Image Metamorphosis

BRUNO COSTA[1,2]
LUCIA DARSA[1,2]
JONAS DE MIRANDA GOMES[1]

[1]IMPA–Instituto de Matemática Pura e Aplicada
Estrada Dona Castorina, 110
22460 Rio de Janeiro, RJ, Brasil
jonas@visgraf.impa.br

[2]Pontifícia Universidade Católica - Rio
Departamento de Informática
Rua Marquês de São Vicente, 225
22453 Rio de Janeiro, RJ, Brasil
bruno | lucia@visgraf.impa.br

Abstract. Metamorphosis is simply a complete change of shape, structure or substance of one object into another (Webster, 1989). This kind of transformation is a very effective tool that has been widely used by the film and video industry. In this paper we give a conceptual overview of the metamorphosis problem and define precisely the meaning of image metamorphosis (also called image morphing). We analyze existing morphing techniques under this conceptual framework, and propose a new technique that explores some advantages of different existing methods.

Keywords: Digital Image, Morphing, Image Warping, Color Transformation.

1 Introduction

Metamorphosis is present in the everyday evolution of forms and shapes. It has been used for decades in the film industry, and more recently in the video industry. Real world metamorphosis sequences of very slow processes can be done by taking time spaced shots and showing them in a regular frame rate.

The use of metamorphosis in the entertainment industry dates back from the old cross-dissolve technique that consists in the superposition of one image fade out sequence, with a fading in sequence of the new shape. More recently, computer graphics and image processing techniques have made it possible to create a large amount of more convincing metamorphosis effects. In this paper we will discuss both the conceptual and practical aspects of image metamorphosis, also known as image morphing.

The structure of the paper is as follows: in section 2 we discuss space deformations and k-parameter transformation groups; in section 3 we apply the results of the previous section to study image transformations; in section 4 we study image morphing; in section 5 we discuss different techniques for image warping that are useful in the construction of image morphing sequences; in section 6 we discuss some implementation issues in a morphing system and show some results of an implemented image morphing technique. Finally in section 7 we conclude with some general remarks and possible future works.

2 Space Deformations

In this paper space means a vector space. The euclidean n-dimensional space will be denoted by $\mathbb{R}^n$; for $n = 2$ we have the plane and for $n = 3$, we capture the mathematical abstraction of the usual $3D$-space. We are interested in the deformation of objects in the space. This can be achieved in two different ways:

- space deformations;
- intrinsic deformations.

A *space deformation*, also called *global deformation*, of the space, is a map $T : \mathbb{R}^n \to \mathbb{R}^n$. An *intrinsic deformation* changes the object geometry or topology without affecting the surrounding space. Very common global deformations are linear maps, and in particular the rigid motions of the space. A simple, but effective, family of non-linear global deformations of the space, with applications to modeling, can be found in (Barr, 1984). Non-linear global

deformations are much more flexible in the applications, but they are harder to specify and computationally much more expensive.

Global deformations are, in general, used to change the shape of solid objects of the space; intrinsic deformations are used to reshape non-solid objects in the space (e.g. curves and surfaces);

Animation. In most applications we are indeed interested in observing the intermediate states of the metamorphosis sequence between two objects. For this purpose, instead of the final deformation, we need a family of intermediate transformations.

This is well captured by the mathematical concept of *group of transformations*. A k-parameter group of transformations of the space $\mathbb{R}^m$ is a map $T: \mathbb{R}^m \times \mathbb{R}^k \to \mathbb{R}^n$. For each vector v_0 of $\mathbb{R}^k$, k coordinates, we get a transformation $T_{v_0}: \mathbb{R}^m \to \mathbb{R}^n$. The transformation group is continuous (differentiable) if the map T is continuous (differentiable). In most of the applications in the film and video industry we have a one-parameter group of transformation (i.e. $\mathbb{R}^k = \mathbb{R}$) and the one-dimensional parameter, t, is interpreted as being the time. By properly discretizing the time parameter, t, we get an animated morphing sequence with the desired frame rate. Also, for most applications we need at least a continuous family in order to get a pleasant morphing animation sequence.

Given two objects $\mathcal{O}_1$ and $\mathcal{O}_2$ in the space a *metamorphosis*, or simply *morphing*, is a continuous k parameter group of transformations, such that there exists two different parameters vector v_0 and v_1, satisfying $T_{v_0}(\mathcal{O}_1) = \mathcal{O}_1$, and $T_{v_1}(\mathcal{O}_2) = \mathcal{O}_2$. Intuitively the object $\mathcal{O}_1$ is continuously deformed into the object $\mathcal{O}_2$ with k degrees of freedom.

