MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY
and
CENTER FOR BIOLOGICAL AND COMPUTATIONAL LEARNING
DEPARTMENT OF BRAIN AND COGNITIVE SCIENCES

A.I. Memo No. 1531 March, 1995
C.B.C.L. Paper No. 119

# Linear Object Classes and Image Synthesis from a Single Example Image

**Thomas Vetter and Tomaso Poggio**
This publication can be retrieved by anonymous ftp to publications.ai.mit.edu.
Version revised December, 1995.

## Abstract

The need to generate new views of a 3D object from a single real image arises in several fields, including graphics and object recognition. While the traditional approach relies on the use of 3D models, we have recently introduced [11, 6, 5] techniques that are applicable under restricted conditions but simpler. The approach exploits image transformations that are specific to the relevant object class and learnable from example views of other "prototypical" objects of the same class.

In this paper, we introduce such a new technique by extending the notion of linear class first proposed by Poggio and Vetter [12]. For *linear object classes* it is shown that linear transformations can be learned exactly from a basis set of 2D prototypical views. We demonstrate the approach on artificial objects and then show preliminary evidence that the technique can effectively "rotate" high-resolution face images from a single 2D view.



This report describes research done at the Max-Planck-Institut für biologische Kybernetik Tübingen, Germany and the Artificial Intelligence Laboratory and within the Center for Biological and Computational Learning in the Department of Brain and Cognitive Sciences at the Massachusetts Institute of Technology. This research is sponsored by grants from ONR under contract N00014-93-1-0385 and from ARPA-ONR under contract N00014-92-J-1879; and by a grant from the National Science Foundation under contract ASC-9217041 (this award includes funds from ARPA provided under the HPCC program). Additional support is provided by the North Atlantic Treaty Organization, ATR Audio and Visual Perception Research Laboratories, Mitsubishi Electric Corporation, Sumitomo Metal Industries, Kodak, Daimler-Benz and Siemens AG. Support for the A.I. Laboratory's artificial intelligence research is provided by ARPA-ONR contract N00014-91-J-4038. Tomaso Poggio is supported by the Uncas and Helen Whitaker Chair at MIT's Whitaker College. Thomas Vetter holds a Helmholtz fellowship from the BMFT Germany.

# 1 Introduction

View-based approaches to 3D object recognition and graphics may avoid the explicit use of 3D models by exploiting the memory of several views of the object and the ability to interpolate or generalize among them. In many situations however a sufficient number of views may not be available. In an extreme case we may have to do with only one real view. Consider for instance the problem of recognizing a specific human face under a different pose or expression when only one example picture is given. Our visual system is certainly able to perform this task – even if at performance levels that are likely to be lower than expected from our introspection [10, 15]. The obvious explanation is that we exploit prior information about how face images transform, learned through extensive experience with other faces. Thus the key idea (see [12]), is to learn class-specific image-plane transformations from examples of objects of the same class and then to apply them to the real image of the new object in order to synthesize virtual views that can be used as additional examples in a view-based object recognition or graphic system. Prior knowledge about a class of objects may be known in terms of invariance properties. Poggio and Vetter [12] examined in particular the case of bilateral symmetry of certain 3D objects, such as faces. Prior information about bilateral symmetry allows the synthesis of new virtual views from a single real one, thereby simplifying the task of generalization in recognition of the new object under different poses. Bilateral symmetry has been used in face recognition systems [5] and psychophysical evidence supports its use by the human visual system [15, 13, 18].

A more flexible way to acquire information about how images of objects of a certain class change under pose, illumination and other transformations, is to learn the possible pattern of variabilities and class-specific deformations from a representative training set of views of generic or prototypical objects of the same class – such as other faces. Although our approach originates from the proposal of Poggio and Brunelli [11] and of Poggio and Vetter [12], for countering the curse-of-dimensionality in applications of supervised learning techniques, similar approaches with different motivations have been used in several different fields. In computer graphics, actor-based animation has been used to generate sequences of views of a character by warping an available sequence of a similar character. In computer vision the approach closest to the first part of ours is the *active shape models* of Cootes, Taylor, Cooper and Graham [14]. They build flexible models of known rigid objects by linear combination of labeled examples for the task of image search – recognition and localization. In all of these approaches the underlying representation of images of the new object are in terms of linear combinations of the shape of examples of representative other objects. Beymer, Shashua and Poggio [6] as well as Beymer and Poggio [5] have developed and demonstrated a more powerful version of this approach based on non-linear learning networks for generating new grey-level images of the same object or of objects of a known class. Beymer and Poggio [5] also demonstrated that new textures of an object can be generated by linear combinations of textures of different objects. In this paper, we extend and introduce the technique of linear classes to generate new views of an object. The technique is similar to the approach of [5, 6] but more powerful since it relies less on correspondence between prototypical examples and the new image.

The work described in this paper is based on the idea of *linear object classes*. These are 3D objects whose 3D shape can be represented as a linear combination of a sufficiently small number of prototypical objects. Linear object classes have the properties that new orthographic views of any object of the class under *uniform affine 3D transformations,* and in particular rigid transformations in 3D, can be generated exactly if the corresponding transformed views are known for the set of prototypes. Thus if the training set consist of frontal and rotated views of a set of prototype faces, any rotated view of a new face can be generated from a single frontal view – provided that the linear class assumption holds. In this paper, we show that the technique, first introduced for shape-only objects can be extended to their grey-level or colour values as well, which we call texture.

