HOLISTIC SYNTHESIS OF HUMAN FACE IMAGES

João Maciel and João Costeira

Instituto de Sistemas e Robótica - Instituto Superior Técnico
Av. Rovisco Pais, 1096 Lisboa Codex, PORTUGAL
e-mail: {maciel,jpc}@isr.ist.utl.pt

ABSTRACT

This paper presents a method to automatically synthesize human face images from holistic descriptions. We compactly represent the face set by a small set of prototypes, which can be used in simple ways to generate controlled morphings. This becomes possible because separation of 2D-shape and texture provides a faithful, closed and convex representation of images, and smoothes the mappings between images and their properties. With this approach, the user watches an image being continuously morphed according to his indications, and the synthesized images always obey the natural physiognomic constraints.

1. INTRODUCTION

Police investigation often requires a face sketch to be drawn from a description. Traditional patchwork systems that combine typical face parts (composite pictures) require very qualified human operators and usually produce unrealistic results. With our approach, a non-specialized user describes a face using global characteristics (like typical expression or physique) and watches an image being continuously morphed accordingly. Even when indications are inherently local (like feature size) the results will obey the natural physiognomic constraints.

1.1. Approach and Contribution

In order to ease the generation of smooth morphings, an image representation should meet some requirements. We show why separated 2D-shape and texture representations meet these requirements and we also describe a method of computing this representation automatically.

Shape and texture representation allows us to identify well defined regions corresponding to certain face characteristics. We introduce the concept of *extreme prototype* to represent these regions and to indicate directions that lead to the desired monotonous changes in face characteristics.

1.2. Previous Work

This synthesis paradigm was first used in binary images by Ullman and Basri [8]. Poggio and Brunelli [6], extended the approach to gray-level images, using shape and texture representation. Since then, shape and texture separation has been widely used in recognition [2] and image manipulation [9, 7]. However, as far as we know, this is the first application to the synthesis of images of *faces with completely new identity*. Recently, Cootes *et al* (see [5]) use shape and texture separation in a representation model with a promising range of high quality applications.

2. A SYNTHESIS PARADIGM

A discrete image $\mathbf{I}_{[n \times n]}$ can be interpreted as a vector $\mathbf{i}$ in the n^2-dimensional space $\mathcal{S}_{image}$, whose coordinate $\mathbf{i}_{x+n\times(y-1)}$ corresponds to pixel (x, y). The subset of $\mathcal{S}_{image}$ containing all faces ($\mathcal{C}_{faces}$) includes images varying in pose, rotation, scaling, illumination, facial expression and identity. In [3], Bichsel and Pentland show that, if we exclude binary-type modifications (like adding of glasses), these images form a *visually homogeneous image class*, where any pair of points can be joined by a continuous line, itself contained in $\mathcal{C}_{faces}$. This means that $\mathcal{C}_{faces}$ is connected.

2.1. Direct synthesis paradigm and its distinction

The connectivity of $\mathcal{C}_{faces}$ allows us to synthesize smooth image sequences in a given class by generating continuous trajectories inside the correspondent image subset. Trajectories should be generated according to high level descriptions, so this *synthesis paradigm* involves inverting the image-to-classification mappings, which can be learned from examples. This contrasts with the traditional approaches that resort to physical models and rendering.

Example-based face image synthesis is a usual solution to *morphing* (interpolation between two given examples) and *feature modification* (pose, illumination, age and expression changes). Our paradigm extends to a broader concept of parameter-controlled *extrapolation* of an example set, allowing us to change face identity.

2.2. Direct synthesis: main features

Continuous parameter-to-image mappings must be learned from examples. Due to the *curse of dimensionality*, the huge dimensionality of $\mathcal{C}_{faces}$ imposes that too many examples must be gathered, urging to the use of a *compact data representation*. Furthermore, $\mathcal{C}_{faces}$ is *highly non-convex* because of the nonlinearities between edge positions and pixel intensity — see [3]. Generalization from examples is an *ill-posed problem*, so our representation has to provide smooth mappings so that regularization constraints make sense. In short, a good representation must comply with the *compromise among its fidelity, smoothness and closedness*. In section 5 we show how to experimentally demonstrate the properties of our image representation.

