1. INTRODUCTION

Right now, the trendiest technique in computer graphics is morphing. Examples of it are showing up all over the place. Morphing has quickly become a standard tool in all animation laboratories, with clients expecting animators to be able to morph on demand. (Not always a pretty sight). Unlike other terms used in computing and graphics, most lay-persons have heard of "morphing". Even if they don't know of the term, given a quick description and a few examples, it will soon be obvious that they know what you're talking about.

The most common questions asked about morphing are:

What is "morphing"?
How is it done?
What do you need to know to do it yourself?
Where can I see more examples?
Who is doing it?
Where can I get some code?
Where can I read more about it?

This paper sets out to answer these questions by giving basic descriptions and by guiding the reader towards further information. Literature on morphing is difficult to find. The term "morphing" is new, but the process has been around for at least ten years. To find background information, it is best to read up on related areas such as digital image warping and texture mapping.

A large part of the information in this paper is based on articles posted to Usenet along with personal email correspondence. This was a result of the difficulty there is in finding information on morphing. There has been a lot of discussion about morphing through Usenet over the last few months and it can not be overlooked as a source of information about computer graphics.

2. WHAT IS MORPHING?

Finding an exact definition of morphing isn't as easy as you'd expect. There is ambiguity as to the meaning of the term. The main problem is that different people include different techniques under the umbrella of morphing. To omit any of the possible areas from an introductory paper on morphing would be wrong, therefore I will use a very broad definition.

The term morphing originated at Industrial Light and Magic (ILM) (Dovey,1992). The source of the term may have been a program, "morf", which ILM used to interpolate between two images (Anderson,1990). Morf was originally developed for the transformation sequence in Willow (see Fig. 2.1) and has been used on many projects since then. Morphing is derived from the Greek word morphe which means form or shape. The study of shapes is known as morphology and morphing has come to mean shape-changing. In particular, changing the shapes of objects using computers. Morphing can be done in 2-D and in 3-D. 2-D morphing gives the visual effect of a 3-D change of shape by warping and blending images. In 3-D morphing the 3-D model of the object is transformed from one shape into another.

Primitive versions of morphing were done in the past by cross-dissolving images. The most memorable of these types of image changes were in horror movies like "The Wolf Man" in 1941 (Sorensen, 1992). As technology improved, so did special effects. The advent of digital image processing gave way to more realistic images from computer animation. Sophisticated techniques are a necessity when trying to convince the human eye that something impossible is really happening. Parke (1982) notes that as computer animation gets closer to reality, the audience tends to become more critical of its quality.

There are many companies working on soon to be released software that is capable of doing morphing. Quite a few graphics packages already offer such facilities. In some cases it can be hard to get information about the methods being used by different animators as their software is proprietary.

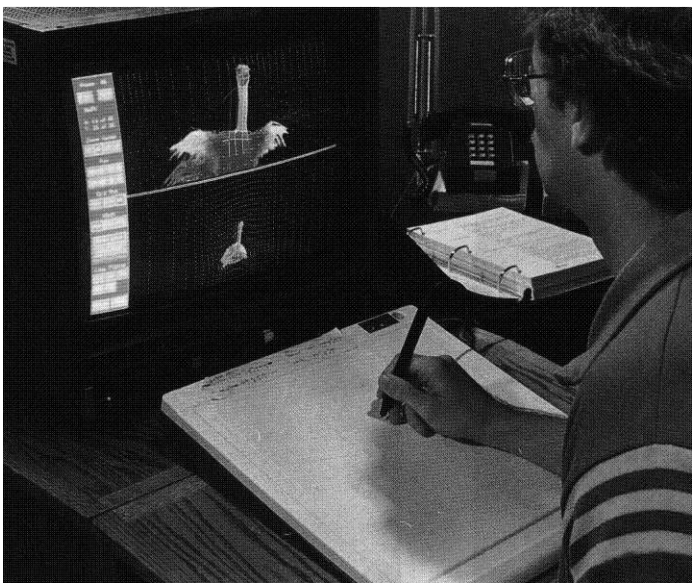


Fig. 2.1 : Morphing process for Willow, Wolberg (1990)

3. TYPES OF MORPHING

Morphing techniques can be split into two main areas: 2-D and 3-D morphing. 2-D morphing is an application of digital image warping. It involves stretching and deforming an initial image to the shape of the final image. At the same time, the textures for each image need to gradually change from the initial to the final texture. For greater control, the images are broken up into small regions that are mapped onto each other for the morph. For 3-D morphing, a correspondence has to be set up between parts of the initial and final 3-D shapes. The difficulty in 3-D morphing is in morphs of objects that are structurally different. For example, to morph a torus (donut shape) to a rectangle poses problems with the donut's hole and the rectangle's edges. Descriptions of different approaches to 2-D and 3-D morphing follow.

3.1. 2-D Morphing

As more and more people work on 2-D morphing, the techniques being used are becoming more and more sophisticated. As a result, the quality of the sequences being produced are improving. Most of the techniques for 2-D morphing come from digital image warping; a growing branch of image processing. Digital image warping deals with the geometric transformation of digital images, redefining the spatial relationship between points in an image. One of the most common applications for image warping is texture mapping, a technique that is central to morphing. To do a morph, the animator needs to define a correspondence between the initial and final images. This can be done using points, triangles or meshes. Once the relationship has been defined, the textures in the images are blended from the initial image to the final image to produce the morphing sequence.