Key to our approach is a representation of an object view in terms of a *shape vector* and a *texture vector* (see also Jones and Poggio [9] and Beymer and Poggio [5]). The first gives the image-plane coordinates of feature points of the object surface; the second provides their colour or grey-level. *On the image plane* the shape vector reflects geometric transformation in the image due to a change in view point, whereas the texture vector captures photometric effects, often also due to viewpoint changes.

For linear object classes the new image of an object of the class is analyzed in terms of shape and texture vectors of prototype objects in the same pose. This requires correspondence to be established between all feature points of the prototype images – both frontal and rotated – which can be done in a off-line stage and does not need to be automatic. It also require correspondence between the new image and one of the prototypes in the same pose but does *not* need correspondence between different poses as in the parallel deformation technique of Poggio and Brunelli [11] and Beymer et al.[6].

The paper is organized as follows. The next section formally introduces *linear object classes*, first for objects defined only through their shape vector. Later in the section we extend the technique to objects with textures and characterize the surface reflectance models for which our linear class approach is valid. Section 3 describes an implementation of the technique for synthetic objects for which the linear class assumption is satisfied by construction. In the last section we address the key question of whether the assumption is a sufficiently good approximation for real objects. We consider images of faces and demonstrate promising results that indirectly support the conjecture that faces are a linear class at least to a first approximation. The discussion reviews the main features of the technique and its future extensions.

## 2 Linear Object Classes

Three-dimensional objects differ in shape as well as in texture. In the following we will derive an object representation consisting of a separate texture vector and a 2D-shape vector, each one with components referring to the same feature points, usually pixels. Assuming correspondence, we will represent an image as follows: we code its 2D-shape as the deformation field of selected feature points – in the limit pixels – from a reference image which serves as the origin of our coordinate system. The texture is coded as the intensity map of the image with feature points e.g. pixels set in correspondence with the reference image. Thus each component of the shape and the feature vector refers to the same feature point e.g. pixel. In this setting 2D-shape and texture can be treated separately. We will derive the necessary and sufficient conditions for a set of objects to be a linear object class.

### 2.1 Shape of 3D objects

Consider a 3D view of an three-dimensional object, which is defined in terms of pointwise features [12]. A 3D view can be represented by a vector $\mathbf{X} = (x_1, y_1, z_1, x_2, ....., y_n, z_n)^T$, that is by the $x, y, z$-coordinates of its $n$ feature points. Assume that $\mathbf{X} \in \Re^{3n}$ is the linear combination of $q$ 3D views $\mathbf{X}_i$ of *other* objects of the same dimensionality, such that:

$$\mathbf{X} = \sum_{i=1}^{q} \alpha_i \mathbf{X}_i. \qquad (1)$$

$\mathbf{X}$ is then the linear combination of $q$ vectors in a $3n$ dimensional space, each vector representing an object of $n$ pointwise features. Consider now the linear operator $L$ associated with a desired uniform transformation such as for instance a specific rotation in 3D. Let us define $\mathbf{X}^r = L\mathbf{X}$ the rotated 3D view of object $\mathbf{X}$. Because of the linearity of the group of uniform linear transformations $\mathcal{L}$, it follows that

$$\mathbf{X}^r = \sum_{i=1}^{q} \alpha_i \mathbf{X}_i^r \qquad (2)$$

Thus, *if a 3D view of an object can be represented as the weighted sum of views of other objects, its rotated view is a linear combination of the rotated views of the other objects with the same weights.* Of course for an arbitrary 2D view that is a projection of a 3D view, a decomposition like (1) does not in general imply a decomposition of the rotated 2D views (it is a necessary but not a sufficient condition).

*2D projections of 3D objects*
The question we want to answer here is, "Under which conditions the 2D projections of 3D objects satisfy equation (1) to (2)?" The answer will clearly depend on the types of objects we use and also on the projections we allow. We define:

*A set of 3D views (of objects)* $\{\mathbf{X}_i\}$ *is a* **linear object class** *under a linear projection* $P$ *if* $dim\{\mathbf{X}_i\} = dim\{\mathbf{P}\mathbf{X}_i\}$ *with* $\mathbf{X}_i \in \Re^{3n}$ *and* $\mathbf{P}\mathbf{X}_i \in \Re^p$ *and* $p < 3n$

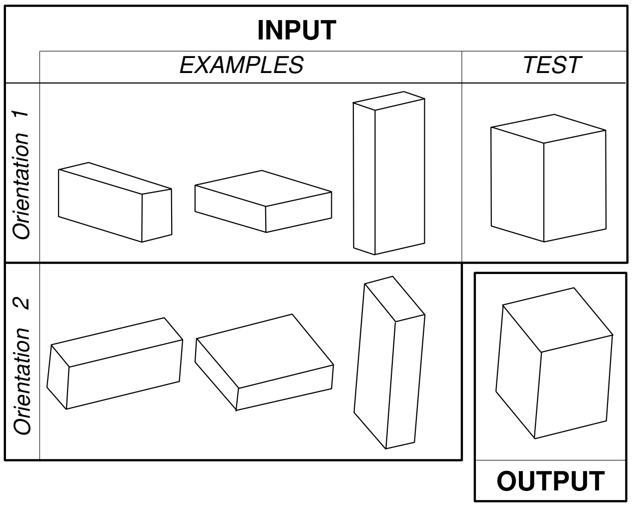


Figure 1: *Learning an image transformation according to a rotation of three-dimensional cuboids from one orientation (upper row) to a new orientation (lower row). The 'test' cuboid (upper row right) can be represented as a linear combination of the two-dimensional coordinates of the three example cuboids in the upper row. The linear combination of the three example views in the lower row, using the coefficients evaluated in the upper row, results in the correct transformed view of the test cuboid as output (lower row right). Notice that correspondence between views in the two different orientations is not needed and different points of the object may be occluded in the different orientations.*

