Appears in Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, October 1996

Facial Analysis and Synthesis Using Image-Based Models

Tony Ezzat Tomaso Poggio
tonebone@ai.mit.edu tp-temp@ai.mit.edu
MIT Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

Abstract

In this paper, we describe image-based modeling techniques that make possible the creation of photo-realistic computer models of real human faces. The image-based model is built using example views of the face, bypassing the need for any three-dimensional computer graphics models. A learning network is trained to associate each of the example images with a set of pose and expression parameters. For a novel set of parameters, the network synthesizes a novel, intermediate view using a morphing approach. This image-based synthesis paradigm can adequately model both rigid and non-rigid facial movements.

We also describe an analysis-by-synthesis algorithm, which is capable of extracting a set of high-level parameters from an image sequence involving facial movement using embedded image-based models. The parameters of the models are perturbed in a local and independent manner for each image until a correspondence-based error metric is minimized.

A small sample of experimental results is presented.

1 Introduction

Facial analysis and synthesis have emerged to be two important requirements for a vast array of vision-based applications. Facial analysis refers to the extraction from video sequences of information concerning the location of the head, its pose, and the movement of facial features such as the eyes and the mouth. Facial synthesis refers to the reverse process of animating a facial model using a set of high-level parameters that control the face's gaze, mouth orientation, and pose.

In this work, we adopt an *image-based* facial modelling approach, in which we model facial movements using *example images*. In doing do we are motivated by the fact that a number of researchers ([2], [3], [8], [9]) have noticed the viability of a *view interpolation* approach to image synthesis, where novel, intermediate images of a scene are synthesized from example endpoints using a morphing technique. In particular, we adopt the approach of Beymer, Shashua, Poggio [2], who cast the view interpolation approach in a *learning-by-example* framework: each example image is associated with a position in a high-level, multi-dimensional parameter space denoting pose and expression. By training on the examples, a learning network can then generalize, and generate suitable novel images that lie at intermediate points in the example space. The trained network, in essence, becomes a *synthesis network*, which generates images as output, for suitable parameters as input. Beymer, Shashua, Poggio [2], in fact, showed that this technique is capable of modeling rigid facial transformations such as pose changes, as well as non-rigid transformations such as smiles.

From the analysis standpoint, we are motivated in particular by the work of Jones and Poggio [6], who constructed models of line drawings, and used a stochastic gradient descent algorithm to match the models to novel line drawings input by the user. The models themselves consisted of a linear combination of prototypes, and the error metric which the gradient descent algorithm tried to minimize was the pixel-wise error between the novel drawing and the current guess for the closest model image. At every iteration, the algorithm would compute the gradient of this error metric with respect to the model parameters, and proceed to a new guess for a set of parameters that would produce a new model image closer to the novel image.

The first contribution of this work is to extend the *synthesis network* paradigm of Beymer, Shashua, Poggio [2] into a *synthesis module* paradigm more suitable for analysis: Firstly, each synthesis network is additionally parameterized with a set of affine parameters, such as translation, rotation, and scale. Secondly, a flexible mask-based segmentation scheme is incorporated into the synthesis module that is capable of segmenting the head in any of the images output by the network. Thus, from an input-output perspective, the synthesis module is capable, for the appropriate input parameters, of producing images of segmented faces at various

scales, rotations, positions, poses, and expressions.

The second contribution of this work is to embed the synthesis modules mentioned previously in an analysis-by-synthesis algorithm similar to that of Jones and Poggio [6]. In our case, however, we define a *correspondence-based error metric* instead of a pixel-based error metric, in an attempt to make the analysis algorithm more robust to changes in lighting, position, scale, and rotation. Essentially, the parameters of the embedded synthesis modules are perturbed in a local and independent manner for each image in the sequence until the correspondence-based error metric is minimized.

In Section 2, we describe the construction of the synthesis modules to be used for analysis and synthesis. In Section 3, we sketch an outline of our analysis-by-synthesis algorithm. In Section 4, we describe and depict a small sample of experimental results. Finally, in Section 5, we briefly discuss our approach.

2 Building the Synthesis Modules

2.1 Choosing the Example Set and the Parameter Space

The first step in the creation of the synthesis module is the selection of the example images and the association of each example with a point in a hand-crafted parameter space x. Figure 1 depicts nine example images arranged in a two-dimensional parameter space where each axis is limited to values between -1.0 and 1.0. One axis denotes degree of *horizontal pose*, while the other denotes degree of *vertical pose.* The top-right example image, for instance, would be associated with the position in parameter space $x = (1.0, 1.0)$.