The method of subdividing the images varies between different morphing programs. Michael Kaufman (1992) defines common points in the two images. For a human face, these points could include the eyes, eyebrows, nose and mouth. He then builds triangles from these points to create corresponding areas on the images. From there, the technique is similar to those described below. Triangulations can be done by hand or specialised software can be used. One of the most popular triangulation algorithms is the Delaunay triangulation. Delaunay's algorithm maximises the angles within each triangle and minimises the lengths of the sides. A more detailed description of Delaunay's triangulation is in the Helpful Information section.

The morphing programs written by Mark Hall (1992) and Jay Windley (1992) start with the user specifying the triangles to be morphed in each image. Hall warps the triangular mesh of the input image to match the mesh of the output image. As the animation goes from the initial to the final image, the colours change from that of the first image to that of the second. Windley does a linear blend of the contents of the triangles using barycentric geometry.

Morphing isn't always done with triangular patches. Kermit Woodall (1992) is working on morphing software for the Amiga. Their system, "Montage", uses cubic splines in a deformable mesh to define the input and output images. Cubic splines are better for defining curves in the image. Triangular patches have to be small and well chosen to give a similarly smooth result. On the positive side for triangular meshes is the fact that they are simpler to work with. The choice will often depend on the platform the morphing system will be running on. Some systems provide facilities for working with triangles, and some have facilities for cubic splines. A quick check of what is available on your system can make life a lot easier when writing a morphing system.

2-D morphing is a very manual process. It relies heavily on the experience and know-how of the animator. The choice of subject matter can make all the difference. Care should also be taken to use images of the subject that have been taken from close to the same angle. Similarly, the position of parts of the subject should be closely matched. To illustrate this, picture a morph between two people. Their position within the scene, as well as their stance, should match closely.

3.2. 3-D Morphing

When doing a 3-D morph, the animator has to do a metamorphosis between 3-D objects before creating a 2-D image of the scene. 3-D morphing alters the surface description of the object, as opposed to 2-D morphing which interpolates texture maps of the 2-D images. The objects involved can be very similar in type, for example people's faces - all faces have the same components; or they can be as geometrically different as a cube and a sphere. In cases of similar topology (connectivity of points), the morphing will be a point to point mapping of some type of mesh. Finding corresponding points is a lot more difficult when the topology is different in each object. The methods used for 3-D morphing depend on the type of objects the animator is working with. Convex shapes are the easiest. They have no inward curves on their surface. A sphere is convex, but a golf ball is not because of the little dimples on the golf ball's surface. Star-shaped objects would include the golf ball. A star-shaped object is an object for which there exists an interior point that can reach all the points on the surface, without intersecting any edges. A torus is not a star-shaped object as there are some parts of the shape that would be hidden from any interior point that could be defined. (see Fig. 3.1) Some 3-D morphing methods are described below.

The simplest 3-D morphs are done when the initial and final shapes have similar topologies. For example, regions of one facial model can be mapped onto the same region on another face. (see Fig. 3.2) In their paper on human prototyping, Magnenat-Thalmann et. al. (1988) use morphing to create new synthetic actors. They reorganise the input faces by creating a new structure of facets. These facets need to line up with reference points on each face. Magnenat-Thalmann et. al. define regions on each face to imply their correspondence and then the insides of the regions are divided up using an automated algorithm. Once the correspondence between the faces is defined, a morph can be done using linear interpolation. The results of their techniques are shown in a morph from Marilyn to Humphrey Bogart. (see Fig. 3.3) (see Fig. 3.4)

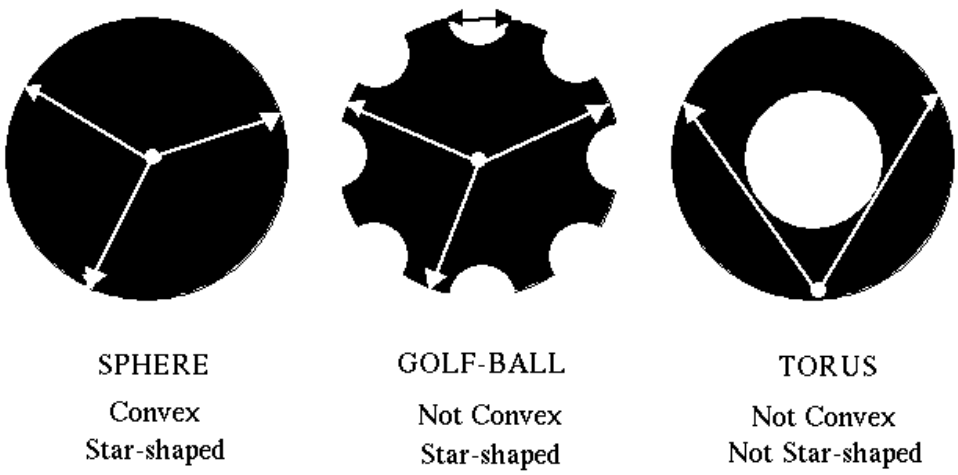


Fig. 3.1 : Convex and star-shaped objects.