In this work we are interested in digital image morphing. In order to understand this problem properly, we must first understand a good mathematical model of the image space.

3 Image transformations

For our purposes, an adequate mathematical model is to consider an image as a function $f: U \subset \mathbb{R}^2 \to C$, where U is a subset of the plane $\mathbb{R}^2$ and C is a vector space (in general a finite dimensional color space representation). In order to represent such an image model in the computer, U and C are discretized and some digital coding scheme is used. This codified and discretized image is called *digital image* (Rosenfeld and Kak, 1976).

The space of images, $\mathcal{I}$, is therefore a space of functions, $f: U \subset \mathbb{R}^2 \to C$, and has a natural vector space structure. A group of transformations on the image space $\mathcal{I}$ is a family of transformations that map one image function into another.

If U is the domain of an image function f and (x, y) are the coordinates of a pixel on some coordinate system of U, the image function is completely characterized by its values $f(x, y)$, $(x, y) \in U$. Therefore an image transformation is accomplished by changing either the function domain, *domain transformation*, or the function values, *color transformation*.

The first kind of transformation is called *image warping* in the literature (Wolberg, 1990). It is used in dozens of special effects in the film and video industry (e.g. the Ampex Digital Optics box, ADO, uses a projective warping of the image). The second kind, *color transformation*, also has widespread use in the film and video industry (e.g. cross-dissolve between two images and color-map manipulation techniques)

3.1 Image Warping

The diagram in Figure 1 clarifies the image warping transformation. In general $U = V$, and the map g is a diffeomorphism that performs a change of coordinates in the image domain. Each pixel (u_1, u_2) in U is associated with a pixel (v_1, v_2) in the domain V, such that $(v_1, v_2) = g(u_1, u_2)$. In some applications g is not a diffeomorphism, therefore we do not have a "good" change of coordinates, and much more difficulties arise to work with.

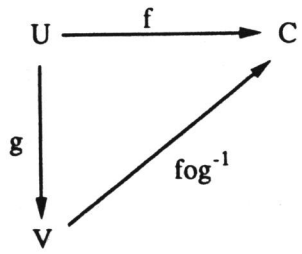


Figure 1: *Commutative diagram illustrating the warp, $f \circ g^{-1}$, of the image f by a deformation g.*

Direct and Inverse Mapping. There are two ways to compute the pixels values under the transformation g: direct and inverse mapping. In the first case g is applied to each pixel of the image domain U in order to find its coordinates on the image domain V. Inverse mapping, on the other, hand works backwards: for each pixel coordinate in the discretized image domain V, we look for the pixels in the domain U such that when transformed by g overlap

with the area of the original pixel. Inverse mapping is more effective because it assures that every output pixel is computed and no work is wasted in computing unnecessary pixels. For more details the reader should consult (Wolberg, 1990).

Aliasing. In a digital image the pixel is not really the mathematical abstraction of a point in the plane, but it occupies a finite area in the image domain. Therefore, even when the warp function g is a diffeomorphism, the pixels overlap when they undergo the transformation: several pixels are mapped to one pixel. Also, numerical errors arise in the computation of the image warping. In order to minimize these problems, it would be desirable to work with a continuous version of the image, i.e. non-discretized domain and non-quantized color space. Very delicate problems arise in this context involving the pipeline of operations

$$\text{discretize} \rightarrow \text{reconstruct} \rightarrow \text{transform} \rightarrow \text{sample}$$

with the image. An elementary discussion of these problems can be found in (Gomes e Velho, 1992). For a more detailed study the reader should consult (Wolberg, 1990).

3.2 Color transformation

The diagram in Figure 2 illustrates a color transformation h of an image f. The domain U of the image remains unchanged, and the color space is transformed into a new one D. In general $D = C$ and h is just a deformation of the color space C.