This is equivalent to saying that the minimal number of basis objects necessary to represent a object is not allowed to change under the projection. Note that the linear projection $P$ is not restricted to projections from $3D$ to $2D$, but may also "drop" occluded points. Now assume $\mathbf{x} = P\mathbf{X}$ and $\mathbf{x}_i = P\mathbf{X}_i$ being the projections of elements of an linear object class with

$$\mathbf{x} = \sum_{i=1}^{q} \alpha_i \mathbf{x}_i \qquad (3)$$

then $\mathbf{x}^r = P\mathbf{X}^r$ can be constructed without knowing $\mathbf{X}^r$using $\alpha_i$ of equation (3) and the given $\mathbf{x}_i^r = P\mathbf{X}_i^r$ of the other objects.

$$\mathbf{x}^r = \sum_{i=1}^{q} \alpha_i \mathbf{x}_i^r. \qquad (4)$$

These relations suggest that we can use "prototypical" 2D views (the projections of a basis of a linear object class) and their known transformations to synthesize an operator that will transform a 2D view into a new 2D view when the object is a linear combination of the prototypes. In other words we can compute a new 2D view of such an object without knowing explicitly its three-dimensional structure. Notice also, that knowledge of the correspondence between equation (3) and equation (4) is not necessary (rows in a linear equation system can be exchanged freely). Therefore, the technique does not require to compute the correspondence between views from different viewpoints. In fact some points may be occluded. Figure 1 shows a very simple example of a linear object class and the construction of a new view of an object. Taking the 8 corners of a cuboid as features, a 3D view $\mathbf{X}$, as defined above, is an element of $\Re^{24}$; however, the dimension of the class of all cuboids is only 3, so any cuboid can be represented as a linear combination of three cuboids. For any projection, that preserve these 3 dimensions, we can apply equations (3) and (4). The projection in figure 1 projects all non occluded corners orthographically onto the image-plane ( $\mathbf{x} = P\mathbf{X} \in \Re^{14}$) preserving the dimensionality. Notice, that the orthographic projection of an exactly frontal view of a cuboid, which would result in a rectangle as image, would preserve 2 dimensions only, so equation (4) could not guarantee the correct result.

Before applying this idea to grey-level images, we would like to introduce a helpful change of coordinate systems in equations (3) and (4). Instead of using an absolute coordinate system, we represent the views as the difference to the view of a reference object of the same class, in terms of the spatial differences of corresponding feature points in the images. Subtracting on both sides of equations (3) and (4) the projection of a reference object gives us

$$\Delta\mathbf{x} = \sum_{i=1}^{q} \alpha_i \Delta\mathbf{x}_i \tag{5}$$

and

$$\Delta\mathbf{x}^r = \sum_{i=1}^{q} \alpha_i \Delta\mathbf{x}_i^r . \tag{6}$$

After this change in the coordinate system, equation (6) now evaluates to the new difference vector to the rotated reference view. The new view of the object can be constructed by adding this difference to the reference view.

## 2.2 Texture of 3D objects

In this section we extend our linear space model from a representation based on feature points to full images of objects. In the following we assume that the objects are isolated, that is properly segmented from the background. To apply equations (5) and (6) to images, the difference vectors between an image of a reference object and the images of the other objects have to be computed. Since the difference vectors reflect the spatial difference of corresponding pixels in images, this correspondence has to be computed first. The problem of finding correspondence between images in general is difficult and outside the scope of this paper. In the following we assume that the correspondence is given for every pixel in the image. In our implementation (see next section) we approximated this correspondence fields using a standard optical flow technique. For an image of $n$-by-$n$ pixels $\Delta\mathbf{x}$ in equations (5) and (6) are the correspondence fields of the images to a reference image with $\Delta\mathbf{x} \in \Re^{2n^2}$.

The computed correspondence between images enables a representation of the image that separates 2D-shape and texture information. The 2D-shape of an image is coded as a vector representing the deformation field relative to a reference image. The texture information is coded in terms of a vector which holds for each pixel the texture map that results from mapping the image onto the reference image through the deformation field. In this representation, all images – the shape vector and the texture vector – are vectorized relative to the reference image. Since the texture or image irradiance of an object is in general a complex function of albedo, surface orientation and the direction of illumination, we have to distinguish different situations.

Let us first consider the easy case of objects all with the same identical texture: corresponding pixels in each image have the same intensity or color. In this situation a single texture map (e.g. the reference image) is sufficient. Assuming a linear object class as described earlier, the shape coefficients $\alpha_i$ can be computed (equation 5) and result (equation 6) in the correspondence field from the reference image in the second orientation to the new 'virtual' image. To render the 'virtual' image, the reference image has to be warped along this correspondence field. In other words the reference image must be mapped onto the image locations given through the correspondence field. In Figure 2 the method is applied to grey level images of three-dimensional computer graphic models of five dog-like objects. The 'dogs' are shown in two orientations and four examples of this transformation from one orientation to the other are given. Only a single test view of a different dog is given. In each orientation, the correspondence from a chosen reference image (dashed box) to the other images is computed separately (see also section 'An implementation'). Since the dogs were created in such a way that the three-dimensional objects form a linear object class, the correspondence field to the test image could be decomposed exactly into the other fields (upper row). Applying the coefficients of this decomposition to the correspondence fields of the second orientation results in the correspondence of the reference image to a new image, showing the test object in the second orientation. This new image ("output" in the lower row) was created by simply warping the reference image along this correspondence field, since all objects had the same texture. Since in this test a three-dimensional model of the object was available, the synthesized output could be compared to the model. As shown in the difference image, there is only a small error, which can be attributed to minor errors in the correspondence step. This example shows that the method combined with standard image matching algorithms is able to transform an image in a way that shows an object from a new viewpoint.