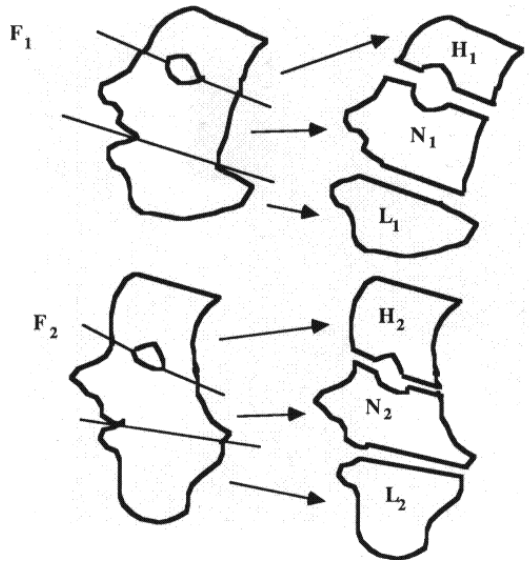


Fig. 3.2 : Facial Regions, Magnenat-Thalmann (1988)

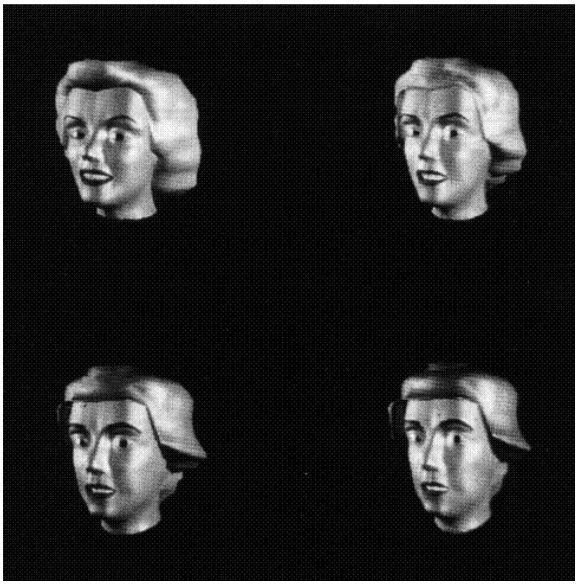


Fig. 3.3 : Marilyn to Bogart (early stages), Magnenat-Thalmann (1988)

Fig. 3.4 : Marilyn to Bogart (later stages), Magnenat-Thalmann (1988)

In their 1989 article on shape distortion, Wes Bethel and Sam Uselton outlined their 3-D morphing system. Their system performed a warp between two objects. The objects had to be polyhedra, but the principle behind the system could work on b-spline surfaces as well. The first step was to construct a B* tree for each object. The nodes of the tree represent faces and its branches represent adjacency relationships between faces. The user then selects one face and one vertex from that face, for each object, to set up the correspondence between the objects.

Once the trees and the correspondences have been defined for the objects, a union of their topologies has to be created. A new B* tree is created to hold the information about the union. The goal is to preserve the adjacency relationships among the faces. If one of the objects has more faces than the other, then an extra face is added to the union tree. Missing faces and vertices are added as needed. If one of the objects has a hole in it, then the union will also have a hole. The result is a master object that has all the topological characteristics of each of the input objects.

The most difficult part of the processing is assigning new coordinate values to the master object. These coordinates need to be defined for the initial and final keyframes. Convex and star-shaped objects are fairly straight forward, but objects with holes in them are more difficult. If there is a hole in the initial frame and not in the final, then the hole should not be visible in the initial frame. The hole is really there, but it has been geometrically collapsed to hide it. The resultant system can semi-automatically match the components of two objects. Using a simple interpolation scheme, the user is able to do keyframe animation of 3-D scenes of shape-changing objects.

In their paper on topological merging, James Kent et. al. (1992) present a technique for computing smooth transformations between two polyhedral models. By taking topological and geometric information into account, their system gives results that maintain their connectivity at intermediate steps of the transformation, a desirable quality in this type of system. Their system displays far less distortion than that obtained by previous shape-transformation methods. The algorithm, when given two 3-D models, generates two new models that have the same shapes as the original ones. The new models share the same topology, which is a merger of the original objects' topologies. The merger allows the transformations from one model to the other to be easily computed.

Kent's system is currently restricted to star-shaped models with no holes. This restriction is only temporary as the concepts involved are applicable for arbitrary polyhedral models.

There are many things that need to be thought of when developing a shape transforming system. Kent (1992) evaluates previous transformation systems using the following criteria:

1) Is the face connectivity maintained for all intermediate shapes?
2) Are the intermediate shapes overly distorted?
3) What restrictions on the original models exist?

Few systems pass this evaluation. Kent also outlines two problems that may arise during interpolation. The first is that when there are faces that have more than three sides, they may not stay planar during transformation. Depending on the rendering system, this may or may not be a problem. If it is a problem, then the objects should be triangulated before transformation. The second problem is that the object may intersect itself during the interpolation. This usually happens if the shapes are extremely concave. No solution to this second problem has been found.

Kent's system establishes a correspondence between the objects in three stages. First, each model is projected onto the surface of a unit sphere. Then the union of the two model's projections is taken using a modified version of the Weiler (1980) polygon clipping algorithm. The merged topology is then projected back onto the original models. The results of Kent's approach are highly encouraging (see Fig. 3.5) working with convex and star-shaped objects. Future work on their project will look at developing projections for general concave polyhedra and polyhedra with holes. They are also looking at giving the animator more control over the transformations. (see Fig. 3.6) By using both topological and geometric information in their system, Kent et. al. maintain the integrity of the objects (in a Euler sense) while the output is intuitive in appearance. There is much less distortion of the shapes in this method.