Human intervention brings other difficulties, since witnesses provide very *incomplete and subjective descriptions*. This is why photo-fit operators are usually psychologists with artistic skills.

3. SEPARATED SHAPE AND TEXTURE REPRESENTATION

Most of the stated difficulties can be overtaken by picking the appropriate representation. Compaction requires data redundancy, that is, requires some correlation among all images in our set. An effective way of increasing image correlation is feature alignment by geometric deformation. Unlike low-pass filtering (usual choice for redundancy enhancement), geometric deformation can be easily memorized and inverted. So, if *deformation takes part in the representation*, the original images can be recovered.

This suggests a representation made of two parts: 2D-shape (a geometric deformation that aligns corresponding pixels) and texture (geometrically rectified image). Shape and texture of an image class (now defined by a set of features admitting one-to-one correspondences) make up highly redundant separated sets [9], and admit PCA-based compaction.

3.1. Representation operators

We want to represent images $\mathbf{I}_k$, $(k = 1, \ldots, m)$ of $n \times n$ pixels, belonging to the same image class. Each image $\mathbf{I}_k$ is related to a class reference image $\mathbf{R}$ through a shape vector field $\mathbf{S}_k = (\Delta x, \Delta y)_{[n \times n]}$ (coordinates of every pixel of $\mathbf{I}_k$ in respect to $\mathbf{R}$) and a texture matrix $\mathbf{T}_k$ (intensity differences between corresponding pixels in $\mathbf{I}_k$ and $\mathbf{R}$). Texture can be found by subtracting $\mathbf{R}$ from shape-normalized $\mathbf{I}_k$:

$$\mathbf{T}_k(x, y) = \mathbf{I}_k\left[x - \Delta x(x, y), y - \Delta y(x, y)\right] - \mathbf{R}(x, y) \qquad (1)$$

We define two operators $\mathbf{S}_k = shape\left(\mathbf{I}_k, \mathbf{R}\right)$ and $\mathbf{T}_k = texture\left(\mathbf{I}_k, \mathbf{R}\right)$ to perform shape and texture extraction. Reconstruction is performed by the inverse of equation (1):

$$\begin{aligned} rec\left(\mathbf{R}, \mathbf{T}_k, \mathbf{S}_k\right)_{(x,y)} \;=\; & \mathbf{T}_k\left[x + \Delta x_{(x,y)}, y + \Delta y_{(x,y)}\right] + \\ & + \mathbf{R}\left[x + \Delta x_{(x,y)}, y + \Delta y_{(x,y)}\right] \end{aligned} \qquad (2)$$

Figure 1 illustrates this proces.

3.2. Shape estimation

We now investigate two ways of implementing the $shape()$ operator. The first method uses f manually identified features in a small set of l images. Their coordinates $\mathbf{c}_k$ (of size $[f \times 2]$) are used in the following way:

1. Find the mean value $\bar{\mathbf{c}}_{[f \times 2]}$ of all features.

2. Compute shape for every image $(k = 1, \ldots, l)$:

  i) Find the displacement of each feature, with respect to the reference: $\mathbf{d}_k = \mathbf{c}_k - \bar{\mathbf{c}}$.

  ii) Build $\mathbf{S}_k\,(x, y)$ by interpolating for every pixel.