Let us next consider the situation in which the texture is a function of *albedo* only, that is independent of the surface normal. Then a linear texture class can be formulated in a way equivalent to equations (1) through (4). This is possible since the textures of all objects were

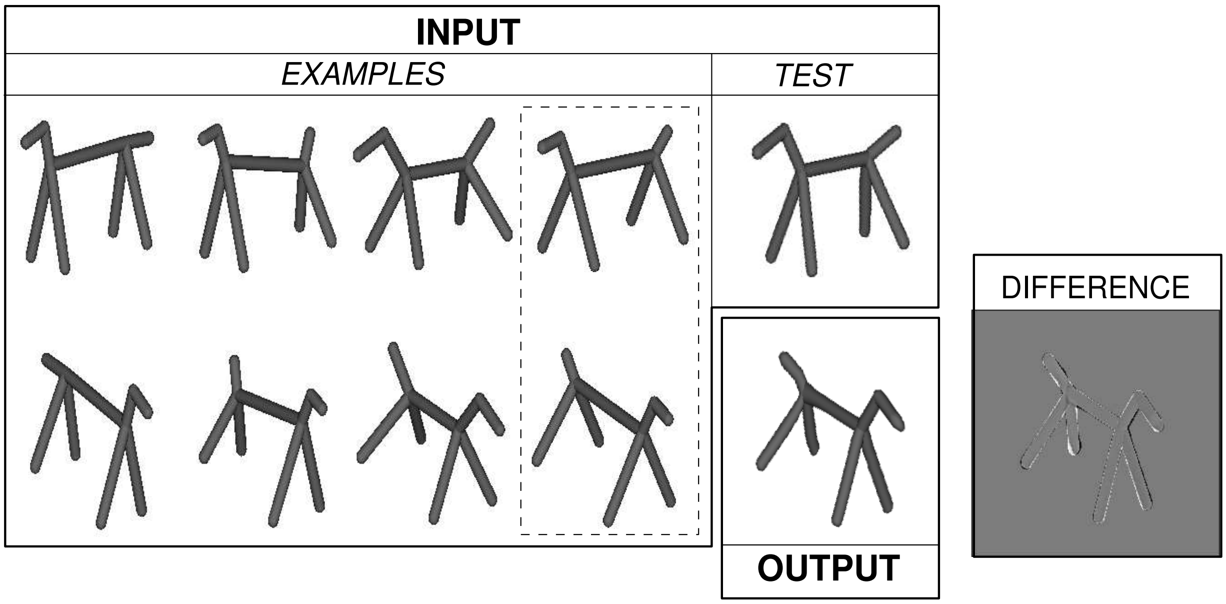


Figure 2: *Grey level images of an artificial linear object class are rendered. The correspondence between the images of a reference object (dashed box) and the other examples are computed separately for each orientation. The correspondence field between the test image and the reference image is computed and linearly decomposed into the other fields (upper row). A new correspondence field is synthesized applying the coefficients from this decomposition to the fields from the reference image to the examples in the lower row. The output is generated by forward warping the reference image along this new correspondence field. In the difference image between the new image and the image of the true 3D model (lower row, right), the missing parts are marked white whereas the parts not existing in an image of the model are in black.*

mapped along the computed deformation fields onto the reference image, so all corresponding pixels in the images are mapped to the same pixel location in the reference image. The equation

$$\mathbf{t} = \sum_{i=1}^{q} \beta_i \mathbf{t}_i \tag{7}$$

with $\beta_i$ (different to $\alpha_i$ in equation (3)) implies

$$\mathbf{t}^r = \sum_{i=1}^{q} \beta_i \mathbf{t}_i^r \tag{8}$$

assuming that the appearance of the texture is independent of the surface orientation and the projection does not change the dimensionality of the texture space. Here we are in the nice situation of a separate shape and texture space. In an application the coefficients $\alpha_i$ for the shape and coefficients $\beta_i$ for the texture can be computed separately. In face recognition experiments [5] the coefficients $\beta_i$ were already used to generate a new texture of a faces using textures of diffent faces. Figure 3 shows a test of this linear approach for a separated 2D-shape and texture space in combination with the approximated correspondence. Three example faces are shown, each from two different viewpoints accordingly to a rotation of 22.5°. Since the class of all faces has more than three dimensions a synthetic face image is used to test the method. This synthetic face is generated by a standard morphing technique [1] between the two upper left images. This ensures that the necessary requirements for the linear class assumption hold, that is the test image is a linear combination of the example images in texture and 2D-shape. In the first step for each orientation the correspondence between a reference face (dashed box) and the other faces is computed. Using the same procedure described earlier, the correspondence field to the test image is decomposed into the other fields evaluating the coefficients $\alpha_i$. Differently from figure 2, the textures are mapped onto the reference face. Now the texture of the test face can be linearly decomposed into the textures of the example faces. Applying the resulting coefficients $\beta_i$ to the textures of the example faces in the second orientation (lower row of figure 3), we generate a new texture mapped onto the reference face. This new texture is now warped along the new correspondence field. This new field is evaluated applying the coefficients $\alpha_i$ to the correspondence fields of the examples to the reference face in the second orientation. The output of this procedure is shown below the test image. Since the input is synthetic, this result can not be compared to the true rotated face, so it is up to the observer to judge the quality of the applied transformation of the test image.