Dave Bock (1992) takes a different approach to morphing by transforming objects using their mathematical descriptions. In his technique, Bock generates volume data files that contain varying percentages of the two objects' data sets. The initial and final frames have 100% of one object and 0% of the other. He interpolates both both volume data sets to define their respective objects at a common data threshold to use with the isosurface generator. Then a series of data files is created to hold the combined percentages of the initial data files at different stages of the morph. The number of data files depends on the number of inbetween frames the user specifies. For each frame, volume data files are created containing increasing percentages of the final image and decreasing percentages of the initial object. Then the isosurface generator is used to create the geometric object for each frame, based on the volume data in the files. Bock states that he has "obtained some very interesting and

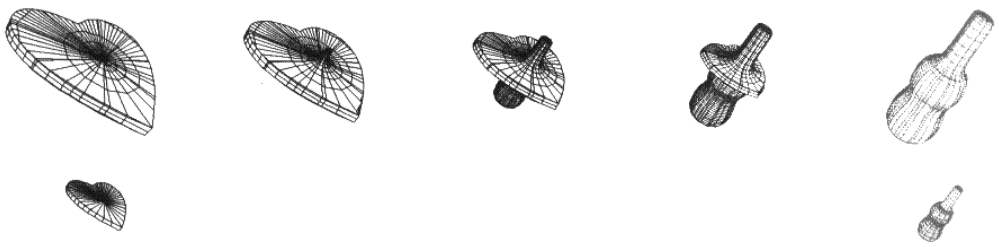


Fig. 3.5 : 3-D transformation, Kent et. al. (1992)

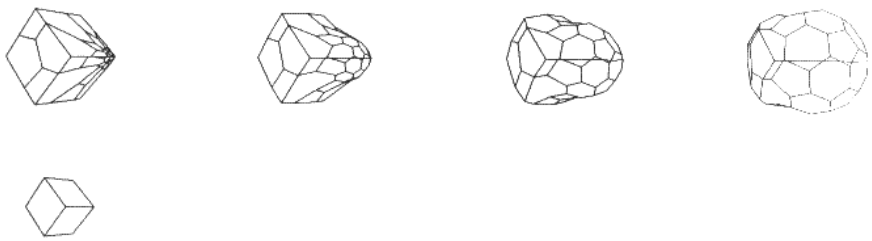


Fig. 3.6 : User control of morphing process, Kent et. al. (1992)

exciting results" with his morphing technique. He is working on making the system accept geometrically defined objects as well as mathematically defined ones.

In all of the 3-D morphing systems above, the emphasis has been on getting the 3-D models defined. Any rendering and texture mapping is done later. This is one of the main differences between 2-D and 3-D morphing. Another difference is that 3-D morphing systems tend to be more automated than their 2-D counterparts.

3.3. WAYS TO IMPROVE RESULTS

Morphing software does not produce brilliant results on its own. The animator has to work hard to make the morph smooth and believable. Michael Kaufman (1992) noted that the results can be improved by defining more points on the initial and final images. The use of a smaller mesh gives a higher resolution to the morph so it can stand close scrutiny. This is particularly important if the sequence is to be shown on the large screen. Film audiences have become very spoilt with the quality of graphics they see, and expect everything to be at least as good as what they've already seen. The number of people who have gone through Terminator 2 laser disks frame by frame illustrates how picky audiences can be. (see Fig. 3.7)

Key areas on the images need to be separated to stop them from dissolving during morphing. Animators need to pay particular attention to features such as the eyes and mouth on a face. The human eye is drawn to the eyes and mouth more than any other feature (see Fig. 3.8) according to Desmond Morris (1982). Therefore a good strategy is to try to choose images that are fairly closely matched on key features. If the features don't match up, the images can be warped before use to try to line up reference points. Benson (1992) states that, even with something as familiar as a human face, observers will rarely notice a well executed warp. Humans remember the shapes of certain features rather than an exact image of the whole face.

A good idea for a more deceptive morphing sequence is staggering the morphs as the image changes. That is, to morph different parts of the image at different times. A good example of this is Michael Jackson's "Black or White" film-clip. As Juhana Kouhia (1992) pointed out, in part of the dance sequence, as the faces changed to the different dancers, each frame would show some of the features as the initial image while others were the final image. This distracts the viewer's eye so that they don't know where the next feature change will take place. Donald Dovey (1992) states that in some parts of "Black or White" there were up to seven planes of morphing going on independently.

Fig. 3.7 : The T-1000 morphs out of Sarah Connor, Sorenson (1992)

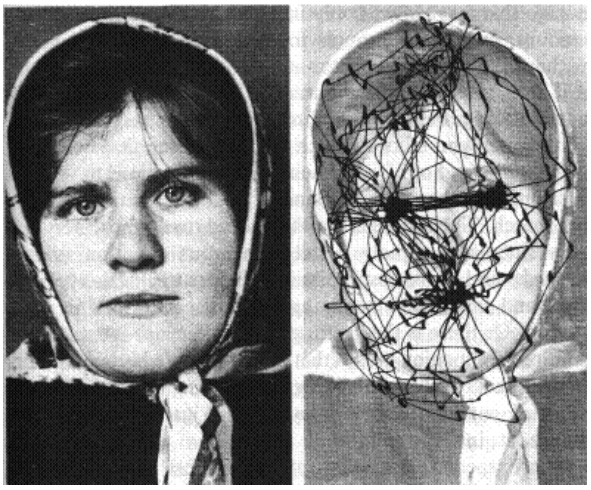


Fig. 3.8 : Tracking the eye movement when looking at a face, Morris (1982)

As in all animation, there is no law against doing touch-ups after the computer animation part has been done. Cleaning up rough edges to give an improved finish is very common. The need for touching-up can be minimised by careful planning before animation begins. Patience on the part of the animator is a necessity.

