Appears in the Proceedings of the Workshop on the Algorithmic Foundations of Robotics, August 1996

Facial Analysis and Synthesis Using Image-Based Models

Tony Ezzat, *MIT Artificial Intelligence Laboratory, Cambridge, MA, USA*
Tomaso Poggio, *MIT Artificial Intelligence Laboratory, Cambridge, MA, USA*

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 view 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. Facial analysis would be useful for such applications as eye-tracking, facial expression recognition, and visual speech understanding. Facial synthesis would be useful for animating cartoon characters or digital actors. Together, facial analysis and facial synthesis *in tandem* would be useful for model-based coding applications such as video email and video-teleconferencing, as well as interactive animation of cartoon characters using facial motions.

Many of the attempts at facial analysis and synthesis involve modeling the human face in three dimensions using computer graphics techniques. In this work, we adopt an *image-based model*, whose basis is to completely forego any underlying computer graphics models, and instead model the face using *example images*. Within the image-based synthesis literature, a number of researchers ([4], [7], [15], [16]) 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 this work, we adopt the particular approach of Beymer, Shashua, Poggio [4], 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 [4], 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 who constructed models of line drawings [11] and faces [12], and used a stochastic gradient descent algorithm to match the models to novel line drawings or faces input by the user. The models themselves consisted of a linear combination of prototypes [14], 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 [4] 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 [11]. 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, rotation, and hairstyle. 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 basic synthesis networks to be used for analysis and synthesis. In Section 3, we describe additional techniques that allow for more complicated synthesis networks to be constructed. In Section 4, we transition from the synthesis network to the synthesis module, and sketch an outline of our analysis-by-synthesis algorithm. In Section 5, we describe and depict a small sample of experimental results. Finally, in Section 6, we briefly critique our approach, and discuss future work.

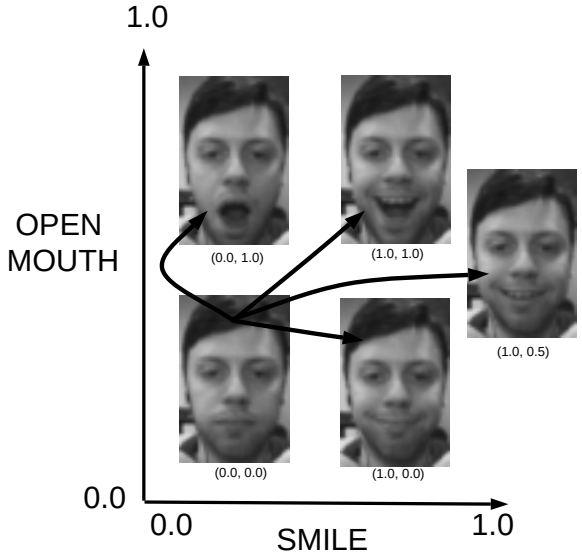

Figure 1: *A 5-example, 2-dimensional example set in a smile/open-mouth configuration.*

2 Building the Synthesis Networks

2.1 Choosing the Example Set and the Parameter Space

The first step in the creation of the synthesis network 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 five example images arranged in a two-dimensional parameter space where each axis is limited to values between 0.0 and 1.0. One axis denotes degree of *smile*, while the other denotes degree of *mouth openness*. 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

Given the example images and the associated parameters, the desired task is to generate novel intermediate images lying in the space spanned by the examples. Beymer, Shashua, and Poggio [4] re-cast this task as a *learning problem*, in which it is necessary to learn 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 $f(x)$.

Learning such a function would allow one to generalize the function at points other than the example points, and hence synthesize appropriate novel intermediate images.

Poggio and Brunelli [13], however, made the crucial observation that trying to approximate a function $y = f(x)$ that maps between the parameter space x and an example space y *of images* would probably not work due to the discontinuous nature of the underlying map. Instead, Poggio and Brunelli [13] 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.

A helpful, and often important, way (discussed in Beymer [3]) to think about the distinction between images and correspondences is to view correspondence as a way to sample the *motion* (or shape) of an object between two views, and to view images as a way to sample the object's *texture* for those views. Synthesizing novel intermediate correspondences is thus equivalent to synthesizing novel intermediate motions (or shapes) of the face.

2.3 Defining and Obtaining the Correspondence

In this work, a *dense*, *pixel-wise* correspondence is defined 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 or matrices that contain the x and y components of the correspondence vectors, respectively. In the rest of this paper, we use the symbol y to refer to both the x- and the y-components of the correspondences, and the reserve symbol x to refer to the imposed multidimensional parameter space. Equations involving the use of y imply that they are performed twice: once on the x-components and once on the y-components.

From the standpoint of synthesis network 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 bottom-left image is the reference example, and four correspondence vectors y_i are obtained between it and the other examples. A fifth, and null, correspondence vector, y_0, is designated to represent the correspondence between the reference example and itself.

To obtain such a dense, pixel-wise correspondence between the example images, optical flow algorithms borrowed from the computer vision literature are utilized. We specifically use the coarse-to-fine, gradient-based optical flow algorithms developed by Bergen and Hingorani [2], which have yielded good results in practice. In cases where they have not yielded good results, such as in cases when there is significant movement or occlusion, we have found [9] that concatenating optical flow between a set of intermediate images improves the final correspondences dramatically.

2.4 Constructing the Mapping Function

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*:

$$f(x) = \sum_{\alpha=1}^{n} c_\alpha G(\|x - t_\alpha\|) \tag{1}$$

where

$$G(x) = e^{-\frac{x^2}{\sigma^2}} \tag{2}$$

and the $t'_\alpha s$ are arbitrary parameters termed *centers*.

A radial basis function as defined in Equation 1 was shown in [10] to be a type of *regularization network* which incorporates *a priori* knowledge about the smoothness of the function that can be learned from a set of samples. Such a smoothness prior is necessary because the problem of generalizing a function from a set of samples is inherently ill-posed, and many functions may be found which pass through the sample points. The approximating function in Equation 1 is chosen from among all the possible solutions because it is simultaneously close to the data samples and the smoothness constraints.

The learning stage of a radial basis function consists of the specification of three sets of variables: the centers t_α, the σ's of the Gaussians, and, most importantly, the coefficients c_i.

In this work, we associate one example parameter x_i with each center t_α, so the approximating function $f(x)$ may be rewritten as

$$f(x) = \sum_{i=1}^{N} c_i G(\|x - x_i\|) \tag{3}$$

where N denotes the total number of examples used to train the network. One can now visualize a Gaussian center associated with each example parameter x_i used to train the network.