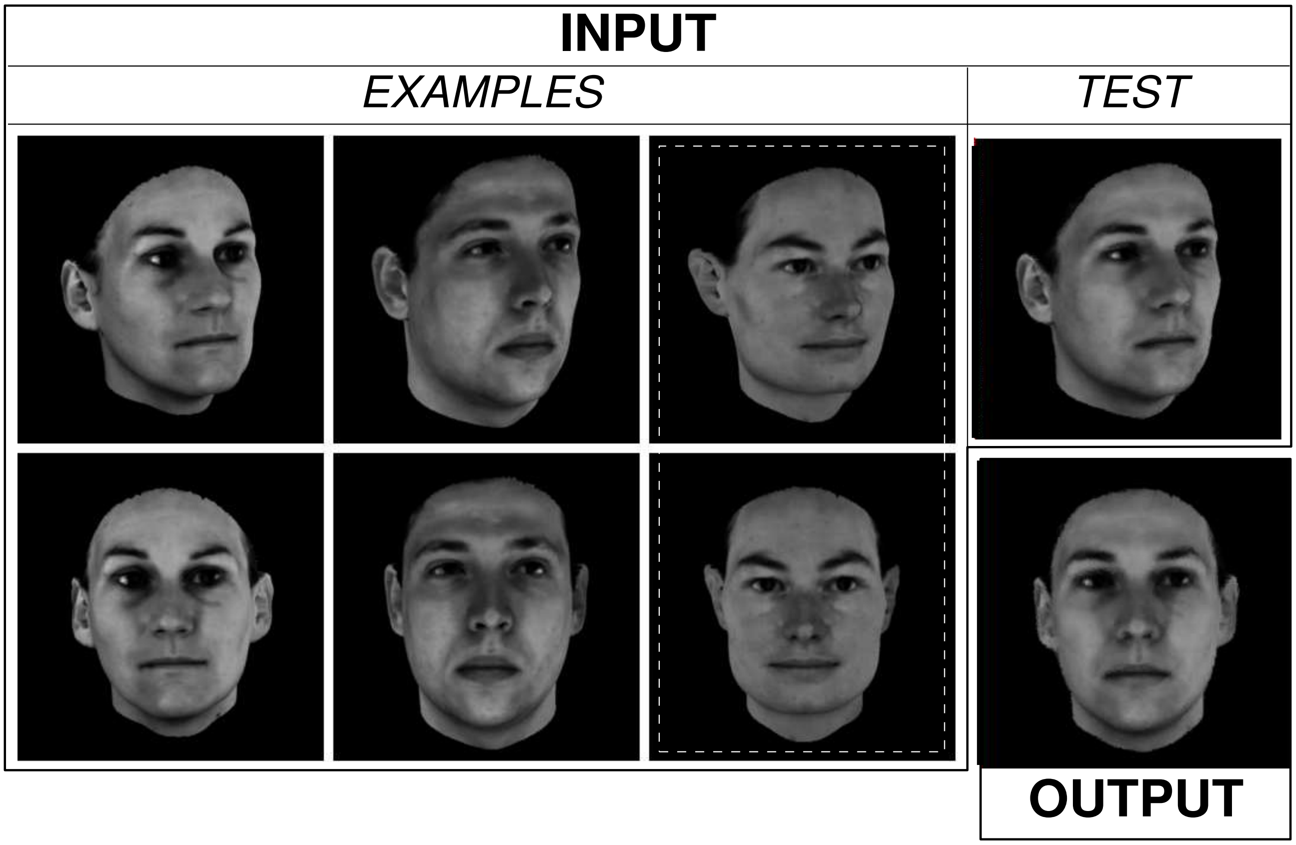


Figure 3: *Three human example faces are shown, each in two orientations (the three left columns), one of these faces is used as reference face (dashed box). A synthetic face, a 'morph' between the two upper left images, is used as a test face to ensure the linear combination constraint (upper right). The procedure of decomposing and synthesizing the correspondences fields is as described in figure 2. Additionally all textures, for each orientation separately, are mapped onto the reference face. Here the test texture is decomposed into the other example textures. Using the evaluated coefficients a new texture is synthesized for the second orientation on the reference face. The final output, the transformed test face, is generated by warping this new texture along the new synthesized correspondence field.*

## 3 An Implementation

The implementation of this method for grey-level pixel images can be divided into three steps. First, the correspondence between the images of the objects has to be computed. Second, the correspondence field to the new image has to be linearly decomposed into the correspondence fields of the examples. The same decomposition has to be carried out for the new texture in terms of the example textures. And finally we synthesize the new image, showing the object from the new viewpoint.

### 3.1 Computation of the Correspondence

To compute the differences $\Delta x$ used in equations (5) and (6), which are the spatial distances between corresponding points of the objects in the images, the correspondence of this points has to be established first. That means we have to find for every pixel location in an image, e.g. a pixel located on the nose, the corresponding pixel location on the nose in the other image. This is in general a hard problem. However, since all objects compared here are in the same orientation, we can often assume that the images are quite similar and that occlusion problems should usually be negligible. These conditions make it feasible to compare the images of the different objects with automatic techniques. Such algorithms are known from optical flow computation, in which points have to be tracked from one image to the other. We use a coarse-to-fine gradient-based gradient method [2] and follow an implementation described in [3]. For every point $x, y$ in an image $I$, the error term $E = \sum(I_x \delta x + I_y \delta y - \delta I)^2$ is minimized for $\delta x, \delta y$, with $I_x, I_y$ being the spatial image derivatives and $\delta I$ the difference of intensity of the two compared images. The coarse-to-fine strategy refines the computed displacements when finer levels are processed. The final result of this computation $(\delta x, \delta y)$ is used as an approximation of the spatial displacement ($\Delta x$ in equation (5)and (6))of a pixel from one image to the other. The correspondence is computed in the direction towards the reference image from the example and the test images. As a consequence all vector fields have a common origin at the pixel locations of the reference image.