4. HELPFUL INFORMATION

Before creating a morphing system, it is good to get a bit of background knowledge of the area. For 3-D morphing, it is necessary to know how to store information in an efficient manner. The section on B*-trees covers one method of storing geometric information. This method was used by Wes Bethel and Sam Uselton (1989). References for further reading are included.

For 2-D, and sometimes 3-D, morphing, a reliable method for triangulation can be useful. A description of the Delaunay triangulation is given below, along with references to papers that include algorithms for the Delaunay triangulation. Another handy concept for image warping is the Fast Fourier Transform. I give a quick overview of the transform and how it is used below.

Before the sections on B*-trees and triangulation comes an introduction to digital image warping. Digital image warping is an area within image processing which provides the techniques needed to do texture mapping and 2-D morphing. Having some knowledge of the ideas behind image processing can help when creating a morphing system. A lot of the concepts within digital image warping have become utilities that the programmer can include in their system. This can make programming such a system much easier as thorough knowledge will not be so important. Similarly, utilities for working with triangles and search trees may already exist on the platform the morphing system is planned for. The programmer should make themselves aware of what is available before re-inventing the wheel.

4.1. Digital Image Warping

When doing a 2-D morph, the animator is mapping a texture onto an arbitrary shape. This mapping can be referred to as texture mapping or warping a digital image. Digital image warping refers to the geometric transformation of digital images. There are a lot of applications for image warping techniques. In computer graphics, when we synthesise an image, we can map a 2-D texture image onto a 3-D surface and then project the scene to create a 2-D image (Heckbert, 1991). The net effect of this is a warp from a 2-D texture image to a 2-D screen image. The following two sections give an overview of texture mapping and a quick summary of a book, "Digital Image Warping", by George Wolberg.

4.1.1. Texture Mapping

Texture mapping is a technique that can enhance the visual quality and realism of an image for a relatively small cost. Early computer graphics was criticised for its lack of realism, its lack of texture. In recent years, methods of texture mapping have improved to the point that computer generated images are often able to pass for photographs.

A texture in computer graphics is an image. This image can be seen as a function of N dimensions, given an N-dimensional image. 2-D textures are the most common, but 3-D textures do exist and are used for solid modelling. A texture is mapped onto a surface in 3-D object space which is, in turn, mapped onto the image plane by the viewing projection (Heckbert, 1986). Bier (1986) uses the analogy of clothing to illustrate texture mapping. With clothing, you start with an object (body) and then wrap an image (article of clothing) around it. This is one way of visualising image warping.

The mapping from texture space to screen space occurs in two phases:

1) Texture space to object space - using a surface parameterisation,
2) Then object space to screen space - using a viewing transformation.

The intermediate step, object space, is often forgotten. Without that stage, texture mapping becomes image

warping (Heckbert, 1986). There are three general approaches to texture mapping: scanning in screen space, scanning in texture space and two-pass methods. Screen order is the most common. For each screen pixel, the corresponding value in the texture is calculated. This method is best when the screen must be written sequentially, when the mapping is readily invertible and the texture is random access. Texture order can lead to problems with holes and overlaps in the screen space, it is best when their are restrictions with access to the texture. Two-pass methods decompose the 2-D mapping into two 1-D mappings. To map a texture onto a surface, the 3-D surface needs to be parameterised.

Different projections can be applied when doing a texture map. The common projections are the orthographic and perspective projections. A matrix calculation has to be done for each pixel in the image so there is a lot of interest in finding faster methods for evaluating the equations. Texture mapping onto surfaces made up of patches is common as the patches are already parameterised and the extra computation required is small compared to the cost of patch rendering.

Once the warping of the image has been done the image is then sampled to match the screen grid. The easiest way to do this is point sampling - taking the pixel closest to the one that is wanted. The problem with point sampling is that it may not be representative of the true colour of the image in that area. Some points of colour could be missed out completely. This is a case of "aliasing". An image can be treated as a continuous signal which is sampled at each pixel. The sampling theorem states that the sampled picture cannot represent spatial frequencies greater than 1 cycle/2 pixels. Aliasing refers to the result of sampling a signal containing frequencies higher than this limit. Aliasing gives a "steppy" look to the picture as high frequencies reappear under the alias of low frequencies (Blinn, 1976). Crow (1981) gives a comparison of techniques for reducing aliasing. Two approaches are:

1) Point sampling at higher resolution.
2) Low-pass filtering before sampling.

The first method uses information about the image to decide the sampling rate required. The second method is preferable as it addresses the causes of aliasing rather than the symptoms. The filter To eliminate aliasing, the signal is band-limitede by the filter. For affine warps (linear) the filter shape remains constant as it moves across the image. It is a space-invariant filter. Space invariant filtering is often using a fast Fourier transform (FFT), a multiply and an inverse FFT. Non-linear mapping requires space-variant filters. Space-variant filters are more complex and less-well understood than space-invariant filters. The cross-sectional shape of the filter is important. Filters include the box, the triangle, the cubic B-spline and the truncated Gaussian. (see Fig. 4.1) Methods of random access space-variant filtering fall into three categories: direct convolution; prefiltering and Fourier series. The best filter for a task depends on the texture representation in use (Heckbert, 1986).

"Despite the recent explosion of diverse applications for texture mapping, a common set of fundamental concepts and algorithms is emerging." (Heckbert, 1986). There is a lot of literature around covering texture mapping. Finding out more about the topic is not too difficult. My coverage is intended serve as a primer for reading about texture mapping. Most of this information has come from Heckbert (1986). Other articles used include: Bier(1986), Wolberg (1988, 1990), Fiume (1987), Frederick (1990) and Blinn (1976).