The sigmas of the Gaussians, which denote the width of their influence, are determined using an *average inter-example distance strategy*. For each Gaussian, the average distance between its associated example parameter and all the other example parameters is found. The final sigma value for that Gaussian is chosen to be some fixed constant times the resulting average.

Finally, the c_i coefficients are chosen in a manner that minimizes the empirical error between the approximating function $f(x)$ and the sample points $(y_i, x_i)_{i=1}^{N}$. If we substitute all the sample pairs into the Equation 3 we obtain the equation

$$Y = CG \tag{4}$$

where

$$Y = [\ y_1 \quad y_2 \quad \ldots \quad y_N\ ], \tag{5}$$

$$C = [\ c_1 \quad c_2 \quad \ldots \quad c_N\ ], \tag{6}$$

and

$$G = \begin{bmatrix} G(\|x_1 - x_1\|) & G(\|x_2 - x_1\|) & \ldots & G(\|x_N - x_1\|) \\ G(\|x_1 - x_2\|) & G(\|x_2 - x_2\|) & \ldots & G(\|x_N - x_2\|) \\ \vdots & & \ddots & \vdots \\ G(\|x_1 - x_N\|) & G(\|x_2 - x_N\|) & \ldots & G(\|x_N - x_N\|) \end{bmatrix} \tag{7}$$

The coefficients C are then determined by computing

$$C = YG^{+} \tag{8}$$

where G^{+} is the pseudo-inverse of G.

2.5 The Dual Representation for Synthesis

It is extremely helpful to rewrite the approximation function in Equation 3 into its *dual representation*, which illustrates the nature of its interpolative properties. Continuing from Equation 3, we have

$$y(x) = Cg(x) \tag{9}$$

where

$$g(x) = [\ G(\|x - x_1\|) \quad G(\|x - x_2\|) \quad \ldots \quad G(\|x - x_N\|)\ ]. \tag{10}$$

Substituting Equation 8 into Equation 9 we obtain

$$y(x) = YG^{+}g(x). \tag{11}$$

Gathering the terms not related to Y together, we have

$$y(x) = \sum_{i=1}^{N} b_i(x) y_i \tag{12}$$

where

$$b_i(x) = (G^{+})_i g(x). \tag{13}$$

Equation 12, which represents the dual representation of Equation 3, is arguably the most central equation for synthesis. Equation 12 represents any novel intermediate correspondence vector y as a *linear combination* of the N example correspondence vectors y_i. The coefficients of the combination, $b_i(x)$, depend nonlinearly on the parameter x. The learning stage defines the structure of the b_i kernels, which are typically Gaussian-like in nature, centered around each of the example parameters x_i used for training. This is not surprising given the approximation of $y = f(x)$ using a radial basis function with Gaussian centers.

2.6 Warping

A new correspondence vector y synthesized from Equation 12 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 a novel intermediate image, but such an approach would not utilize the image texture from *all* the examples in the network. To utilize the image texture

from all the examples, we adopt a correspondence re-orientation procedure, described in [4], that re-orients the synthesized correspondence vector from Equation 12 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 forward warp algorithm used does not explicitly treat *pixel overlaps*, or *folds*, in any special way, since there is no a-priori visibility model [16] built into the algorithm, unlike [6] and [7]. Hence, the order of the warp is a simple *top-down, left-to-right* order. Pixel destination values are rounded to the nearest integer location.

2.7 Hole-filling

Since the correspondences produced by the optical flow algorithms are not strictly one-to-one mappings, forward warping usually exposes regions in the warped image that are unfilled. These regions, called *holes*, must be explicitly treated, since they lead to noticeable degradations in the quality of the final images that are synthesized. In particular, holes due to local image expansion and inaccuracies in the optical flow algorithms due to lack of discriminating texture usually lead to small *specks* in the warped image. These holes are identified and eliminated as in [7], by filling the warped image with a reserved "background" color prior to warping. For those pixels which retain the background color after the warp, new colors are computed by interpolating the colors of the adjacent non-background colors.

2.8 Blending

Warping all the example images along the re-oriented correspondence vectors results in a set of warped images that need to be combined to produce the final image, which is done using *blending*. Blending refers to multiplying each image with a blending coefficient, and then adding all the scaled images together to form the final image. The blending coefficients chosen are the same as the coefficients $b_i(x)$ from Equation 12. Intuitively, the blending coefficient associated with a particular example image decreases with the distance of the novel synthesis image from the example image. Consequently, more weight in the blending stage is given to the warped images from the closer examples.

Figure 2: *Two sets of intermediate, novel images generated from 1-dimensional, 2-example synthesis networks for smile (top) and rightwards pose (bottom). The original images are the leftmost and rightmost images for each synthesis segment.*

2.9 Results and Discussion

Figure 2 illustrates a number of novel images synthesized from two one-dimensional networks. The original images for both networks are the leftmost and rightmost images. The correspondences were obtained using direct optical flow estimation between both endpoint images.

Figure 3 illustrates a large number of novel images synthesized from the two-dimensional, 5-example network shown in Figure 1. The correspondences were obtained by concatenating optical flow between a number of intermediate images.

It is important to note that the combination of warping and blending, also known as morphing, is more powerful than either technique on its own. Blending alone can generate intermediate images, but a sense of *movement* between images will be lacking. Warping alone will expose the deficiencies of the optical flow algorithms, and particularly in our case their linearization errors: the flow estimates at each pixel are only a *linear* approximation to the actual flow. As a result, warping from one image by itself will lead to suitable intermediate images *only* when the parameters are close to the parameters of the example image, but will lead to incorrect images as the parameters move farther away. Warping from all the examples combined