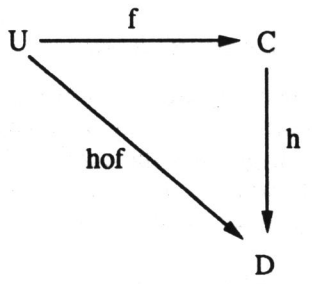


Figure 2: *Commutative diagram illustrating the warp, $f \circ g$, of the image f by a deformation g.*

A very useful color transformation technique is the *cross-dissolve*. Given two images $f(x, y)$ and $g(x, y)$, a cross-dissolve between them is a one-parameter group of transformations of the image color space

$$T(x, y, t) = G(x, y, t).f(x, y) + H(x, y, t).g(x, y),$$

such that $G(x, y, 0) = 1$, $H(x, y, 0) = 0$, $G(x, y, 1) = 0$, $H(x, y, 1) = 1$, and $G(x, y, t) + H(x, y, t) = 1$, for all t.

Note that the above equation explores the vector space structure of the color space. The functions G and H simply interpolate between the color values of the image pixels. The dependence of G and H on the pixel coordinates (x, y) enable us to get an adaptive cross-dissolve between the two images, that is, the rate of change of each pixel color depends on the pixel coordinates. In most of the applications a linear, non-adaptive, cross-dissolve is used. In this case we have

$$G(x, y, t) = 1 - t \quad \text{and} \quad H(x, y, t) = t.$$

4 Image Morphing

Let's first summarize our morphing definition for image morphing. Given two digital images $f, g: U \subset \mathbb{R}^2 \rightarrow C$, a morphing between them is a continuous k-parameter group of transformations, $T: \mathcal{I} \times \mathbb{R}^k \rightarrow C$, of the image space $\mathcal{I}$, such that for some vector parameters v_0 and v_1 we have $T(f, v_0) = f$ and $T(g, v_1) = g$. When $k = 1$, that is $\mathbb{R}^k = \mathbb{R}$, the group parameter is denoted by t, and is interpreted as time. In this case by a simple reparameterization we may suppose that $v_0 = t_0 = 0$ and $v_1 = t_1 = 1$. In order to obtain a finite representation of the morphing between two images we must discretize the parameter group $\mathbb{R}^k$, obtaining a finite sequence of images

$$\begin{aligned} f \;=\; & T(f, v_0) \rightarrow T(f, v_1) \rightarrow \cdots \rightarrow \\ & \rightarrow T(f, v_{n-1}) \rightarrow T(f, v_n) = g. \end{aligned}$$

When exhibited with the desired frame rate, the above sequence produces an animation of the morphing transformation between f and g.

Since a morphing transforms one image into another one, it performs both an image warping and a color transformation. In general the morphing is done in an uncoupled way: the image warping accomplishes an alignment phase, where each pixel of the source image changes until its coordinates are the same as those of the corresponding pixel in the target image; color transformation then changes the pixel values source image to those of the target image. In practice a linear cross-dissolve gives reasonable results in the color transformation.

4.1 Morphing Animated Sequences

Intuitively an image morphing accomplishes a continuous deformation between two images with k degrees of freedom. These degrees of freedom can be interpreted in different ways in the applications. When

$k = 1$, the parameter is interpreted as time, and we obtain a deformation between two fixed images (geometrically we have a continuous path on the image space that starts at one image and finishes at the other one). Although the morphing of fixed images is an interesting effect, it still lacks the natural movement in an animated sequence: a moving subject must stop, metamorphose, and then continue its movement with the new shape. In order to avoid this, we must be able to do morphing of animated sequences. This can be formalized by using a 2-parameter group of transformations, as will be explained in the next paragraph.

Morphing of two animated sequences can be interpreted as morphing with two degrees of freedom $T: \mathcal{I} \times \mathbb{R}^2 \rightarrow \mathcal{I}$. Consider two images f, g, and parameter vector (u, v) defined on the plane domain $[u_0, u_1] \times [v_0, v_1]$. Starting with f and fixing the parameter $v = v_0$ we get an animated sequence, $T(f, u, v_0)$, that begins with $T(f, u_0, v_0)$ and ends at the image $T(f, u_1, v_0)$. In an analogous way, starting with the image g and fixing the parameter $v = v_1$ we get an animated sequence, $T(f, u, v_1)$, that begins with $T(f, u_0, v_1)$ and ends at the image $T(f, u_1, v_1)$. Now for each fixed value, $\overline{u}$, of the parameter u we have a group of deformation with one parameter v, $T(\cdot, \overline{u}, v)$, that performs a morphing between the images $T(f, \overline{u}, v_0)$ and $T(g, \overline{u}, v_1)$ of the two animation sequences defined above. For any continuous path $\gamma(t)$ of the parameter space $[u_0, u_1] \times [v_0, v_1]$ such that $\gamma(0) = (u_0, v_0)$ and $\gamma(1) = (u_1, v_1)$ the one parameter group of transformations $G(\cdot, t) = T(\cdot, \gamma(t))$ performs an animated morphing between the two animated sequences above.