### 3.2 Learning the Linear Transformation

The decomposition of a given correspondence field in equation (5) and the composition of the new field in equation (6) can be understood as a single linear transformation. First, we compute the coefficients $\alpha_i$ for the optimal decomposition (in the sense of least square). We decompose a "initial" field $\Delta\mathbf{x}$ to a new object $X$ into the "initial" fields $\Delta\mathbf{x}_i$ to the $q$ given prototypes by minimizing

$$||\Delta\mathbf{x} - \sum_{i=1}^{q} \alpha_i \Delta\mathbf{x}_i||^2. \quad (9)$$

We rewrite equation (5) as $\Delta\mathbf{x} = \mathbf{\Phi}\boldsymbol{\alpha}$ where $\mathbf{\Phi}$ is the matrix formed by the $q$ vectors $\Delta\mathbf{x}_i$ arranged columnwise and $\boldsymbol{\alpha}$ is the column vector of the $\alpha_i$ coefficients. Minimizing equation (9) gives

$$\boldsymbol{\alpha} = (\mathbf{\Phi})^{+}\Delta\mathbf{x}. \quad (10)$$

The observation of the previous section implies that the operator $L$ that transforms $\Delta\mathbf{x}$ into $\Delta\mathbf{x}^r$ through $\Delta\mathbf{x}^r = L\Delta\mathbf{x}$, is given by

$$\Delta\mathbf{x}^r = \mathbf{\Phi}^r\boldsymbol{\alpha} = \mathbf{\Phi}^r\mathbf{\Phi}^{+}\Delta\mathbf{x} \quad as \quad L = \mathbf{\Phi}^r\mathbf{\Phi}^{+} \quad (11)$$

and thus can be learned from the 2D example pairs $(\Delta\mathbf{x}_i, \Delta\mathbf{x}_i^r)$. In this case, a one-layer, linear network (compare Hurlbert and Poggio, 1988) can be used to learn the transformation $L$. $L$ can then transform a view of a novel object of the same class. If the $q$ examples are linearly independent $\mathbf{\Phi}^{+}$ is given by $\mathbf{\Phi}^{+} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T$; in the other cases equation (9) was solved by an SVD algorithm.

Before decomposing the new texture into the example textures, all textures have to be mapped onto a common basis. Using the correspondence, we warped all images onto the reference image. In this representation the decomposition of the texture can be performed as described above for the correspondence fields.

### 3.3 Synthesis of the New Image.

The final step is image rendering. Applying the computed coefficients to the examples in the second orientation results in a new texture and the correspondence fields to the new image. The new image can be generated combining this texture and correspondence field. This is possible because both are given in the coordinates of the reference image. That means that for every pixel in the reference image the pixel value and the vector pointing to the new location are given. The new location generally does not coincide with the equally spaced grid of pixels of the destination image. A commonly used solution of this problem is known as forward warping [19]. For every new pixel, we use the nearest three points to linearly approximate the pixel intensity.

## 4 Is the linear class assumption valid for real objects?

For man made objects, which often consist of cuboids, cylinders or other geometric primitives, the assumption of linear object classes seems almost natural. However, are there other object classes which can be linearly represented by a finite set of example objects? In the case of faces it is not clear how many example faces are necessary to synthesize any other face and in fact, it is unclear if the assumption of a linear class is appropriate at all. The key test for the linear class hypothesis in this case is how well the synthesized rotated face approximates the "true" rotated face. We tested our approach on a small set of 50 faces, each given in two orientations (22.5° and 0°). Figure 4 shows four tests using the same technique as described in figure 3. In each case one face was selected as test face and the 49 remaining faces were used as examples. Each test face is shown on the upper left and the output image produced by our technique on the lower right, showing a rotated test face. The true rotated test face from the data base is shown on the lower left. We also show in the upper right the synthesis of the test face through the 49 example faces in the test orientation. This reconstruction of the test face should be understood as the projection of the test face into the shape and texture space of the other 49 example faces. A perfect reconstruction of the test face would be a necessary (not sufficient!) requirement that the 50 faces are a linear object class. The results are not perfect but, considering the small size of the example set, the reconstruction is quite good. The similarity of the reconstruction to the input test face allows to speculate that an example set size of the order of hundred faces may be sufficient to construct a huge variety of different faces. We conclude that the linear object class approach may be a satisfactory approximation even for complex objects as faces. On the other hand it is obvious that the reconstruction of every specific mole or wrinkle in a face requires to an almost infinite number of examples. To overcome this problem, correspondence between images taken from different viewpoints should be used to map the specific texture on the new orientation [9, 5].

## 5 Discussion

Linear combinations of images of a single object have been already successfully used to create a new image of that object [16]. Here we created a new image of an object using linear combinations of images of different objects of the same class. Given only a single image of an object, we are able to generate additional synthetic images of this object under the assumption that the "linear class" property holds. This is demonstrated not only for objects purely defined through their shape but also for smooth objects with texture.