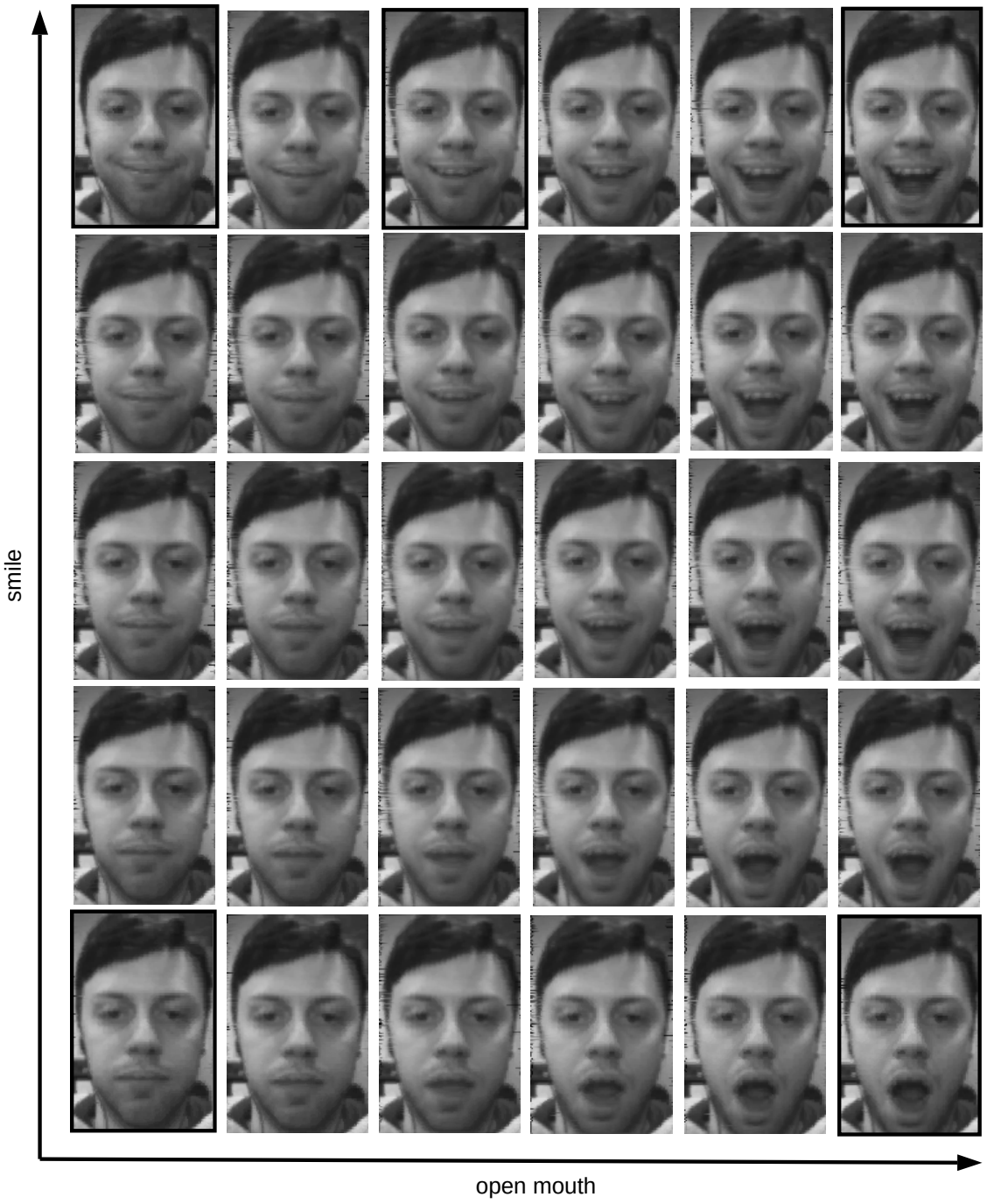

Figure 3: *Examples of the intermediate, novel images generated from the 2-dimensional, 5-example synthesis network for facial expressions shown in Figure 1. The original images are high-lighted with darker borders.*

with weighted blending, however, eases the linearization errors because, as the parameters move farther away from one example image, the pixel motion and pixel values of another example image begin to take effect.

It is also extremely heartening that a technique that combines warping, blending, and concatenated optical flow can lead to results that are good for cases in which large occlusions are present, as was the case in Figure 3. This is considerably surprising especially in light of the fact that our optical flow algorithms have no a-priori visibility model [16] built into them, as mentioned earlier.

3 Other Types of Networks

Although the synthesis network paradigm described above is adequate for modeling a large number of facial motions, it is necessary to augment it with additional techniques to address certain issues. In this section, we describe those issues, and the techniques adopted to address them.

3.1 Regional Networks

One of the problems associated with the example-based synthesis paradigm explored in this work is that a large number of example images are needed whenever a new dimension is added to a synthesis network. For example, suppose we are modeling 6 eye positions and 4 mouth positions. Modeling a fifth mouth position would require 6 additional examples, one for each separate modeled eye position.

One approach to alleviate the need for such a large number of example images is to create separate, *regional* networks for different parts of the face that move independently of each other. Such an approach would decorrellate the eye-mouth network described above into two separate, regional networks: one regional eye network composed of 6 example images modeling the various eye positions, and one regional mouth network composed of 4 images modeling the various mouth positions. Modeling a fifth mouth position would thus require only one additional example image.

Regional decomposition needs to address two issues: how to specify which regions each network controls, and how to combine the synthesized outputs of all the regional networks back together again.

A *mask-based* approach was adopted to specify which regions of the face each network controls. At the outset of the example set selection, the example set designer uses a special tool to "mask out" which region of the face each network controls. The mask produced by the tool is essentially a binarized image. During synthesis, a navigational mechanism first determines which parameters have changed relative to previous parameters, and identifies which regional network is activated. The parameters associated with that regional network are then used to synthesize an image. The mask associated with that regional network is then used to extract the appropriate portion of the synthesized image.

To combine the masked regions back together again, a simple *paste* approach was adopted, where the regions are pasted on top of a *base image* of the face. This approach works extremely well if the motion is contained *within* the regions themselves. Ideally, one would want to *blend* the regions onto the base image using more sophisticated techniques.

As an example, a regional network was constructed for left eye motions, right eye motions, and mouth motions, as shown in Figure 4. The regional left and right eye networks were composed of the *same* six images placed in a 2-dimensional arrangement, with different masks for each eye regional network. The mouth regional network consisted only of two examples to model an opening mouth. The mask for the mouth network consisted of all the pixels not contained in the left and right eye regional networks; this approach enables one to avoid creating a mask with a more explicit segmentation of the mouth region, which is hard to do because mouth movements affect a large portion of the face. The masked outputs of each regional network are pasted onto the base image shown in the center of the figure.

The gain in possible eye-mouth configurations given the number of example images is now much higher than in a standard approach not involving regional networks. Using only 6 original eye images (since the left and right eye regional networks use the same images) and 1 additional open-mouth image (since the reference image is the same for all the regional networks), various combinations of eye-mouth positions may be synthesized, as shown in Figure 5. This added flexibility is also a boon for the example set designer, who needs fewer example images to build the desired synthesis model. On

the other hand, the example set designer now needs to specify the mask regions.

3.2 Composed Networks

Another problem with the example-based synthesis network paradigm described in the Section 2 is that it requires warping and blending of *all* the example images within the network. Such an approach does not take advantage of the inherent *locality* of the image space. For example, suppose we wanted to model vertical and horizontal head pose movements using a two-dimensional, 3-by-3 network such as the one shown in Figure 6.