Geometrically the above computations are easy to understand: The two parameter group of transformation, $T(\cdot, u, v)$, defines a parameterized surface of the image space $\mathcal{I}$, and a morphing of two animated sequences is defined by a path on this surface. There are many choices of the morphing path. A natural choice is to get the diagonal path, as illustrated in figure 3.

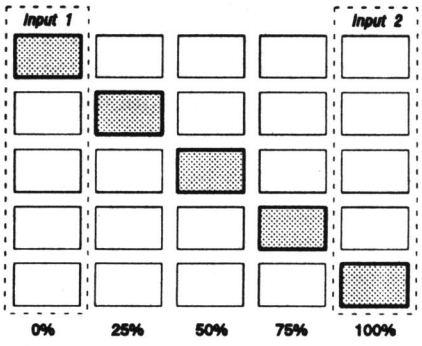


Figure 3: *Animated sequences morphing: The first and last columns represent two animated sequences. Each row represents a morphing sequence between the two extremes. The highlighted boxes along the diagonal are a possible animated morphing sequence.*

5 Image Warping Techniques

As stated before, the color transformation in a morphing sequence is in general accomplished by an straightforward cross-dissolve. It is the warping phase that will dictate the quality and the performance of the morphing rendering. The specification of a generic image warping can be done in several ways, and the choice of a particular method will influence both the user interface and the deformation algorithm.

An image warping can be specified by a mapping function that establishes a spatial correspondence between all points in the input image and in the deformed image. Among the well known mappings, are the projective transformations, which are usually described simply by an homogeneous transformation matrix. In practice, however, much more powerful mapping functions are needed, with, naturally, much more complex descriptions, which must still be relatively easy to manipulate.

By using some kind of local control on the transformation, we are able to map distinguished features in one image into distinguished features on the other image. This is an important issue in order to apply the warping technique to do image morphing.

5.1 Warping Specification

A generic image warping will be usually specified by two sets of "marks": one defining a parameterization of the input image, and the other a deformation of that parameterization. The first set will be referred to as the "source set" and the second, as the "destination set".

The best technique to be used to get the image warping is greatly influenced by the way the user specifies the warping transformation. In what follows we will describe two morphing specification techniques:

- mesh warping;
- feature-based warping;

Mesh Warping

In the mesh based approach, two combinatorially

identical meshes are overlaid over the image. As each mesh defines a coordinate system on the image domain, the warp is obtained by doing a change of coordinates between the two systems.

A mesh defines a partition of the image domain, and the structure of decomposition obtained is greatly dependent on the mesh specification. Well structured decompositions are very easy to work with, while arbitrary irregular decompositions are more difficult to represent.

An early implementation of a morphing technique using a mesh warping specification was done at Industrial Light and Magic, and was used to create the special effects of the movie "Willow " (see (Wolberg, 1990)). More recently the technique was used in the special effects of the movie "Terminator 2".

Feature-based Warping

In this kind of warping specification, only distinguished features and their transformation are specified by the user. The warp is computed in such a way to map each feature of the image to the corresponding transformed position.

An implementation of a morphing system using feature-based specification was done at Pacific Data Images, considering as features oriented line segments (Beier and Neely, 1992). This system has been used to create the morphing sequence of Michael Jackson's video clip "Black or White".

An important particular case of a feature-based warping is the *point-based specification*, where each feature is distinguished by a point (pixel) of the image. An implementation of morphing using a point-based specification has been reported by Ken Perlin (Perlin, 1992).

5.2 Warping Algorithms

Corresponding to each kind of warping specification, there is a different technique to construct the deformation of the image domain. These techniques are described in this section.

The image warping is a change of coordinates in the domain of the image. A natural way to warp an image is to introduce a new coordinate system adapted to the image details, and proceed to a change of coordinates. Techniques for warping construction are as abundant as techniques for deformation construction. Some of them will be described below.

Triangle Mesh Warping

A simple way to specify a generic image warping is to use triangle meshes. By triangulating each image domain, we introduce a coordinate system using barycentric coordinates (Preparata, 1985). Any two triangles can be mapped using a linear transformation. Also, if the triangle meshes have the same combinatorial properties, this transformation can be extended to a global, piecewise linear transformation that executes the change of coordinates. A scan conversion of each triangle in the destination mesh (Foley et alli, 1990), coupled with the above mapping, produces an inverse mapping algorithm to deform an image. Among the many problems of this approach are providing a convenient user interface and keeping the consistency of the mesh representation during its creation.