This approach based on two-dimensional models does not need any depth information, so the sometime difficult step of generating three-dimensional models from two-dimensional images is superfluous. Since no correspondence is necessary between images, representing objects in different orientations, fully automated algorithms can be applied for the correspondence finding step. For object recognition tasks our approach has several implications. Our technique can provide additional artificial example images of an object when only a single image is given. On the other hand the coefficients, which result from a decomposition of shape and texture into example shapes and textures give us already a representation of the object which is invariant under any

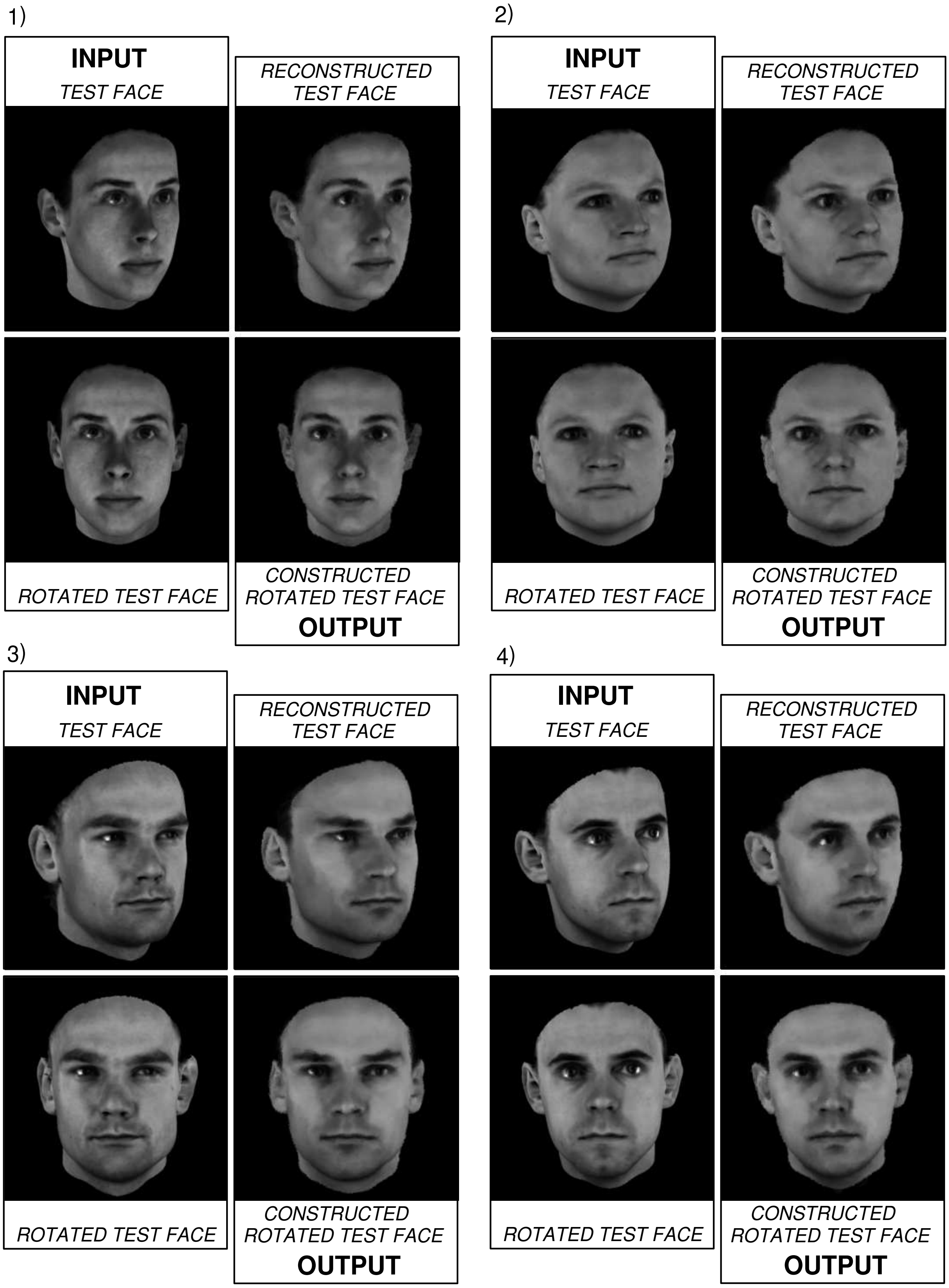


Figure 4: *Four examples of artificially rotated human faces, using the technique described in figure 3 are shown. Each test face (upper left) is "rotated" using 49 different faces (not shown) as examples, the results are marked as output. Only for comparison the "true" rotated test face is shown on the lower left (this face was not used in the computation). The difference, between synthetic and real rotated face is due to the incomplete example set, since the same difference can already be seen in the reconstruction of the input test face using the 49 example faces (upper right).*

affine transformation.

In an application our approach is confronted with two types of problems. As in any approach based on flexible models, there is the problem of finding the correspondence between model and image. In our implementation we used a general method for finding this correspondence. However, if the class of objects is known in advance, a method specific to this object class could be used [9, 7]. In this case the correspondence field is linearly modeled by a known set of deformations specific to that class of objects.

A second problem, specific to our approach is the existence of linear object classes and the completeness of the available examples. This is equivalent to the questions of whether object classes defined in terms of human perception can be modeled through linear object classes. Presently there is no final answer to this question, apart for simple objects like (e.g. cuboids, cylinders), where the dimensionality is given through their mathematical definition. The application of the method to a small example set of human faces, shown here, provides preliminary promising results at least for some faces. It is, however, clear that 50 example faces are not sufficient to model accurately all human faces. Since our linear model allows to test the necessary conditions for an image being a member of a linear object class, the model can detect images where a transformation fails. This test can be done by measuring the difference between the input image and its projection into the example space, which should ideally vanish.