Such a 3-by-3 network, however, may be viewed as *four 2-by-2* networks that share a common set of example images along the adjacent edges. Instead of traversing one large network space, smaller, *local* network spaces are traversed, and a navigational mechanism is utilized to determine which local network is currently activated. Experiments were performed with exactly such a set of 4 composed local networks denoting horizontal and vertical pose movement, and some of the synthesized results are shown in Figure 7. The navigational mechanism used in this case performs a horizontal and vertical threshold check based on the input parameters to check which network is activated.

There are several major advantages of using such a network composition technique. Firstly, *composition is natural*, at least within a synthesis framework based on morphing. If the 2-dimensional space is large, chances are that the intermediate examples should only be determined from the example images that are the closest.

Secondly, *composition maintains constant computation complexity.* No matter how large an example set space becomes, if one synthesizes only from the four closest examples, then the computational complexity remains constant. The only price to be paid is the price of having to decide which network to activate, which is not as computationally intensive as having to warp and blend from a large number of examples.

Thirdly, *composition improves final image quality.* By using only the most relevant images for synthesis, the final image quality is improved. Image quality tends to decrease when a large number of examples are warped and blended together, due to the accumulated errors.

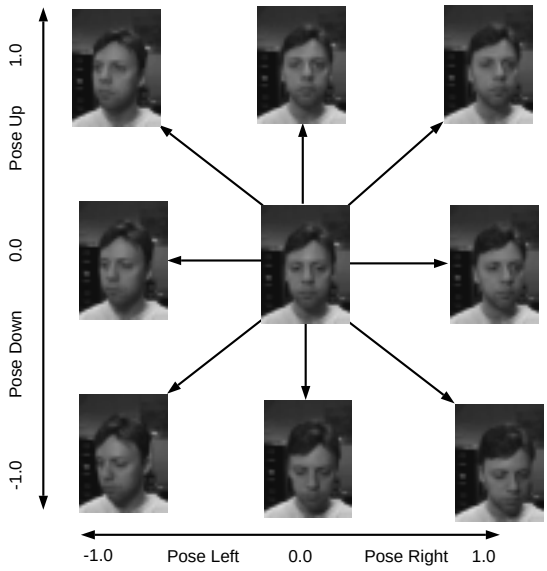

Figure 6: *The examples for a 3-by-3 network involving pose movements in all directions.*

Figure 7: *Some intermediate examples generated from the synthesis network of Figure 6, and their positions in the imposed parameter space.*

3.3 Hierarchical Networks

Another modification to the general example-based synthesis paradigm introduced in the second section emerged in the course of attempting to model eye, mouth, and pose movements simultaneously. It became apparent that there is an *inherent hierarchical relationship* between certain facial motions. For example, eye motions and mouth motions are subordinate to pose motions: changing pose necessarily affects the appearance of the eyes and the mouth, while movements of the mouth and the eyes do not change the overall pose of the head. Consequently, there was a need for a modified synthesis approach which attempted to en-

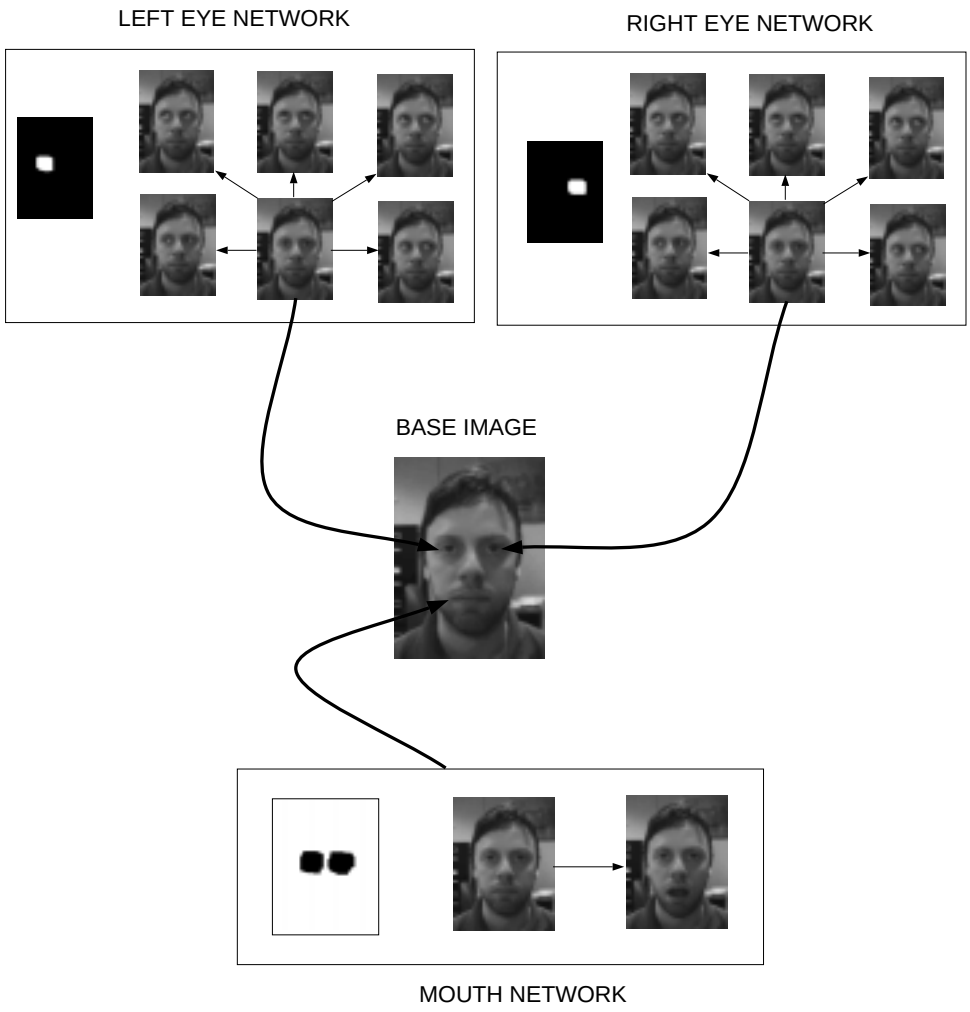

Figure 4: *Construction of a 7-example, 5-dimensional regional synthesis network controlling mouth movement and eye movement.*

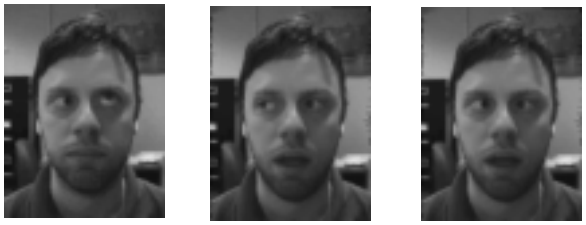

Figure 5: *Synthesized images generated from the network in the Figure 4.*

code this new notion of hierarchy between networks.