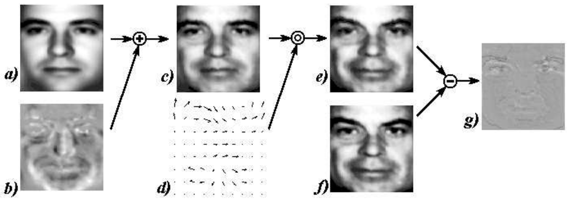

Figure 1: Reconstruction from shape and texture: a) Reference; b) Texture; c) Texture + reference; d) Shape; e) Reconstruction; f) True image; g) Reconstruction error.

3. Build the reference image by point-wise averaging all shape-normalized images:

$$\mathbf{R} = \frac{1}{l}\sum_{k=1}^{l}\left[rec^{-1}\left(\mathbf{I}_k, \mathbf{0}, \mathbf{S}_k\right)\right] \qquad (3)$$

$$rec^{-1}\left(\mathbf{I}_k, \mathbf{T}_k, \mathbf{S}_k\right) = rec\left[\mathbf{I}_k, \mathbf{0}, -rec\left(\mathbf{S}_k, \mathbf{0}, \mathbf{S}_k\right)\right] - \mathbf{T}_k \qquad (4)$$

4. Find the textures by subtracting the reference from each shape-normalized image: $\mathbf{T}_k = rec^{-1}\left(\mathbf{I}_k, \mathbf{0}, \mathbf{S}_k\right) - \mathbf{R}$

At this point we can feed this reference image $\mathbf{R}$ into an automatic procedure for the whole set of m images. This procedure is a best-fit block search algorithm. The displacement $\mathbf{S}_k\,(x, y)$ maximizes the normalized cross-correlation (see [4]) between $d \times d$ blocks in the image $\mathbf{I}_k$ and in the reference $\mathbf{R}$.

Since shape and texture representation is not unique (differences in shape deformations can be canceled by texture modifications), we used a multi-resolution procedure to impose smoothing constraints and to speed up the search.

In order to avoid the errors coming from differences on correspondent pixel intensities, we performed an iterative improvement of shape and texture. Start by considering zero textures ($\mathbf{T}_k = \mathbf{0}$), and proceed with:

1. Compute the shape $\mathbf{S}_k = shape\left(\mathbf{I}_k, \mathbf{R} + \mathbf{T}_k\right)$ for all images $(k = 1, \ldots, m)$, using *multi-resolution best-fit block search*.

2. Compute SVD of shape and texture collections. If $\mathbf{s}_k$ and $\mathbf{t}_k$ are the column vector versions of $\mathbf{S}_k$ and $\mathbf{T}_k$, factorize: $\left[\mathbf{s}_1|\cdots|\mathbf{s}_m\right] = \mathbf{u}_s\Sigma\mathbf{v}_s^T$ and $\left[\mathbf{t}_1|\cdots|\mathbf{t}_m\right] = \mathbf{u}_t\Sigma\mathbf{v}_t^T$.

3. Project the shape over a reduced base $\mathbf{u}_s^{(l)}$(first l components): $\mathbf{s}_k^{(l)} = \mathbf{u}_s^{(l)}\left(\mathbf{s}_k^T\mathbf{u}_s^{(l)}\right)^T$.

4. Compute the texture associated to each $\mathbf{s}_k^{(l)}$; if $\mathbf{S}_k^{(l)}$ is the matrix form of $\mathbf{s}_k^{(l)}$, then: $\mathbf{T}_k = rec^{-1}\left(\mathbf{I}, \mathbf{0}, \mathbf{S}_k^{(l)}\right) - \mathbf{R}$.

5. Project the computed textures over the reduced base $\mathbf{u}_t^{(l)}$(first l components): $\mathbf{t}_k^{(l)} = \mathbf{u}_t^{(l)}\left(\mathbf{t}_k^T\mathbf{u}_t^{(l)}\right)^T$, and reorganize it in matrix form $\mathbf{T}_k^{(l)}$.

6. Update $\mathbf{S}_k = \mathbf{S}_k^{(l)}$ and $\mathbf{T}_k = \mathbf{T}_k^{(l)}$, and go back to step **1.**

Beymer [1] shows that a procedure of this sort converges to solutions insensible to data noise.