2.2 Learning the Map from Parameters to Correspondences

Beymer, Shashua, and Poggio [2] framed the learning problem as a problem of approximating an unknown function $y = f(x)$ that maps between the parameter space, x, and the example space, y, given a set of N training samples $(x_i,\ y_i)$ of the function $f(x)$. Rather than trying to approximate a function $y = f(x)$ that maps between the parameter space x and an example space y *of images*, Poggio and Brunelli [7] instead argued that it is better to try to learn a map between a parameter space x and an example space y *of correspondence vectors* that define corresponding features across the example images. The underlying intuition is that such a map is easier to learn because the correspondence vectors factor out lighting effects, and also because they undergo reasonably smooth change during motion of the underlying object to be modeled.

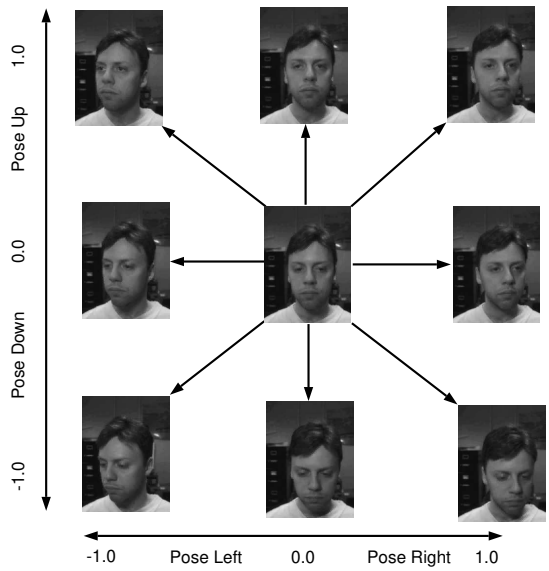

Figure 1. The examples for a 3-by-3 network involving pose movements in all directions.

In this work, we define a *dense*, *pixel-wise* correspondence between two images: for a pixel in image A at position (i, j), the corresponding pixel in image B lies at position $(i + \Delta x(i, j), j + \Delta y(i, j))$, where Δx and Δy are arrays that contain the x and y components of the correspondence vectors, respectively. To obtain such a dense, pixel-wise correspondence between the example images, we utilize optical flow algorithms borrowed from the computer vision literature. Specifically, we use the coarse-to-fine, gradient-based optical flow algorithms developed by Bergen and Hingorani [1]. In practice, these algorithms work very well when the two example images are similar and not too far apart. In cases where the example images are far apart, we have found that concatenating optical flow by using many intermediate images between examples produces very good final correspondences.

From the standpoint of synthesis module design, in which more than two images may be involved, a *reference example* image is designated, and correspondence between it and the rest of the images in the example set is found. For example, in Figure 1, the central image is the reference example, and eight correspondence vectors y_i are found between it and the other examples. A ninth, and null, correspondence vector, y_0, is designated to represent the correspondence between the reference example and itself.

We approximate the unknown function $y = f(x)$ which maps from parameters to correspondences given the samples $(y_i, x_i)_{i=1}^{N}$, using a *radial basis function with Gaussian centers* [5], which is expressed in its *dual form* as:

Figure 2. Some intermediate examples generated from the synthesis network of Figure 1, and their positions in the imposed parameter space

$$y = \sum_{i=1}^{N} b_i(x) y_i, \tag{1}$$

where the y_i are the example correspondences used for training. Equation 1 essentially means that a novel correspondence vector y at position x in the imposed parameter space is synthesized by taking a linear combination of the example correspondences y_i, with the coefficients b_i depending nonlinearly on the parameter x. The learning stage defines the structure of the b_i coefficients; typically, they are Gaussian-like in nature, centered around each of the example parameters x_i used for training.

2.3 Warping and Blending

A new correspondence vector y synthesized from Equation 1 thus defines a *position* in correspondence space that we would like the novel, intermediate image to be located at. A simple *forward warp* operation that pushes the pixels of the reference example image along the synthesized correspondence vector is sufficient to generate the image. To utilize the image texture from *all* the examples, however, we also adopt a simple correspondence re-orientation procedure, described in [2], that re-orients the synthesized correspondence vector from Equation 1 so that it originates from each of the other example images and points to the same position as the original synthesized correspondence. This allows us to subsequently forward warp all the examples along their respective re-oriented correspondence vectors.

The final stage of the synthesis process is the *blending* stage, when all of the warped images are blended together to produce the final image. Blending refers to scaling each image with a blending coefficient, and then adding all the scaled images together to form the final image. The blending coefficients chosen are normalized versions of the coefficients $b_i(x)$ from Equation 1.