The modified hierarchical synthesis paradigm was applied to a 14-example, 4-dimensional network that involved eye, mouth, and pose movements, shown in Figure 8. Firstly, a 7-example eye-mouth network was constructed for eye and mouth movements *at a single head pose*. The eye-mouth network was composed of two regional networks for the eyes and mouth, as described in the previous section on regional networks. A similar 7-example network was also created for the eye and mouth movements *at a second rightwards pose*. The two 7-example eye-mouth networks, shown in Figure 8 a), thus constitute *subnetworks* to be placed within a larger pose network.

The next step in the creation of the eyes-mouth-pose network, shown in Figure 8 b), is to compute a set of *cross-flows* linking the images between the two subnetworks. The cross-flows may be obtained using any one of various methods described in [9], and essentially allow the two eye-mouth subnetworks to be placed in correspondence, in the same manner as two images would be placed in correspondence.

The third and most important step in the hierarchical synthesis paradigm, shown in Figure 8 c), is to *synthesize an intermediate eye-mouth subnetwork for a change in pose*. Synthesizing such a new intermediate eye-mouth subnetwork consists of two steps:

- The first involves synthesizing the new, intermediate *images* that belong in the new, intermediate subnetwork. The synthesis of the new images proceeds along the respective cross-flow vectors. Essentially, temporary 1-dimensional synthesis networks are created, where the corner images are the corresponding images in the mouth subnetworks, and the correspondence vector is the cross-flow vector. Synthesis of the intermediate images proceeds in the standard manner described in Section 2.
- The second step, involves the synthesis of new, intermediate *correspondences* tying the images within the new, intermediate subnetwork together. In this case, a temporary network is created *in which the endpoints are not images, but the two correspondences from the corner mouth subnetworks*. These correspondences are warped to produce the intermediate correspondences that tie the images within the intermediate subnetwork together.

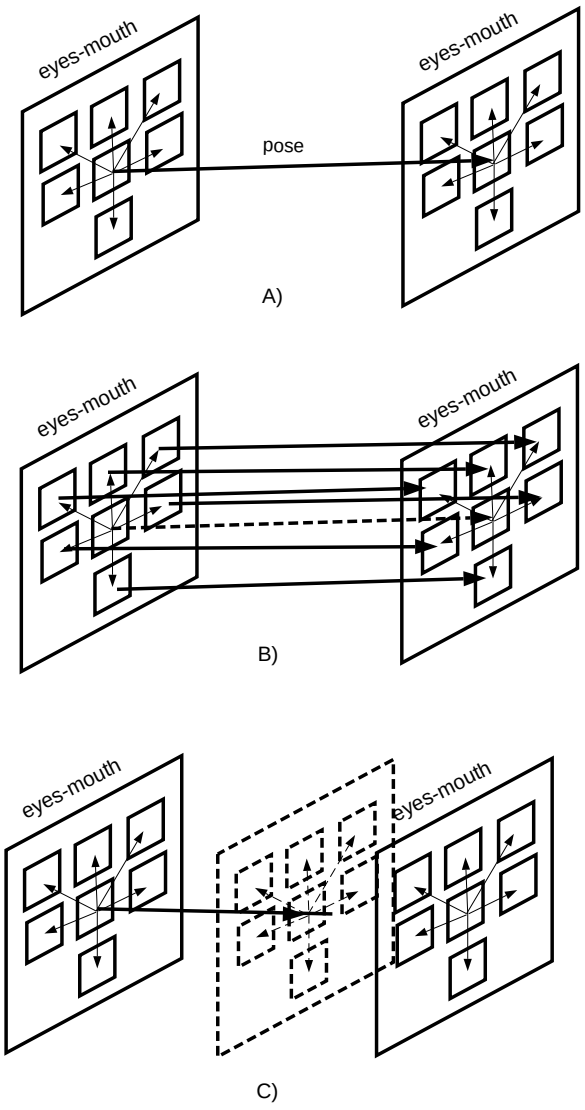

Figure 8: *The stages of the new hierarchical synthesis approach for the 14-example, 4-dimensional, eyes-mouth-pose network.*

Experiments were performed with such a hierarchical eyes-mouth-pose network, and Figure 9 shows two sequences of images generated from the same eyes-pose-mouth synthesis network. In the top row, the mouth is kept closed while the eyes and the pose are changed. In the bottom row, all three facial features are changed.

Figure 9: *Two separate image sequences synthesized from the 14-example, 4-dimensional eyes-mouth-pose network in Figure 8.*

It is interesting to point out that the modified hierarchical synthesis approach is not a new paradigm at all, but a generalization. In the old synthesis method, *images* were warped and blended together to achieve novel, intermediate images. In the new method, this notion of warping and blending is extended to include not only *images*, but also *flows*, and hence *entire networks*. One can alternatively think of the synthesis technique as warping and blending *nodes*, where a node can be an image, a network, a network of networks, and so on.

4 Analysis

4.1 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 networks created in the previous section are themselves used for analysis. An important and useful consequence of this approach is that the only parameters that may be extracted from the novel sequence are those that are encoded by the synthesis networks themselves.

4.2 Affine Parameters

Before analyzing with respect to novel image sequences, the synthesis networks must 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. Essentially, the network first synthesizes the head at the intrinsic parameters imposed by the user, and then it performs an affine transformation according to the desired translation, scale, and rotation. Ideally, we would also like to augment the synthesis network with a set of projective parameters, but this was beyond the scope of this work.

4.3 Segmentation

In addition to augmenting the synthesis network with a set of affine parameters, it is also necessary to incorporate segmentation. This is needed because, in its effort to match the synthesis network with a novel sequence, 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.

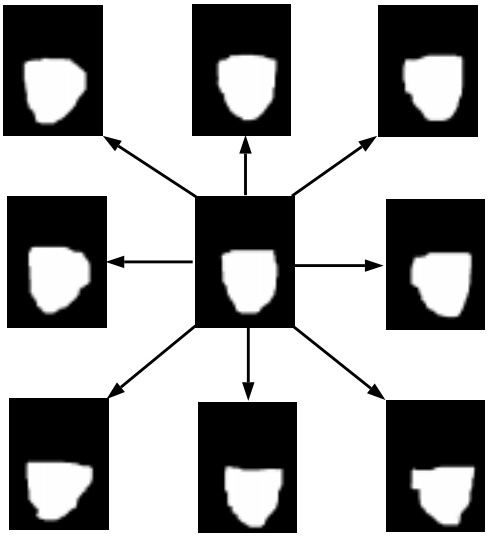

Figure 10: *The masks associated with the 3-by-3 pose network in Figure 6.*

In attempting to segment the head in a *network*, as opposed to segmenting the head in just an *image*, there is a need for a *flexible* segmentation scheme, because the outline of the head changes shape as the head changes pose, position, rotation, and scale. One rigid mask is thus not capable of segmenting the head properly.