An important point to stress is the fact that the change of coordinates is done by a piecewise linear map. Therefore, some artifacts could possibly appear during the warping. We did an experimental implementation, and some tests showed that the artifacts were more noticeable when the mesh had very irregular triangles. Besides using a "regular" triangulation, a more expensive solution consists in the use of a higher degree interpolating function.

The creation of the entire mesh, triangle by triangle, is a long and tedious process that can be made easier through the use of some kind of automatic triangulation. In this case, all that is required from the user is to specify two sets of points, the "source" and the "destination". Then, after an automatic triangulation of the points in the destination set, the triangle mesh algorithm above can be applied. Note that, since a Delaunay triangulation maximizes the smallest of the internal angles of each triangle (Figueiredo e Carvalho, 1991) - producing more regular triangles - it may be used to attempt to attenuate the edge artifacts. This automatic behavior reveals a problem that could be avoided in the manual construction of the meshes, the *foldover*, where the image "folds" upon itself. As the association of the points is arbitrary, the mesh in the source set corresponding to the automatic triangulation in the destination is not necessarily a valid planar subdivision (see figure 4).

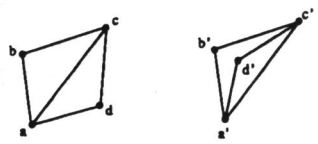


Figure 4: *Triangle mesh foldover: left shows the destination, and right shows the source with the same combinatorial structure.*

Two-Pass Spline Mesh Warping

A quite different approach to construct the change of coordinates consists in constructing explicitly a new coordinate system by specifying some of its coordinates curves. The use of splines to specify the regular meshes that define the coordinate systems presents several advantages. First, the use of a regular structure, instead of an arbitrary topology, significantly simplifies both the data structures and the mapping functions. Also, this mapping can be shown to be separable, i.e., it can be performed in two independent passes, one horizontal, that produces an intermediate image, and one vertical, that uses this intermediate image to produce the final output (Wolberg, 1990). This brings the problem of deforming a $2D$ image down to a $1D$ problem, greatly reducing the complexity of the calculations – specially for antialiasing – and takes advantage of current computer memory organization. It is possible to use any interpolating spline formulation with at least zero degree continuity.

The horizontal pass of the algorithm starts by creating an intermediate mesh I that includes just the horizontal displacements from the source mesh S to the destination mesh D, that is, each point in I has the same x coordinate of the corresponding point in D, and the y coordinate of the point in S (figure 5). Each column of S and I is then fitted with an interpolating spline – we call S_S the set of vertical splines in S, and I_S the set of vertical splines in I. Each horizontal scanline is intercepted independently with S_S and I_S (figure 6). The x coordinates of the interceptions with I_S and the x coordinates of the interceptions with S_S are then fitted with a new interpolating spline, which gives the mapping function for each scanline (figure 7). For each pixel of each horizontal scanline of the intermediate image, it is now easy to use this mapping function to determine which pixels of the input image influence this output pixel.

The vertical pass uses the intermediate mesh and the intermediate image as input to produce the final output (the source mesh and the source image are not used). It is completely analogous to the first pass, with the rows of I and D fitted by horizontal splines, and the mapping functions computed for each vertical scanline. Note that the intermediate image may be distorted to such an extent that there is not enough information to compute a correct vertical pass – this is a problem shared by all two pass algorithms, called *bottleneck*. This problem will not usually happen, unless there are very large rotational components, and can be avoided by the use of a single pass algorithm.