Our implementation, as described in our examples, can be improved by applying the linear class idea to independent parts of the objects. In the face case, a new input face was linearly approximated through the complete example faces, that is for each example face a single coefficient (for texture and 2D-shape separately) was computed. Assume noses, mouths or eyes span separated linear subspaces, then the dimensionality of the space spanned by the examples will be multiplied by the number of subspaces. So in a new image the different parts will be approximated separately by the examples, that will increase the number of coefficients used as representation and will also improve the reconstruction.

Several open questions remain for a fully automated implementation. The separation of parts of an object to form separated subspaces could be done by computing the covariance between the pixels of the example images. However, for images at high resolution, this may need thousands of example images. Our linear object class approach also assumes that the orientation of an object in an image is known. The orientation of faces can be approximated computing the correlation of a new image to templates of faces in various orientations [4]. It is not clear how precisely the orientation should be estimated to yield satisfactory results.

# Appendix

## A Decomposing objects into parts

In the previous section we considered learning the appropriate transformation from full views. In this case, the examples (prototypes) must have the same dimensionality as a full view. Our arguments above show that dimensionality determines the number of example pairs needed for a correct transformation. This section suggests that components of an object – i.e. a subset of the full set of features – that are element of the same object class may be used to learn a single transformation with a reduced number of examples, because of the smaller dimensionality of each component. We rewrite equation (1) to $X = \Phi\boldsymbol{\alpha}$ where $\Phi$ is the matrix formed by the $q$ vectors $X_i$ arranged column-wise and $\boldsymbol{\alpha}$ is the column vector of the $\alpha_i$ coefficients. The basic components in which a view can be decomposed are given by the irreducible submatrices $\Phi^{(i)}$ of the structure matrix $\Phi$ so that $\Phi = \Phi^{(1)} \oplus .... \oplus \Phi(k)$. Each submatrix $\Phi^{(i)}$ represents an isolated object class, formed by a subset of feature points which we would like to call a part of an object. As an example, for objects composed by two cuboids in general six examples would be necessary since all 3D views of objects composed of two cuboids span a six-dimensional space (we suppose a fixed angle between the cuboids). However, this space $\Phi$ is the direct sum $\Phi = \Phi^{(1)} \oplus \Phi^{(2)}$ of two three-dimensional subspaces, so three examples are sufficient. Notice the $\Phi^{(1)}$ and $\Phi^{(2)}$ are only identical when both are in the same orientation. This shows that the problem of transforming the 2D view $\mathbf{x}$ of the 3D objects $\mathbf{X}$ into the transformed 2D views $\mathbf{x}^r$, can be treated separately for each component $\mathbf{x}^{(\mathbf{k})}$.

## References


[1] T. Beier and S. Neely. Feature-based image metamorphosis. In *SIGGRAPH '92 proceedings*, pages 35–42, Chicago, IL, 1992.

[2] J.R. Bergen and E.H. Adelson. Hierarchical, computationally efficient motion estimation algorithm. 4:35, 1987.

[3] J.R. Bergen and R. Hingorani. Hierarchical motion-based frame rate conversion. Technical report, David Sarnoff Research Center Princeton NJ 08540, 1990.

[4] D. Beymer. Face recognition under varying pose. A.I. Memo No. 1461, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1993.

[5] D. Beymer and T. Poggio. Face recognition from one model view. In *ICCV proceedings*, 1995.

[6] D. Beymer, A. Shashua, and T. Poggio. Example-based image anaysis and synthesis. A.I. Memo No. 1431, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1993.

[7] David Beymer. Vectorizing face images by interleaving shape and texture computations. to appear as A.I. Memo, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1995.

[8] A. Hurlbert and T Poggio. Synthetizing a color algorithm from examples. *SCIENCE*, 239:482–485, 1988.

[9] M. Jones and T. Poggio. Model-based matching of line drawings by linear combination of prototypes. In *ICCV proceedings*, 1995.

[10] P. Kalocsai, I.Biederman, and E.E. Cooper. To what extent can the recognition of unfamiliar faces be accounted for by a representation of the direct output of simple cell. In *Annual Meeting of the Association for Research in Vision and Opthalmology*, Sarasota, Fl, 1994.

[11] T. Poggio and R. Brunelli. A novel approach to graphics. Technical report 1354, MIT Media Laboratory Perceptual Computing Section, 1992.

[12] T. Poggio and T. Vetter. Recognition and structure from one 2D model view: observations on prototypes, object classes, and symmetries. A.I. Memo No. 1347, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1992.

[13] P.G. Schyns and H.H. Bülthoff. Sparse observations on cortical mechanisms for object recognition and learning. A.I. Memo No. 1404, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1993.

[14] D.H. Cooper T.F. Cootes, C.J. Taylor and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 60:38–59, 1995.

[15] N. Troje and H.H. Bülthoff. Face recognition under varying pose: The role of texture and shape. *submitted*, 1995.

[16] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:992–1006, 1991.

[17] T. Vetter and T. Poggio. Symmetric 3D objects are an easy case for 2D object recognition. *Spatial Vision*, 8(4):443–453, 1995.

[18] T. Vetter, T. Poggio, and H.H. Bülthoff. The importance of symmetry and virtual views in three-dimensional object recognition. *Current Biology*, 4:18–23, 1994.

[19] Georg Wolberg. *Image Warping*. IEEE Computer Society Press, Los Alamitos CA, 1990.