4. SYNTHESIS

Synthesizing a face image is equivalent to generating trajectories in the face image space $\mathcal{C}_{faces}$ (in its shape and texture representation) by acting on high-level parameters. Imagine, for example, that you want to grow the face's nose, or give it an angry look. This is achieved by introducing the concept of *extreme prototype*, which will allow us to easily perform the desired smooth trajectories.

4.1. Prototypes

A prototype is defined as a point in $\mathcal{C}_{faces}$ that represents a certain combination of classification parameter values. *Extreme prototypes* have a few parameters with values close to the sampled extremes and all other with average values. An extreme prototype is associated with a set of *boundary-valued* parameters (*imposed* properties) and a set of average-valued parameters.

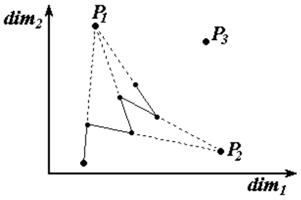

Figure 2: Approximation of $\mathbf{p}_3$ by successive displacements towards $\mathbf{p}_1$ and $\mathbf{p}_2$.

Section 5 will confirm that using separated shape and texture representation provides a convex face-set and monotonous property mappings. So, averaging shape fields and texture matrices of n pre-classified samples with extreme values on parameter l will result on an extreme prototype having:

$$\begin{cases} \mathbf{S}_P(l) = \dfrac{\sum_{k=1}^{n} \mathbf{S}_{k(l)}}{n} \\[2ex] \mathbf{T}_P(l) = \dfrac{\sum_{k=1}^{n} \mathbf{T}_{k(l)}}{n} \end{cases} \tag{5}$$

Using large n assures non-biased averaged properties.

4.2. Using prototypes to perform image synthesis

Starting from a random face image, synthesis is performed using the prototypes in one of the following ways:

1. Local evolution from the present suggested shape and texture $\mathbf{x}$ towards the extreme prototype $\mathbf{p}_l$ holding the desired properties: $\mathbf{x}' = \mathbf{x} + \alpha \cdot (\mathbf{p}_l - \mathbf{x})$. The parameter $\alpha << 1$ is the fraction of distance between $\mathbf{x}$ and $\mathbf{p}_l$ displaced each time. Figure 2 illustrates the application of this method to approximate $\mathbf{p}_3$. Note that it is sufficient to memorize a few *elementary prototypes* (those with only one imposed property), but the synthesized trajectory will never reach the desired point $\mathbf{p}_3$. This method cannot *extrapolate the convex hull* of the elementary prototypes!
2. Extrapolation can be accomplished using: $\mathbf{x}' = \mathbf{x} + \alpha \cdot \mathbf{p}_l$. This method was named as **Parallel Deformation** by Beymer and Poggio [2]. The procedure can be used to apply the transformation between two examples $\mathbf{I}_1 = rec\left(\mathbf{R}, \mathbf{T}_1, \mathbf{S}_1\right)$ and $\mathbf{I}_2 = rec\left(\mathbf{R}, \mathbf{T}_2, \mathbf{S}_2\right)$ to modify a third image $\mathbf{I}_3 = rec\left(\mathbf{R}, \mathbf{T}_3, \mathbf{S}_3\right)$:
   - **a)** Measure the changes $\Delta\mathbf{S}$ in shape and $\Delta\mathbf{T}$ in texture from $\mathbf{I}_1$ to $\mathbf{I}_2$, by:

     $$\begin{cases} \Delta\mathbf{S} = shape\left(\mathbf{I}_2, \mathbf{I}_1\right) \\ \Delta\mathbf{T} = texture\left(\mathbf{I}_2, \mathbf{I}_1\right) \end{cases} \tag{6}$$