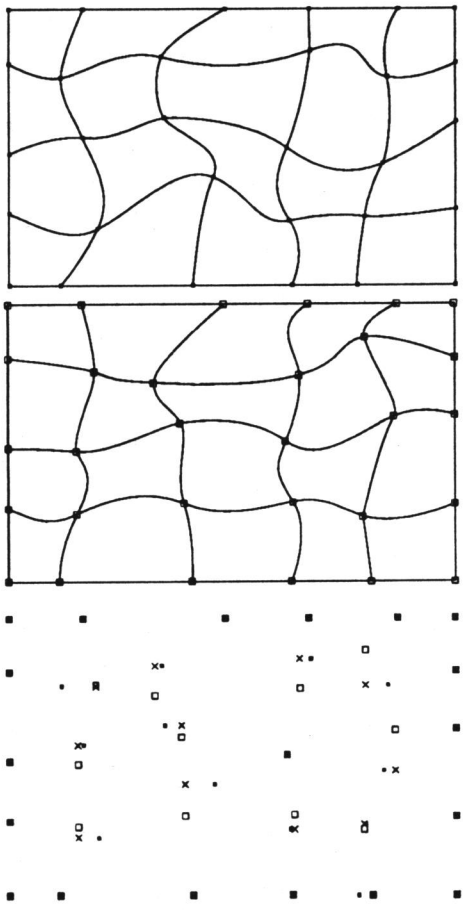


Figure 5: *Spline meshes: top and middle are the source and destination control points shown after fitting the horizontal and vertical splines; at the bottom, the intermediate mesh points are shown as crosses, with source points as filled squares and destination points as empty squares.*

Although this technique proves to be quite versatile and efficient, the user still has to enter too much information while creating the meshes. This is specially true over less important areas of the image, where a precise control is not needed and some kind of automatic process could be used. Also, sometimes the user wants a higher density of points over an important feature to achieve an accurate description of the deformation, but since the mesh resolution is homogeneous, it is necessary to increase the density globally.

Field Warping

The field warping technique allows one to construct somehow a local coordinate change, solving thus one

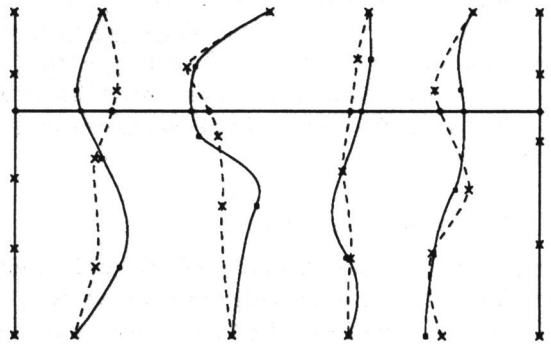


Figure 6: *Columns of the source and intermediate meshes, intersected with a sample scanline.*

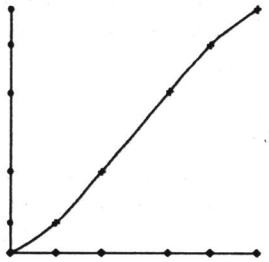


Figure 7: *Sample scanline mapping function. Intermediate mesh intersections are over x-axis and source mesh intersections over y.*

of the major problems of the spline mesh warping. The way this method works is very simple: some distinguished features are marked in the image that should be transformed. The transformation is extended to the neighborhood of the features automatically, through the use of "fields of influence" around them. Different fields of influence from different features of the image are averaged to get a global warping. Recently, an implementation of a field warping technique has been uncovered (Beier and Neely, 1992), where the allowed features are oriented line segments.

Two sets of segments are used to identify the features of the image and show their deformed state. Each line defines a *uv* coordinate system (figure 8): u is the position along the line, and v is the orthogonal distance to the line. Beier and Neely note that the use of u as a fraction of the segment length, and v as an absolute value, is more useful than making both relative to the segment length.

The use of more than one line segment produces conflicting coordinates systems that must be combined in a smooth and controlled manner. Beier and Neely propose the assignment of a weight to each line, that is strongest for points on the line, and decreases the further the points are from it. Each output pixel is inversely mapped into the source coordinate system of each line, producing a set of distinct mapped points. The displacements from the output pixel to each of these mapped points are used in a weighted average to compute the final displacement, using the following weight:

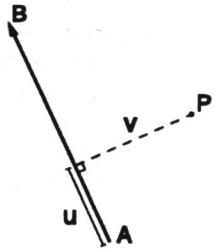


Figure 8: *uv coordinate system of an oriented line segment.*

$$weight = \left(\frac{l^p}{a + \mathrm{d}}\right)^b$$

where l is the length of a line, d is the distance from the output pixel to a line and a, b, and p are constants used to fine tune the warping.

Note that as the mapping of each output pixel depends on all line segments, the addition of a single line influences the whole deformation. Also, the weight calculation that must be repeated for each pixel, relative to each one of the lines, is quite expensive. As the interpolation is mostly automatic, sometimes the algorithm generates unpredictable results that can be avoided by tricky manipulations of the line segments. On the other hand, as this algorithm is executed in one single pass, there is no bottleneck problem.