4.2. Digital Image Warping - George Wolberg

George Wolberg's book has met with near unanimous acclaim. It has "filled a gap in the literature" (Heckbert, 1991). Wolberg intended the book to be a practical guide for scientists and engineers. It gives a basis for doing work in the image warping area, as well as being a good introductory text for those who want to go further in the field. The only person who seems to have a bad word to say about the book is Paul Heckbert (1991). He gave it a scathing review in Computer Graphics and Applications' January 1991 issue. The letters to the editor in the May 1991 issue are replies to his review by Wolberg and another supporter of the book. Take note that Paul

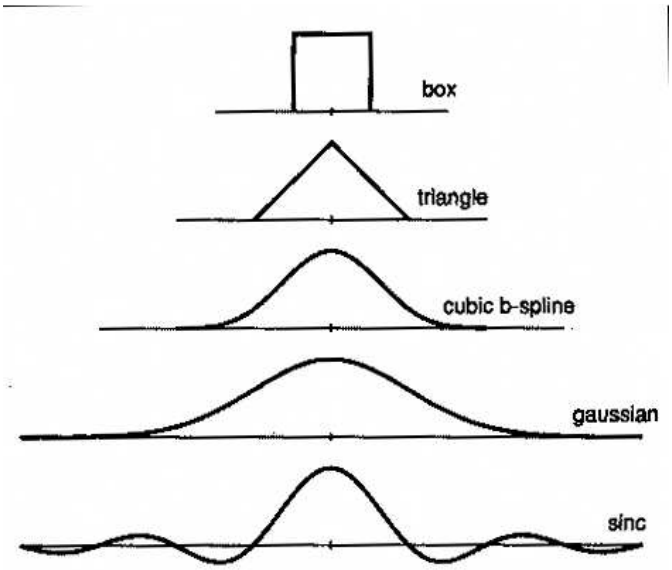


Fig. 4.1 : Filters, Heckbert (1986)

Heckbert is a prolific publisher of papers on texture mapping. This gives his opinions more weight, but it must be remembered that authors may see other authors as competition.It is worthwhile to read both sides of the discussion and then judge the book for yourself.

The sections that follow are meant as a quick overview of Wolberg's book. In my opinion, the book is quite easy to follow, is well supported by its reference section and has generous amounts of code to go with the concepts it examines.

4.2.1. Chapter 1 - Introduction

The first chapter serves as an introduction to the area of digital image warping. Digital image warping is defined to be "a growing branch of image processing that deals with the geometric transformation of digital images." Applications for warping include: remote sensing, medical imaging, computer vision and computer graphics. The earliest work in image warping stemmed from remote sensing, where they tried to reduce distortion. In computer graphics, the aim is to induce geometric distortion. The primary application in graphics is texture mapping.

Image warping is seen as an image processing task as it takes an input image and applies a geometric transformation to produce an output image. This book centres on the three components that comprise all geometric transformations in image warping: spatial transformations, resampling and antialiasing.

The stages involved in a geometric transformation are:

1) The image is aquired by the image aquisition system.
2) The image passes through the image resampling stage, consisting of reconstruction and sampling.
3) The output image is obtained once the resampling has finished.

Spatial transformations, antialiasing filtering and sampling theory are all optional in the image resampling stage.

4.2.2. Chapter 2 - Preliminaries

Wolberg's section on preliminaries covers basic terminology and mathematical primers. The first section covers the fundamentals of images and signals: a signal is a function that conveys information and an image can be represented as a 2-D signal. This 2-D function can be monochrome or colour. A filter is any system that processes an input signal to produce an output signal. If it doesn't change across the image, the filter is said to be spatially invariant. When an impulse is applied to a filter, an altered impulse, referred to as the impulse response, is generated at the output. Imaging systems have limited accuracy, so the input impulse becomes blurred as it is displayed. The blurring response is known as the Point Spread Function (PSF) of the imaging system. The response of a digital filter to an arbitrary input signal is expressed in terms of the impulse response of the filter using the convolution integral. Wolberg draws an analogy to audio signals to help to explain the preceding definitions and concepts.

Fourier transforms offer a powerful set of analytic tools to analyse and process single and multidimensional signals and systems. Periodic signals have discrete Fourier components and are described by Fourier series. Aperiodic signals have continuously varying Fourier components and are defined by the Fourier integral. The discrete Fourier transform is handy for digital data as the data involved is sampled, not continuous. The fast Fourier transform is a computational algorithm that speeds up the transformation.

Images are collected using a digital image aquisition system. These systems consist of an imaging subsystem, a sampling subsystem and a quantiser. There are three broad categories of imaging systems: electronic (Vidicon), solid-state (CCD Cameras) and mechanical (Flatbed scanners).

Input imagery appears in many different media. Digital aquisition systems convert these input sources into digital form. Imaging systems need digital to analog converters to discretise the data. This involves sampling and quantising the analog signal. The result is a digital image, which is an array of integer intensity values.

4.2.3. Chapter 3 - Spatial Transformations

The basis of geometric transformations is the mapping of one coordinate system onto another. This is defined by means of a spatial transformation - a mapping function that establishes a spatial correspondence between all the points in the input and output images. Simple transformations may be specified by analytic expressions including: affine, projective, bilinear and polynomial transformations. More complex mappings can be determined from a sparse lattice of control points for which spatial correspondence is known. A forward mapping copies each input pixel onto the output image at positions determined by the mapping functions. An inverse mapping works in screen order, projecting each output coordinate onto the input via inverse mapping functions. Unlike forward mapping, inverse mapping guarantees that all output pixels are computed.