Consequently, 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. The masks are defined by hand, although it is possible to use other automatic techniques. Whenever the synthesis network 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, with minor modifications to preserve the black-and-white pixel integrity of the mask.

Figure 10 depicts the masks that would be associated with the 3-by-3 pose network in Figure 6. Figure 11 shows various affine-perturbed, segmented images which are synthesized from a network similar to the 3-by-3 pose network of Figure 6.

Figure 11: *Various segmented and affine-perturbed images synthesized from a 3-by-3 pose network similar to the one shown in Figure 6.*

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. It is important to note, in light of the forthcoming description of our analysis algorithm, that in addition to images, the synthesis module can also output correspondences and masks.

4.4 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 [11], 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. Both Beymer, Shashua, and Poggio [4] and Essa and Pentland [8] also found that comparing novel incoming motion with stored motion (or motion energy) templates led to good facial analysis results.

It is important to note that the set of correspondences that can be synthesized by the network are *only* those correspondences involved with the facial motions that the user chose to model, in addition to the correspondences involved with the affine movements. Consequently, even though the novel sequences will generate arbitrary types of correspondences in general, we are constrained, through this synthesis module paradigm, of matching them with only a certain repertoire of acceptable correspondences. Our matching approach may be viewed as a form of *motion regularization*, in which unconstrained optical flow is regularized with the flows from our synthesis module. Other motion regularization approaches were made by Basu, Essa, and Pentland [1] and Black and Yacoob [5]. Basu, Essa, and Pentland regularized the unconstrained optical flow with the motion of a three-dimensional ellipsoid for head-tracking purposes. Black and Yacoob regularized *local* regions of optical flow using projective planar models and curve models.

4.5 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 [9], 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 translation, rotation, scale, pose, and other variables did exist nevertheless.

5 Results

A varied but limited 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. The novel sequences involved changes in lighting, position, scale, rotation, background, clothing, and hairstyle. It should be noted that to change his hairstyle, the author shaved his head! In each illustration, a few frames from the novel sequence are juxtaposed against a few frames from the synthesized sequence. The analysis parameters extracted by the algorithm are also shown.

The synthesis modules embedded within the analysis algorithm were based on the networks described in Sections 2 and 3 of this paper. Specifically, the two-dimensional, 3-by-3 pose network shown in Figure 6 was used to analyze various novel pose movements of the head. Figures 12 through 18 on the following pages depict the results from the experiments. In addition, the 4-dimensional, 14-example eye-pose-mouth network shown in Figure 8 was used to analyze various combinations of mouth, eye, and pose movements. The results from these experiments are shown in Figures 20 through 24. Finally, the 2-dimensional, 5-example expression network shown in Figure 1 was used to analyze both mouth expression movements and a variety of affine head movements. The results from two such experiments are shown in Figures 26 and 28.

6 Discussion and Future Work

Our analysis experiments are still very preliminary, and more thorough testing is needed on a larger database of facial expressions and head movements. On the other hand, the results are extremely encouraging.

At present, the most salient difficulty with the analysis-by-synthesis algorithm presented in this work is that, like many iterative nonlinear optimization techniques, it is computationally inefficient. Formal timing tests were not performed, but it takes between a few minutes to half an hour to analyze one frame, depending on the complexity of the underlying synthesis module. Future work definitely needs to explore improving the efficiency of the algorithm.

It also seems that the analysis-by-synthesis paradigm as presented is also strongly *user-dependent*, although formal tests were not performed to confirm this: the example-based models can only extract analysis parameters *reliably* from faces whose examples were used to build the model. Further work is needed to determine the limitations of the example-based models in this respect, and to overcome those limitations.

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.

Acknowledgments

The authors would like to thank and acknowledge David Beymer, who provided code which formed the basis for this work, as well as David Sarnoff Research Labs for their optical flow code and image libraries. Also, the authors would like to thank Mike Jones, Steve Lines, Federico Girosi, and Theodoros Evgeniou for many thoughtful discussions and criticisms.

References

[1] Sumit Basu, Irfan Essa, and Alex Pentland, "Motion Regularization for Model-based Head Tracking", MIT Media Laboratory Perceptual Computing Section Technical Report No. 362, 1996.

[2] J.R. Bergen and R. Hingorani, "Hierarchical Motion-Based Frame Rate Conversion", David Sarnoff Research Center Technical Memo, April, 1990.

[3] David Beymer, "Vectorizing Face Images by Interleaving Shape and Texture Computations", MIT AI Lab memo, No. 1537, September, 1995.

[4] D. Beymer, A. Shashua, and T. Poggio, "Example Based Image Analysis and Synthesis", MIT AI Lab Memo, No. 1431, 1993.

[5] Michael Black and Yaser Yacoob, "Tracking and Recognizing Rigid and Non-rigid Facial Motions using Local Parametric Models of Image Motion", International Conference on Computer Vision, Cambridge, Massachusetts, pp. 374–381, June, 1995.

[6] Leonard McMillan and Gary Bishop, "Plenoptic Modeling: An Image-Based Rendering System", SIGGRAPH '95 Proceedings, Los Angeles, CA, 1995.

[7] S.E. Chen and L. Williams, "View Interpolation for Image Synthesis", SIGGRAPH '93 Proceedings, Anaheim, California, pp. 279–288, August, 1993.

[8] Irfan A. Essa and Alex Pentland, "Facial Expression Recognition Using a Dynamic Model and Motion Energy", MIT Media Lab Perceptual Computing Section Technical Report No. 307, 1995.

[9] T. Ezzat, "Example-Based Analysis and Synthesis for Images of Human Faces", Master's Thesis, School of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, February 1996.

[10] F. Girosi, M. Jones, and T. Poggio, "Priors, Stabilizers, and Basis Functions: From Regularization to Radial, Tensor, and Additive Splines", MIT AI Lab Memo, No. 1430, June, 1993.

[11] M. Jones and T. Poggio, "Model-Based Matching of Line Drawings by Linear Combinations of Prototypes", International Conference on Computer Vision, Cambridge, Massachusetts, pp. 531–536, June, 1995.

[12] M. Jones and T. Poggio, "Model-Based Matching by Linear Combinations of Prototypes", unpublished MIT AI memo.