Field Controlled Spline Mesh Warping

We propose a warping technique that combines the efficiency of the spline mesh algorithm with the local nature of the field warping. After specifying the features of the image, we use them as constraints to deform the spline coordinate curves. More precisely, the spline control points are moved according to the field of influence of each feature. The spline local control, associated with the local nature of the fields of influence results in an algorithm that exploits most of the interesting aspects of both techniques.

A disadvantage of this approach is that many of the problems of both algorithms are still present,

namely bottleneck and foldover. However, the unpredictable effects of field warping can be detected before rendering the warping by inspecting the automatically generated mesh for any apparent folding. Also, it is possible to control the precision of this approximation by increasing the mesh resolution globally and concentrating control points in the interesting areas.

6 Implementation

In this section, we will make general remarks about implementation issues of a morphing system and give some results of a spline mesh morphing system implemented.

The system was implemented in a Amiga 3000 computer, featuring animated sequences morphing, warping and dissolving, with local and global rate control of both shape and color. The Catmull-Rom spline formulation (Foley et alli, 1990) was used to achieve smooth warpings.

6.1 Techniques and User Specification

From an implementation point of view, the user interface takes most of the time; from an application point of view it is the design of the morphing action, and not the computational time, that consumes most of the production time. For this reason, the interface that allows the warping specification by the user is a difficult and very important part of the implementation.

Each of the algorithms described in section 5.2 is more or less adequate to implement each of the warping specifications studied in section 5.1. The best algorithm should be chosen associated with the desired specification. Therefore, the specification is the driving force of the implementation task. Spline mesh warping is an adequate technique to implement mesh warping specification; field warping should be the correct choice to implement feature based specification; a point based warping specification calls for triangle mesh warping with automatic Delaunay triangulation. Mixed warping techniques as the one proposed in this paper are very attractive, because they enable the construction of different user specifications. We believe that this approach is significantly faster than field warping in the rendering time, and than spline mesh warping in terms of the interactive design time.

6.2 Results

Some of the results of the system are shown in figure 9. Figures 9(a) and 9(b) show two digitized images, in their original, undeformed states. Figure 9(c) shows an instance of the progress of the deformation of figure 9(a) into the shape of 9(b), while 9(d) shows the deformation of 9(b) into 9(a). The second phase of the morphing is shown in figure 9(e), an adaptive cross-dissolve between figures 9(c) and 9(d).

7 Conclusions and Future

Morphing is an active field of research, and recent efforts are aimed at improving both the warping map and the user interface. In this respect, we should mention the use of adaptive mesh techniques and a partial automation of the user interface, with automatic feature detection and association.

We are currently working on an implementation of the field controlled spline mesh morphing technique as described in this work, to compare its ease of use and the final results with those of spline mesh and field morphing.

8 References

Barr, A. (1984). *Global and local deformations of solid primitives.* Computer Graphics, 18, pages 21-29, 1984. Proceedings of SIGGRAPH '84.

Beier, T. and Neely, S. (1992). *Feature-Based Image Metamorphosis.* Computer Graphics, 26, 2, pages 35-42, Proceedings of SIGGRAPH '92.

Figueiredo, L. and Carvalho, P. (1991). *Introduction to Computational Geometry.* XVIII Colóquio Brasileiro de Matemática.

Gomes, J. and Velho, L. (1992). *Digital Image.* Revista da Sociedade de Engenharia de Televisão, Março de 1992. (in portuguese).

Foley, J.; Dam, A.; Feiner, S. and Hughes, J. (1990). *Computer Graphics: Principles and Practice, Second Edition.* Addison-Wesley.

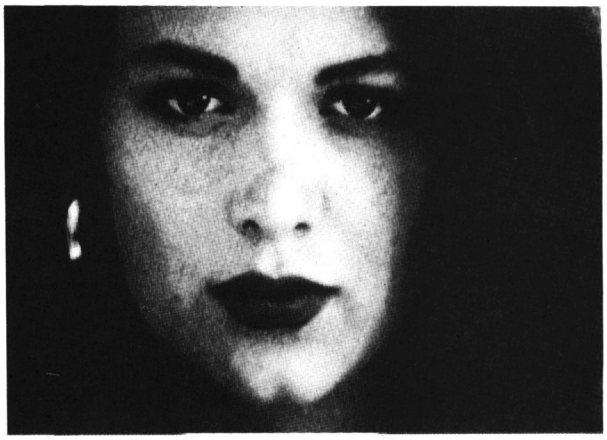
Perlin, K. (1992). *Interactive Image Warping Using Delaunay Triangulation.* Preprint. New York University.