There are standard matrices for doing transformations. The general 3x3 transformation matrix can handle scaling, shearing, rotation, reflection, translation and perspective in 2-D. Wolberg gives examples of each of these transformations, grouped into affine and non-affine transformations. He then covers polynomial transformations and how to compute their coefficients. All of these transformations are global, Wolberg goes on to cover piecewise polynomial transformations which are non-global. Piecewise transformations use only the nearby control points when computing the local transformations. One method of subdividing data is triangulation. Wolberg then discusses the theory behind optimal triangulation.

A more general framework for inferring a mapping function for scattered points is global splines. Global splines are defined through basis functions. They are one of the oldest interpolation techniques. Wolberg goes through different methods of using global splines, and relates them back to earlier techniques.

Surface interpolation requires a lot of information that is often difficult to quantify. The choice of interpolation method depends on the smoothness that the application requires. Fortunately, this does not required complex mathematic analysis as it can be assessed visually by the user.

4.2.4. Chapter 4 - Sampling Theory

Sometimes undesirable artifacts can arise due to the discrete nature of digital images. Sampling theory provides an elegant mathematical formulation describing the relationship between a continuous signal and its samples. It is used to resolve the problems of image reconstruction and aliasing. Reconstruction is an interpolation procedure applied to the sampled data, whereas aliasing refers to the presence of unreproducably high frequencies and the resulting artifacts.

A continuous signal may be reconstructed from its samples if the signal is bandlimited and the sampling frequency is higher than the Nyquist rate. Ideal low-pass filtering is one method of retaining the original spectrum while discarding the copies. Unfortunately, using a low-pass filter in the spatial domain requires an infinitely wide sinc function. Therefore, approximations to the ideal filter are used. These non-ideal filters introduce blurring of the spectrum and artifacts in the image, for example, jagged edges. Aliasing occurs when the signal is undersampled. To combat aliasing, bandlimiting and higher sampling rates must be used.

4.2.5. Chapter 5 - Image Resampling

In digital images, the pixels are restricted to lying on the sampling grid, usually taken to be the integer lattice. The output pixels are passed through a mapping function to generate another grid to resample the input image. The new sampling grid rarely coincides with the integer lattice. Instead, the positions of the grid points can take on any of the continuous values assigned by the mapping function. Since the input is only defined at integer positions, interpolating is needed to fit a continuous surface through the data samples. This surface can then be sampled at arbitrary positions. This interpolation stage is known as image reconstruction. When image reconstruction is followed by image sampling, the whole process is known as image resampling.

The accuracy of the interpolation has a significant effect on the quality of the output image. There is the usual trade-off between computational efficiency and the quality of the approximation. Methods of interpolation include: bilinear, nearest neighbour and cubic convolution. As methods become more accurate, they also become more expensive. The ideal filter is convolution with a sinc function. The problem with this filter is that it requires an infinite number of neighbouring elements. Therefore work is being done to make reasonable approximations to this filter. Some researchers are taking a different approach to filter creation by using analytic functions with a free parameter to tune the transition width. Code for implementing resampling algorithms is given, along with full explanations of the theory behind them.

4.2.6. Chapter 6 - Antialiasing

The first problem with working in the discrete domain is sampling discrete input. Another problem arises when evaluating the discrete output. The output image can be created using point sampling. With point sampling the value of each sampled point is taken independently of its neighbours. This method discards all the information between the sampled points. If these skipped intervals are complex, interpolation of the data will be

impossible and the lost data will be unrecoverable. The input signal is then said to be undersampled, and any attempts at reconstruction gives rise to aliasing. The unreproducably high frequencies that can result from undersampling produces aliasing, visible as jagged edges and moire patterns. Appropriate filters must be used to integrate all the information that will be mapped onto one pixel.

The filtering used to counter aliasing is known as antialiasing. Antialiasing typically requires the input to be blurred before sampling. This forces each pixel to be influenced by its neigbours, diminishing the extent of the artifacts. Limitations on output devices prevent sampling at a high enough rate to counteract aliasing.

All contributions in this area fall into one of two categories: direct convolution and prefiltering. Direct convolution calls for increased sampling to accurately resolve the input preimage that maps onto the output pixel. A low-pass filter is applied to these samples, generating a single, bandlimited output value. This approach raises two issues: sampling techniques and efficient convolution. Sampling techniques covered include:

1) Point sampling,
2) Area sampling,
3) Regular sampling (supersampling, adaptive supersampling),
4) Irregular sampling (stochastic sampling, Poisson sampling, jittered sampling, point-diffusion sampling,
" adaptive stochastic sampling).

Reconstruction from these samples is also covered. Efficient convolution is addressed by algorithms which embed the filter kernels in look-up tables and provide fast access to the appropriate weights. Despite all the optimisations, larger preimages still incur excessive sampling and filtering costs.

A cheaper approach is prefiltering. Prefiltering gives lower quality results with a constant number of computations, independent of the preimage area. The method involves precomputing pyramids and summed-area tables and combines their partially filtered results to produce large performance gains. It is early days yet for these methods. Much work is yet to be done to allow arbitrary preimage areas and filter kernels.

