B & E & H \\ C & F & I \end{bmatrix}$$

This mapping is analogous to the more familiar 3D perspective transformation using 4 × 4 homogeneous matrices. The inverse of this mapping (calculated using the adjoint matrix) is of the same form:

$$[uq \;\; vq \;\; q]=[x \;\; y \;\; 1]\begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}=[x \;\; y \;\; 1]\begin{bmatrix} EL-HI & FG-DI & DH-EG \\ CH-BI & AL-CG & BG-AH \\ BF-CE & CD-AF & AE-BD \end{bmatrix}$$

The composition of two perspective mappings is also a perspective mapping. Consequently, a plane using a perspective parameterization that is viewed in perspective will have a compound mapping of the perspective form. For screen scanning we compute *u* and *v* from *x* and *y* as follows:

$$u = \frac{ax + by + c}{gx + hy + i}, \; v = \frac{dx + ey + f}{gx + hy + i}$$

Perspective mapping simplifies to the affine form when, which occurs when the surface is parallel to the projection plane[7].

RESULT

This approached is implemented and tested through a number of orthogonal front images. Here, an orthogonal front image and the image of the resultant 3D model are given.

DISCUSSION

Segmentation is used for facial feature detection by Ming-Sing Su[1] and ET. Al. It requires background of predefined color. Blue was the predefined color there. Our proposed method can work with any background color. Method proposed by Islam[2, 3] searches in some certain region. In the proposed method the user detects the eye positions by marking them and then other facial features and face boundary are detected as the ratio from the generic face geometry. Method proposed by Islam[2,3] used the method where every snap shot has to be taken in the same light and same distance. This rigidity is not in this proposed method. Method proposed by Zicheng[6] uses existing face detector software to detect the face. In the proposed method no external software was used to detect facial features.

CONCLUSION

A 3D facial model-building algorithm, which is efficient with respect to time and cost, has been proposed. For facial feature detection, eye points are taken and then the other points are measured instead of local search or global search of feature point detection. The searching time has been significantly decreased and it does not need any training session in this way, which is needed in neural network. Here the true color image is used. This method needs not to make the gray scale image or binary color image for feature detection. This also decreases the amount of complex calculations. As, after building the3D mesh, the face is textured instead of storing all pixels with its real color value, the amount of required memory is reduced significantly. The future work includes developing methodology dealing with glasses, mustache and beard. Moreover, it can be tried to create 3D model for other parts of the body. The system can be used instead of high-level CAD software it will be very easy to build a 3D model as we wish. The algorithm proposed in this thesis study can be used to detect facial features depending on the location of two eyes; it takes less time and cost. The modeling techniques used here are very simple to implement. For further implementation, Beizer patch can be used to make the 3D face model smoother.

REFERENCES

1. Ming-Shing Su, Chun-Yen Chen and Kuo-yng Cheng, 2002. An Automatic Construction of a Person's Face Model from the Person's Two Orthogonal Views, Proceedings of the Geometric Modeling and Processing-Theory and Applications, IEEE.
2. Islam, M.A., M.S. Imran and M.B.A. Salam, 2003. A Multifarious Faces and Facial Detection Algorithm for color images Based on Grey-scale Information and Facial Geometry, 6th Intl. Conference on Computer and Information Technology (ICCIT 2003), Dhaka, Bangladesh, pp: 623-626.
3. Islam, M.A., M.S. Imran and M.B.A. Salam, 2003. A Novel Approach for Construction of 3D Facial Model from Two Orthogonal Images of a Face, 6th Intl. Conference on Computer and Information Technology (ICCIT 2003), Dhaka, Bangladesh, pp: 623-626.
4. Won-Sook Lee and Nadia Magnenat-Thalmann, 2000. Fast head modeling for animation, Image and Vision Computing (IVC), pp: 355-364,
5. Won-Sook Lee, P. Karla and Nadia Magnenat-Thalmann, 1997. Model Based Face Reconstruction For Animation, In Proc. Multimedia Modeling (MMM) '97, Singapore, pp: 323-338.
6. Zicheng Liu, 2003. A Fully Automatic System to Model Faces From a Single Image, Microsoft Research, Technical Report MSR-TR-2003-55.
7. Paul Heckbert, S., 1986. SURVEY OF TEXTURE MAPPING, IEEE Computer Graphics and Applications, Nov. 1986, pp. 56-67 and Graphics Interface'86, pp: 207-212.