Preparata, F. and Shamos, M. (1985). *Computational Geometry: An Introduction.* Springer-Verlag, New York.

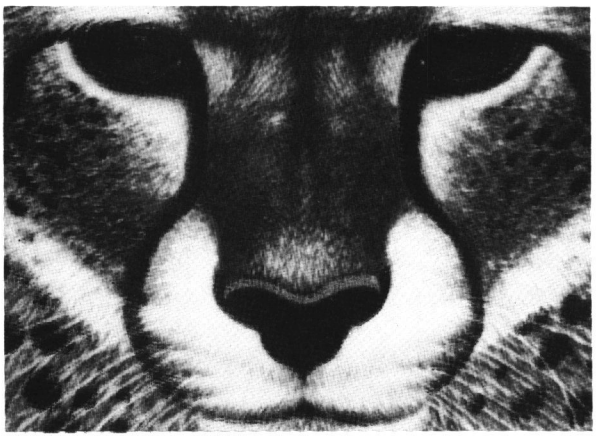
Rosenfeld, A. and Kak, A. C. (1976). *Digital Picture Processing.* Academic Press.

Webster's Encyclopedic Unabridged Dictionary of the English Language. (1989). Portland House.

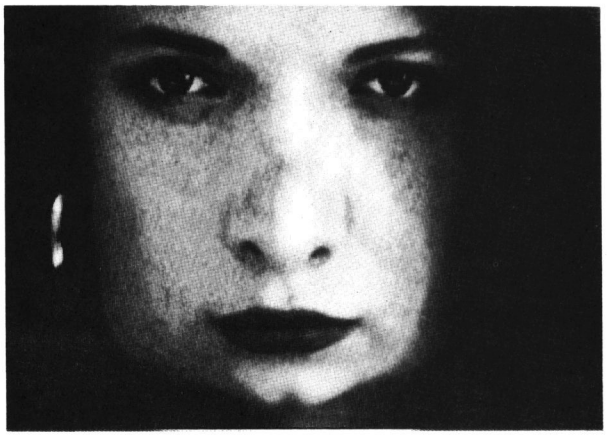
Wolberg, G. (1990). *Digital Image Warping.* IEEE Computer Society Press.



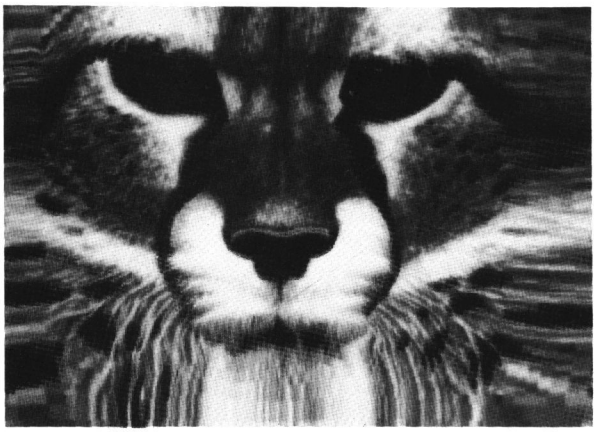
(a)



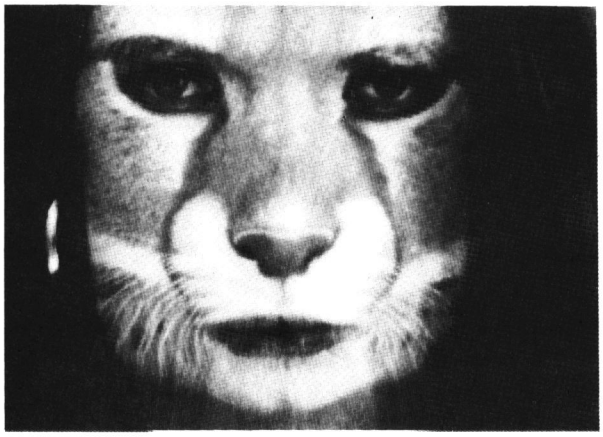
(b)



(c)



(d)



(e)

Figure 9: Morphing steps: (a) and (b) are the original images, in their undeformed states; (c) is a 33% deformation of (a) towards the shape of (b); (d) is a 67% deformation of (b) into (a). Note the alignment of the important features of each image in (c) and (d), like the eyes, nose and mouth. The final morphed image (a blending of (c) and (d)) is shown in (e).