[13] T. Poggio and R. Brunelli, "A Novel Approach to Graphics", MIT AI Memo No. 1354, 1992.

[14] T. Poggio and T. Vetter, "Recognition and Structure from One 2D Model View: Observations on Prototypes, Object Classes, and Symmetries", MIT AI Memo, No. 1347, 1992.

[15] Steve M. Seitz and Charles R. Dyer, "Physically-Valid View Synthesis by Image Interpolation", Proc. IEEE Workshop on the Representation of Visual Scenes, pp. 18–25, June, 1995.

[16] T. Werner, R. D. Hersch, and V. Hlaváč, "Rendering Real-World Objects Using View Interpolation", International Conference on Computer Vision, Cambridge, Massachusetts, pp. 957–962, June, 1995.

Figure 12: *A novel sequence with leftwards pose movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the 9-example 3-by-3 pose network shown in Figure 6 in this paper.*

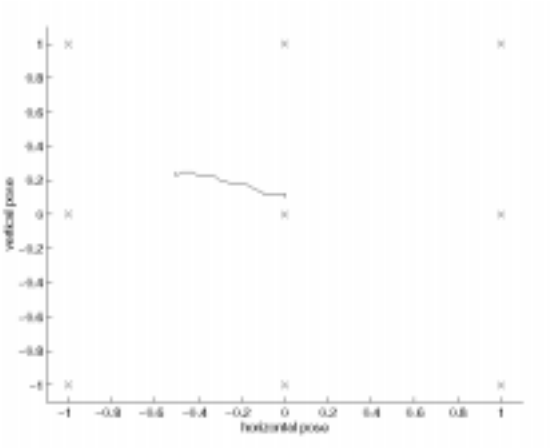

Figure 13: *The pose parameters extracted from the sequence in Figure 12. The analysis algorithm extracts a set of four affine parameters as well as 2 pose parameters, but only pose parameters are shown here for illustration. The "x" marks denote the positions of the 9 examples in pose space.*

Figure 14: *A novel sequence with rightwards pose movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the 9-example 3-by-3 pose network shown in Figure 6 in this paper.*

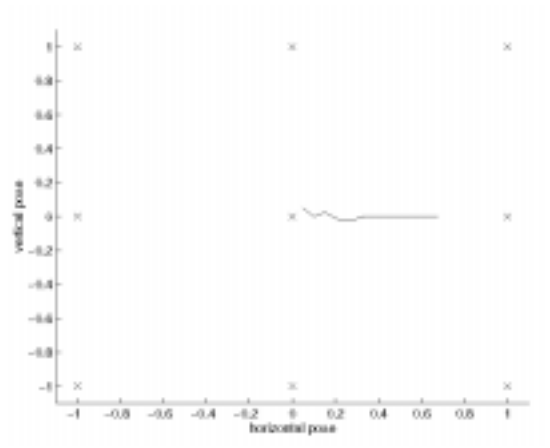

Figure 15: *The pose parameters extracted from the sequence in Figure 14. The analysis algorithm extracts a set of four affine parameters as well as 2 pose parameters, but only pose parameters are shown here for illustration. The "x" marks denote the positions of the 9 examples in pose space.*

Figure 16: *A novel sequence with top-rightwards pose movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the 9-example 3-by-3 pose network shown in Figure 6 in this paper.*

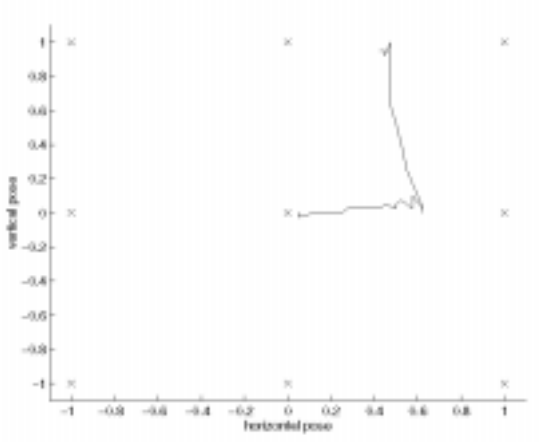

Figure 17: *The pose parameters extracted from the sequence in Figure 16. The analysis algorithm extracts a set of four affine parameters as well as 2 pose parameters, but only pose parameters are shown here for illustration. The "x" marks denote the positions of the 9 examples in pose space.*

Figure 18: *A novel sequence with bottom-leftwards pose movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the 9-example 3-by-3 pose network shown in Figure 6 in this paper.*

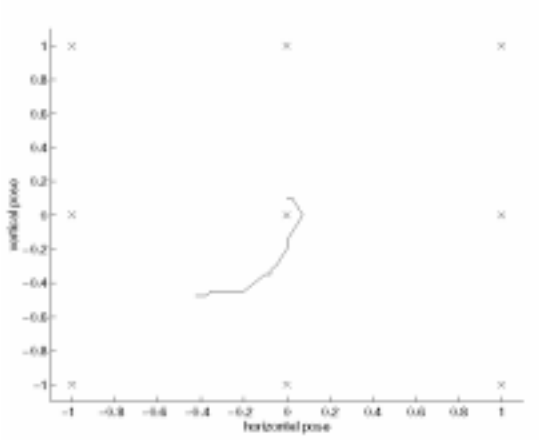

Figure 19: *The pose parameters extracted from the sequence in Figure 18. The analysis algorithm extracts a set of four affine parameters as well as 2 pose parameters, but only pose parameters are shown here for illustration. The "x" marks denote the positions of the 9 examples in pose space.*

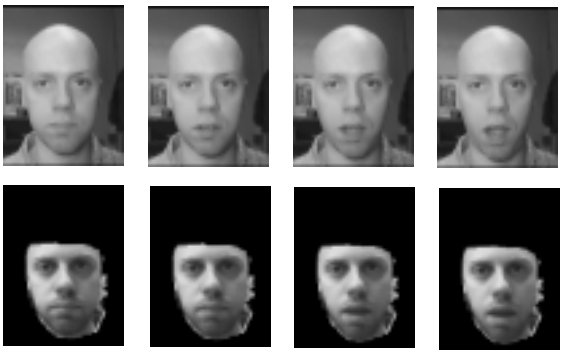

Figure 20: *A novel sequence with mouth movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the 4-dimensional, 14-example network shown in Figure 8.*