Figure 2 illustrates three novel images synthesized from the network shown in Figure 1.

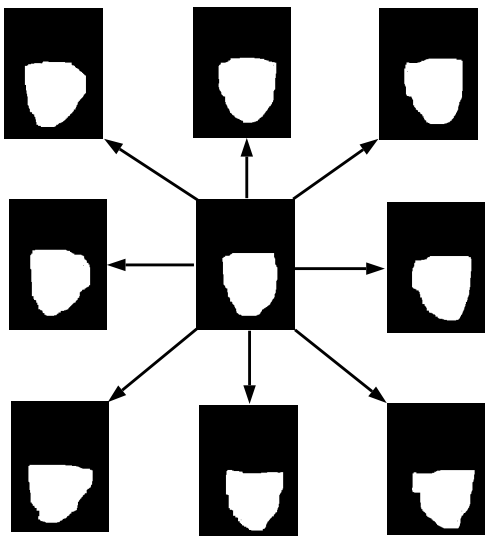

Figure 3. The masks associated with the 3-by-3 pose network in Figure 1.

2.4 Affine Parameters

Before analyzing with respect to novel image sequences, the synthesis networks must also be additionally parameterized with a set of affine parameters. This is needed because novel sequences involve movements of the head that are at scales, positions, and rotation angles that are different from those in the network. Augmenting the synthesis networks with a set of four affine parameters (two translation parameters, an angle parameter, and a scale parameter) is straightforward: the network first synthesizes the head at the intrinsic parameters imposed by the user, and then it performs the desired affine transformation.

2.5 Segmentation

In addition to augmenting the synthesis network with a set of affine parameters, it is also necessary to incorporate segmentation, because the analysis algorithm needs to match only on the region in the synthesized network that corresponds to the face. This will allow the algorithm to be less sensitive to background changes, as well as hairstyle and clothing changes.

A *network scheme* for *flexible* segmentation was adopted where a network of the same dimensions and orientation as the corresponding image synthesis network is created, except that instead of images, the examples are masks. Each mask example serves to segment the head for the corresponding image example, and the correspondence flows relating the masks together are the same as those within the image synthesis network. Whenever the synthesis net-

Figure 4. Various segmented and affine-perturbed images synthesized from a 3-by-3 pose network similar to the one shown in Figure 1.

work synthesizes a new image, it also synthesizes a new mask appropriate for the same image using the same warping and blending technique described in Section 2.3, with minor modifications to preserve the black-and-white pixel integrity of the mask.

Figure 3 depicts the masks that would be associated with the 3-by-3 pose network in Figure 1. Figure 4 shows various segmented and affine-perturbed images which are synthesized from a network similar to the 3-by-3 pose network of Figure 1.

2.6 From Synthesis Networks to Synthesis Modules

We can thus begin to conceptualize a *synthesis module* that, from an input-output perspective, can generate images of a face at a various positions, rotations, scales, poses, expressions, etc., for the appropriate set of input parameters.

As described in [4], several different synthesis modules were constructed to illustrate the ability of our approach in modeling various facial motions: in addition to the 9-example pose network shown in Figure 1, we have constructed a 5-example network that synthesizes intermediate images lying along open-mouth/smile axes, and a 14-example network that synthesizes pose, eye, and mouth movements combined.

3 Analysis

3.1 Overview

In this section, a model-based analysis algorithm is outlined which is capable of extracting a set of high-level parameters from novel image sequences. The analysis approach is, in fact, an *analysis-by-synthesis* approach, where the synthesis modules created in the previous section are themselves used for analysis.

3.2 A Correspondence-Based Error Metric

A key feature of our analysis algorithm is that instead of using the embedded synthesis module to synthesize *images* to match to the novel images, and thereby have to rely on an *image-based* error metric, as in [6], the algorithm instead tries to match novel *correspondence*. For every iteration, the algorithm computes the optical flow between two consecutive novel frames, and then attempts to find the best matching correspondence from within its embedded synthesis module. The rationale for using a correspondence-based metric, as opposed to an image-based metric, is that trying to minimize a correspondence-based error metric is less susceptible to noise, local minima, and lighting changes.

3.3 Parameter Perturbation Strategy

The analysis-by-synthesis algorithm is based on *iterative, local, independent perturbations of the synthesis parameters*. A sketch of the steps of the algorithm are as follows:

1. For a novel correspondence obtained from two consecutive novel images (say images A and B) in the sequence, the parameters of the embedded synthesis model are perturbed. The perturbations include the affine parameters, and vary each parameter independently in the positive and negative directions by a small *delta* factor.