   - **b)** Modify $\mathbf{I}_2$ using these changes with the appropriate geometric normalization: $\mathbf{I}_4 = rec\left(\mathbf{R}', \mathbf{0}, \mathbf{S}_4\right)$, with:

     $$\begin{cases} \mathbf{R}' = rec\left[\mathbf{R}, \mathbf{T}_3 + rec^{-1}\left(\Delta\mathbf{T}, \mathbf{0}, \mathbf{S}_1\right), \mathbf{S}_3\right] \\[2ex] \mathbf{S}_4 = rec\left[rec^{-1}\left(\Delta\mathbf{S}, \mathbf{0}, \mathbf{S}_1\right), \mathbf{0}, \mathbf{S}_3\right] \end{cases} \tag{7}$$

3. **Mixed method**, following the average of the directions found in **1.** and **2.**: $\mathbf{x}' = \left(1 - \alpha/2\right) \cdot \mathbf{x} + \alpha \cdot \mathbf{p}_k$.

One of these *local evolution* procedures is incorporated in step ii) of the following iteration:

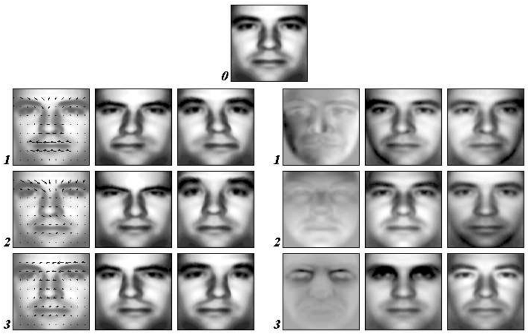

Figure 3: Leading 3 principal components of shape and texture sets, and their effect over the average face 0.

Figure 4: Reconstruction of an example (left) from memorized shape and texture (center) and using 40 PCA components (right).

i) ask the witness for some assured properties and compute non-elementary prototypes that will be used as first approximations;

ii) ask for changes and travel in face set using one of the previous local-evolution methods;

iii) if local evolution is unsuccessful, return to step ii) choosing a new initial prototype.

Avoiding subjective descriptions requires a careful picking of accepted properties. We followed the recommendations in [4] and adopted a three-level domain for each parameter. At the end, the synthesized image can be submitted to a post-processing stage to add some extra features (like hair or glasses) and change facial expression.

5. RESULTS

In this section we start by describing the details of our system. We then show how to experimentally verify the properties of our image representation. Finally, some examples of image synthesis are presented.

We gathered a set of 1500 digitized images (128 × 128 pixels and 32 gray-levels) of Caucasian male faces. The shape and texture separation procedure described in section 3.2 was applied to the image set. An initial reference was built using the feature-based method over 50 images with 63 manually marked features. The number of retained PCA components (in the iterative shape estimation) was chosen taking in account the visual quality of reconstructions. Letting less than 10% of the pixels have errors greater than 3 levels (a visually good criterion) lead to 40 principal components for each space. Figure 3 shows some of the computed principal components. Observe the dominance of pose and illumination; this adverts to insufficient image normalization.

The properties of our representation were experimentally verified. Representation fidelity tests were extensively performed. Figure 4 shows a particular case, and it can be seen that the reconstruction error is small.

To check the closedness of $\mathcal{C}_{faces}$ in our representation, we randomly generated points in this space. Figure 5 contains examples. In every case, valid face images were generated (they all belong to $\mathcal{C}_{faces}$), so *this is a closed representation of* $\mathcal{C}_{faces}$.

Figure 5: Some random examples. Their validity shows that the representation is closed.

Figure 6: Some pairs of elementary prototypes, with imposed properties of: a) Mouth width; b) Nose to mouth distance; c) Eyebrow separation; d) Nose length; e) Eye opening.

The convexity of $\mathcal{C}_{faces}$ is verified if the average of examples is itself a valid face image. Figure 6 shows this in the case of the prototypes. They are built by averaging several images, and they are themselves valid face images.