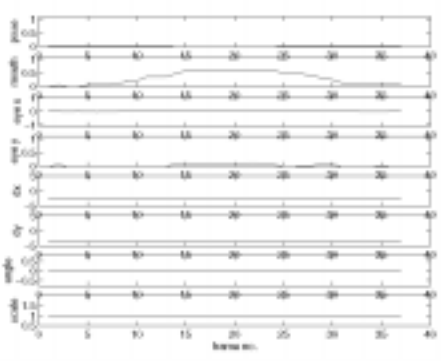

Figure 21: *The complete set of parameters extracted from the sequence in Figure 20. All the activity occurs in the mouth parameter, which denotes degree of openness.*

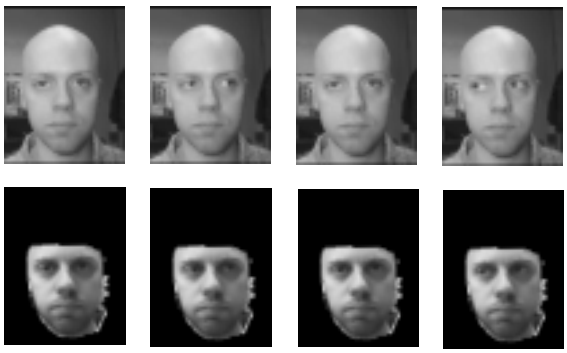

Figure 22: *A novel sequence with eye movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the same 4-dimensional, 14-example network shown in Figure 8.*

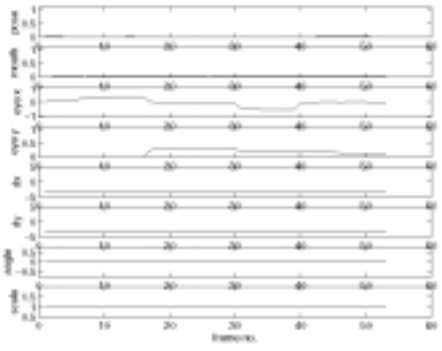

Figure 23: *The complete set of parameters extracted from the sequence in Figure 22. All the activity occurs in the eyes x and y parameters.*

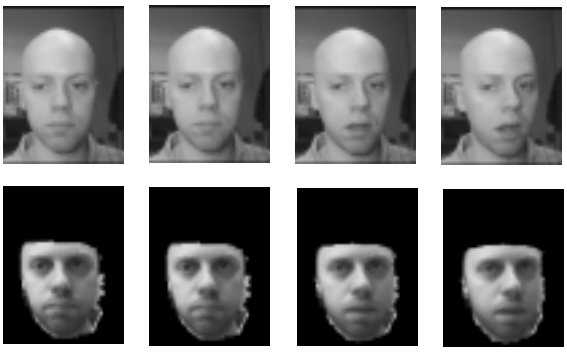

Figure 24: *A novel sequence with eye, pose, and mouth movement (top), juxtaposed along with the synthesized sequence (bottom). The synthesis module is the same 4-dimensional, 14-example network shown in Figure 8.*

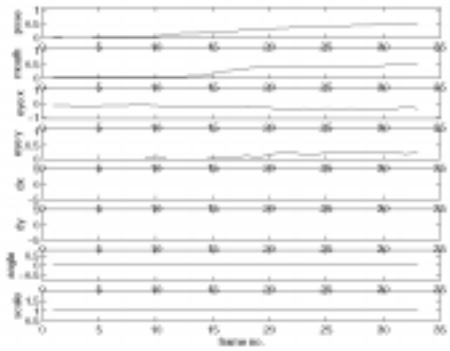

Figure 25: *The complete set of parameters extracted from the sequence in Figure 24. All the activity occurs in the eye, mouth, and pose parameters.*

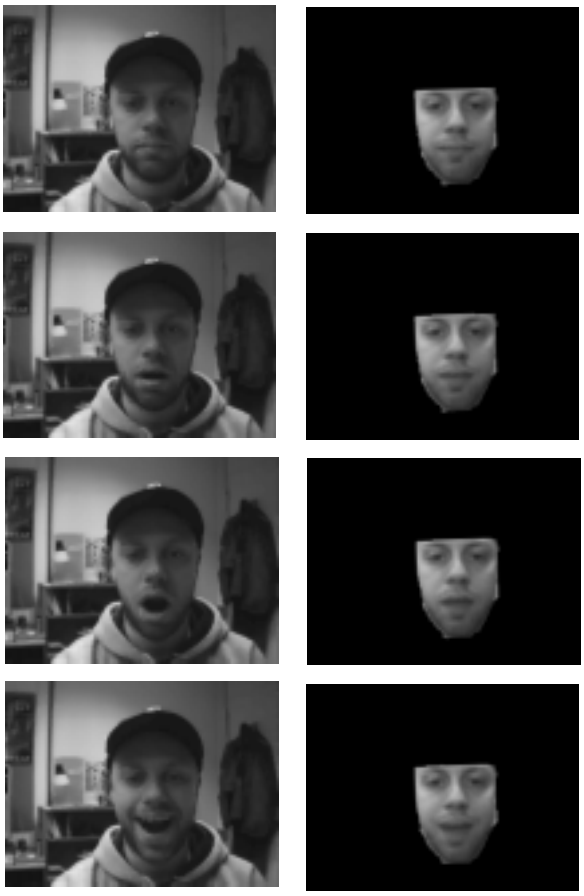

Figure 26: *A novel sequence with mouth movement (left), juxtaposed along with the synthesized sequence (right). The synthesis module is the 2-dimensional, 5-example network shown in Figure 1.*

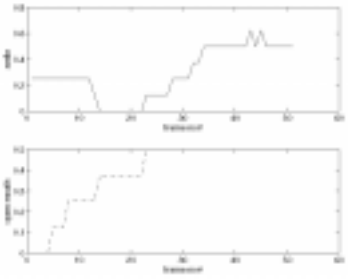

Figure 27: *The smile and open-mouth parameters extracted from the sequence in Figure 26. The analysis algorithm also extracts a set of four affine parameters as well, but only the expression parameters are shown here for illustration.*

Figure 28: *A novel sequence with on-plane rotation of the head (left), juxtaposed along with the synthesized sequence (right). The synthesis module is the same 2-dimensional, 5-example expression network shown in Figure 1.*

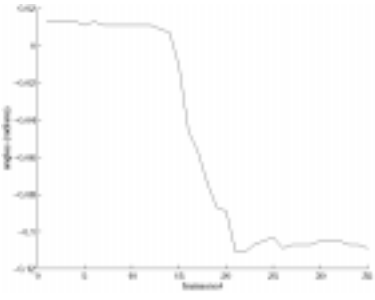

Figure 29: *The on-plane rotation parameter extracted from the sequence in Figure 28. Another set of three affine parameters, as well as two expression parameters are also extracted, but only the rotation parameters are shown here for illustration.*