2. For each set of perturbed parameters, the algorithm then synthesizes a correspondence from the module that corresponds to the perturbation. For reasons described in [4], we have opted to obtain the correspondence associated with the perturbation by synthesizing the two perturbed images first, and then computing optical flow between them.

3. The algorithm then computes the Euclidean distance between each perturbed correspondence and the novel correspondence, and finds the closest synthesized correspondence of the set. All distances are computed only in the regions specified by the masks associated with the perturbed correspondences.

4. The algorithm then repeats steps 1 through 3, iteratively perturbing around the set of parameters associated with the closest synthesized correspondence found in step 3.

5. For each iteration, the synthesized correspondence that yielded the overall smallest distance with respect to the novel correspondence is preserved; if a set of perturbations do not yield any new correspondences that reduce the overall minimum, the delta factors are halved and the iterations proceed once again. Thus when the algorithm gets close to the optimum synthesized correspondence, it proceeds with smaller and smaller perturbations to achieve a better match. The iterations terminate when the delta factors have been reduced to a degree where perturbations made using those factors do not make any significant changes in the synthesized correspondences.

6. Once a parameter estimate is obtained for the given novel flow, the algorithm computes the next consecutive novel correspondence in the sequence (say, between images B and C), and starts to perturb around the set of parameters found in the previous iteration. This whole process is performed across the entire sequence.

The first image in the novel sequence needs to be treated differently from the other images, since there is no prior flow within the sequence itself against which to match. Consequently, we compute the correspondence from the reference image in the network to the image, and then apply the iterative parameter perturbation technique to find the closest synthesized correspondence. This strategy suffers from the weakness that if the optical flow fails due to the fact that the heads are too far away from each other, then the extracted parameters for the first image will be incorrect. Consequently, in the novel sequences that we used to test the analysis algorithm on, the head in the initial frame was not placed too far away from the head in the reference image of the embedded synthesis module, although, of course, significant deviations in many variables did exist nevertheless.

4 Results

A varied set of experiments were performed to test our analysis-by-synthesis technique in estimating different facial movements such as pose movements, eye movements, mouth movements, as well as head translations, rotations, and scales. Figures 5 through 10 on the next page depict a small sample of results from these experiments. The novel sequences involved changes in lighting, position, scale, rotation, background, clothing, and hairstyle (the author shaved his head!).

5 Discussion

One difficulty with the analysis-by-synthesis algorithm presented is that it is computationally inefficient, taking between a few minutes to half an hour to analyze one frame, depending on the complexity of the underlying synthesis module. In addition, it also seems that the analysis-by-synthesis paradigm as presented is also strongly *user-dependent*: the example-based models can only extract analysis parameters *reliably* from faces whose examples were used to build the model.

On the positive side, however, it seems that our decision to use a correspondence-based metric, in addition to the incorporation of affine perturbation and segmentation, have allowed us to achieve very good analysis in spite of changes in background, lighting, hairstyle, position, rotation, and scale. Furthermore, it seems that the analysis-by-synthesis technique is fairly general, and can serve to analyze a wide variety of rigid and non-rigid facial movements, which would be useful for many tasks such as eye-tracking, facial expression recognition, visual speech understanding, and pose estimation.

References

[1] J. Bergen and R. Hingorani. Hierarchical motion-based frame rate conversion. Technical report, David Sarnoff Research Center, Apr. 1990.

[2] D. Beymer, A. Shashua, and T. Poggio. Example based image analysis and synthesis. Technical Report 1431, MIT AI Lab, 1993.

[3] S. E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH '93 Proceedings*, pages 279–288, Anaheim, CA, August 1993.

[4] T. Ezzat. Example-based analysis and synthesis for images of human faces. Master's thesis, Massachusetts Institute of Technology, 1996.

[5] F. Girosi, M. Jones, and T. Poggio. Priors, stabilizers, and basis functions: From regularization to radial, tensor, and additive splines. Technical Report 1430, MIT AI Lab, June 1993.

[6] M. Jones and T. Poggio. Model-based matching of line drawings by linear combinations of prototypes. In *Proc. of International Conference on Computer Vision*, pages 531–536, Cambridge, Massachusetts, June 1995.

[7] T. Poggio and R. Brunelli. A novel approach to graphics. Technical Report 1354, MIT AI Lab, 1992.

[8] S. M. Seitz and C. R. Dyer. Physically-valid view synthesis by image interpolation. In *Proc. IEEE Workshop on the Representation of Visual Scenes*, pages 18–25, Cambridge, Massachusetts, June 1995.