We can also conclude that the mappings between images and the classification space are monotonous, because averaging any pair of opposing extremes will result on the average face. The average of, say, the images in figure 6-a is the image 0 in figure 3.

In an off-line procedure, all images were jury-classified according to pose, expression, age, physique, illumination and skin texture. Measures of width, length, slope and relative position of main face features were performed semi-automatically, completing the list of describing properties.

Figure 6 shows some examples of obtained elementary prototypes, in pairs of opposing extremes. Figure 7 shows samples of output sequences from our system, using two different strategies to explore the prototypes, starting from the same random face (left). The last column has the desired image. The storage of only some elementary prototypes allows the system to work in real time and to use only a negligible amount of memory.

The use of pure parallel deformation failed because it is impossible to control the coordinate bounds. After a few iterations the generated points are way beyond the sampled border of $\mathcal{C}_{faces}$. The mixed method yields consistently faster and more accurate convergence of the synthesis iterations (section 4.2), because it can extrapolate the convex hull of the elementary prototypes.

We verified that, using any of the successful strategies, it is impossible to violate the anthropomorphic constraints: when the user indicates local changes, *a set of highly correlated properties is changed*. For example, when one increases mouth width, expression changes to smile.

Finally, figure 8 exemplifies the application of parallel deformation to change facial expressions (this could be used as one of the post-processing mechanisms referred in section 4.2).

Figure 7: Examples of controlled image evolution: a) Using prototype directions; b) Using the mixed method.

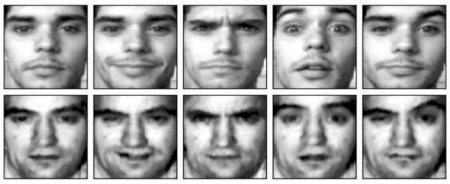

Figure 8: Synthesized facial expressions. The top row contains examples of a reference face. In bottom row, all but the leftmost images are synthesized by application of the exemplified changes.

6. CONCLUSION

Our need to increase data redundancy suggested the use of a separated shape and texture representation. We defined the concept of elementary prototype, as being an image with an extreme value for a single property and average values for the rest. The convexity of shape and texture sets and the monotony of the mappings between these sets and the property space altogether allowed us to build these prototypes by simply averaging shape and texture of many images. This prototype-based approach requires the storage of only a limited set of images.

The closedness, fidelity and convexity properties of the representation were experimentally verified. Experiments also showed that our system does not allow the physiognomic constraints to be violated, so the desired holistic performance was attained.

Future work should be conducted on careful normalization of the original images, especially in illumination, pose and facial expression. A full system should also include some post-processing and high-level interface mechanisms, including the ability to manage a careful interview. We also feel that the concept of extreme prototype can be further exploited in representation and trajectory generation.

7. REFERENCES

[1] D. Beymer. Vectorizing face images by interleaving shape and texture computations. Technical Report AIM 1537, MIT, AI Lab, 1995.

[2] D. Beymer and T. Poggio. Face recognition from one example view. Technical Report AIM 1536, MIT, AI Lab, 1995.

[3] M. Bichsel and A. Pentland. Human face recognition and the face image set's topology. *CVGIP: Image Understanding*, 59(2):254–261, March 1994.

[4] R. Brunelli and T. Poggio. Face recognition through geometrical features. In G. Sandini, editor, *Lecture Notes in Computer Science*, pages 792–800. Springer Verlag, 1992.

[5] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. In *Proc. ECCV'98*, volume II, pages 484–498, 1998.

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

[7] D. Rowland and D. Parrett. Manipulating facial appearance through shape and color. *IEEE Computer Graphics Appl.*, pages 70–76, Sep. 1995.

[8] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Trans. PAMI*, 13(10):992–1005, Oct. 1991.

[9] T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. Technical Report AIM 1531, MIT, AI Lab, 1995.