[9] T. Werner, R. D. Hersch, and V. Hlaváč. Rendering real-world objects using view interpolation. In *Proc. of International Conference on Computer Vision*, pages 957–962, June 1995.

Figure 5. A novel sequence with pose movement (top), along with the synthesized sequence (bottom). The synthesis module is the 9-example 3-by-3 pose network shown in Figure 1 in this paper.

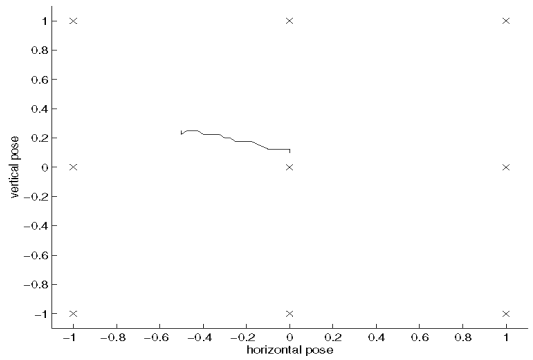

Figure 6. The pose parameters extracted from the sequence in Figure 5. The "x" marks denote the positions of the 9 examples in pose space. Affine parameters are not shown.

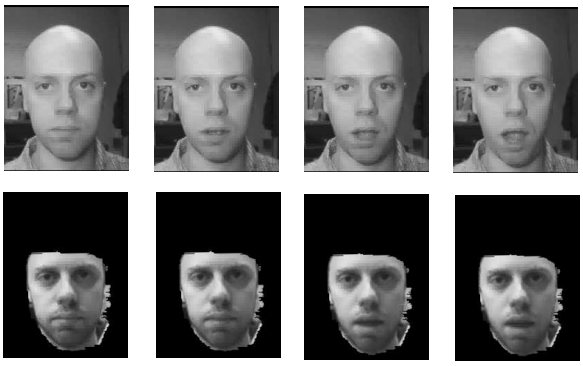

Figure 7. A novel sequence with mouth movement (top), along with the synthesized sequence (bottom). The synthesis module is a 14-example network that can synthesize eye, pose, and mouth movement.

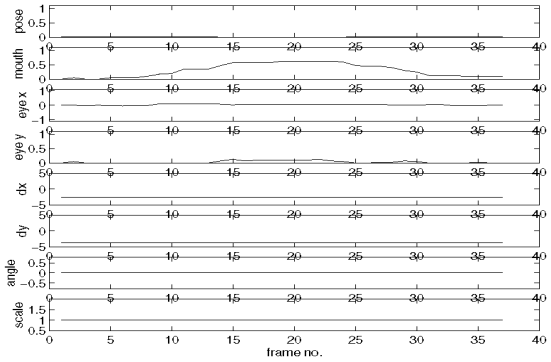

Figure 8. The complete set of parameters extracted from the sequence in Figure 7. All the activity occurs in the mouth parameter, which denotes degree of openness.

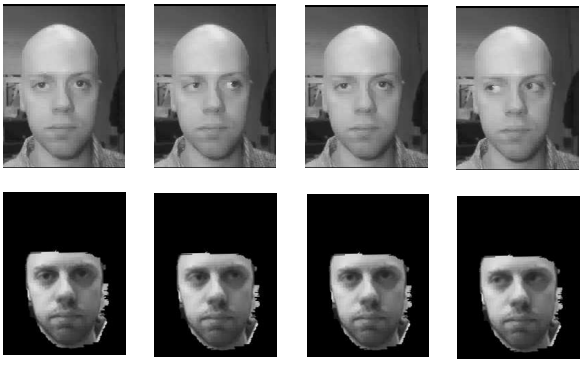

Figure 9. A novel sequence with eye movement (top), along with the synthesized sequence (bottom). The synthesis module is the 14-example network used in Figure 7.

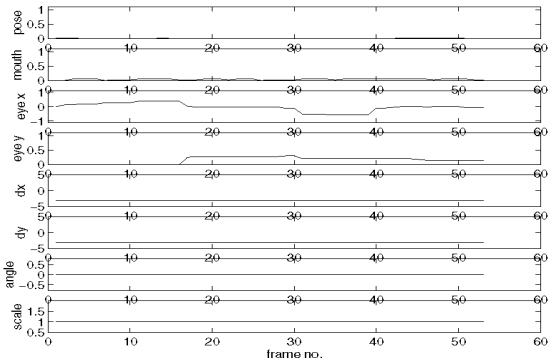

Figure 10. The complete set of parameters extracted from the sequence in Figure 9. All the activity occurs in the eyes x and y parameters.