4.2.7. Chapter 7 - Scanline Algorithms

Speed is of major importance to all areas computer graphics. There is a constant trade-off between speed and accuracy. One method of increasing the speed of warping is to use scanline algorithms. They work on separable geometric transformations, turning 2-D transformations into a series of 1-D resampling problems. Scanline algorithms have been shown to be applicable for affine and perspective transformations, as well as for mappings onto bilinear, biquadratic, bicubic and superquadric patches.

Simplification of the transformation is the idea behind scanline algorithms. Algorithms covered by Wolberg are:

1) Incremental Algorithms - Quadratic and cubic interpolation. Algorithms and code for methods by:

Braccini and Marino, 1980
Weiman, 1980
Catmull and Smith, 1980
Paeth, 1986 / Tanaka et.al., 1986
Volder, 1959 (The CORDIC algorithm)

2) 2-Pass Algorithms - Descriptions of algorithms by:

Catmull and Smith, 1980
Fraser, Schowengerdt and Briggs, 1985
Smith, 1987

3) 2-Pass Mesh Warping - Used by Industrial Light and Magic and developed by Douglas Smythe. A thorough description, with code, is given.

4) Other Separable Mappings - Short descriptions are given for the following algorithms:

Perspective projection: Robertson, 1987
Warping among arbitrary planar shapes: Wolberg 1988
General 2-pass algorithm: Wolberg and Boult, 1989

5) Separable Image Warping - A full description of Wolberg and Boult's 1989 algorithm.

4.2.8. Chapter 8 - Epilogue

Wolberg gives a quick review of what he has said in each chapter, outlining the history and future of digital image warping. Applications fall into two classes:

Geometric correction - remote sensing, medical imaging and computer vision.
Geometric distortion - computer graphics through texture mapping.

All geometric transformations have three principal components: spatial transformation, image resampling and antialiasing. Wolberg gave a lot of practical information on each of these areas, overviewing different techniques and their respective problems.

4.2.9. Appendices

Appendix 1 - Fast Fourier Transformations

This section gives a detailed review of the fast Fourier transform (FFT). It assumes that the reader has some familiarity with the concepts involved. The review starts with a discussion of the discrete Fourier transform (DFT). It covers methods for evaluating the DFT using the FFT. Algorithms by Danielson-Lanczos, Cooley-Tukey and Cooley-Sande are discussed and derived. C source code for using the FFT is also given.

Appendix 2 - Interpolating Cubic Splines

This appendix reviews the fundamentals of interpolating cubic splines. The section begins with a definition for cubic splines, followed by a description of constraints used to guarantee that the spline goes through the data points. Derivations for the coefficients by first and by second derivatives are defined. The C code for implementing cubic spline interpolation for uniformly and non-uniformly spaced points is given.

Appendix 3 - Forward Difference Method

A brief description of the forward difference method is given. Derivations of the forward differences for quadratic and cubic polynomials are given along with some code for their computation.

4.3. The Fast Fourier Transform

The Fourier transform has long been used for characterising linear systems and for identifying the frequency components making up a continuous waveform. When a continuous waveform is sampled or recorded on a digital computer, a different form of the Fourier transform must be used, the discrete Fourier transform (DFT). The DFT has extra constraints put on it as it must operate on sampled waveforms defined over finite intervals. The fast Fourier transform (FFT) is simply an efficient method for computing the DFT. The equations for the continuous, discrete and fast Fourier transforms can be found in Bergland (1969) or in Wolberg (1990).

The FFT gives computational savings in many applications. Some of these are:

1) Computing a spectrogram (a display of the short-term power spectrum as a function of time).
2) The convolution of two time series to perform digital filtering.
3) The correlation of two time series.

Application (2) is of most use in image processing. In a linear system, two problems will often come up:

1) Given the input and the impulse response, determine the output.
2) Find the impulse response, given the input and output for a system.

Both of these problems can be approached fairly easily in the frequency domain. (see Fig. 4.2)

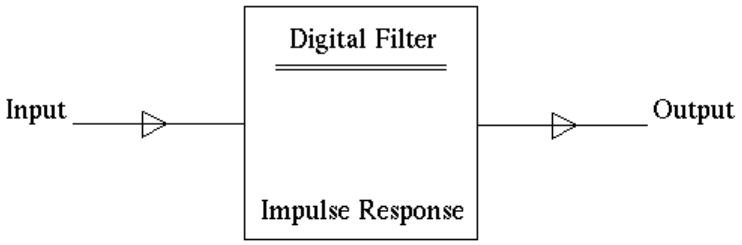


Fig. 4.2 : A linear system with input, impulse response and output.

Before using the FFT, a programmer should know its limitations. Most of the problems that stem from using the DFT and FFT are due to misunderstandings about the effect of approximating a continuous function. Aliasing can occur if the sampling rate for the approximation is too low. It is possible that a function can mimic another function if they happen to coincide at the sample points. The cure for this is to sample the signal at a rate at least twice as high as the highest frequency present. If the signal is known to be within a certain band, the sampling rate can be chosen accordingly. Other problems come with the limited time that the sample is taken in. This sample assumes that nothing interesting is happenning before or after the sample. Whether this assumption is correct or not for a given case can not be assessed. Modifications of DFT's can help to solve these problems (Bergland, 1969).

If further references on Fourier transforms are required, the reference section of Wolberg (1990) and Bergland (1969) should help. Books such as Brigham (1988) are completely devoted to the Fourier transform and its applications are available for those with a burning desire to know all about the transform.

4.4. B*-Trees

B-trees are balanced m-way search trees. Balanced search trees speed up the search procedure by keeping to a minimum the number of nodes on each branch and the depth of the tree. Bayer and McCreight (1972) applied a modification to the B-tree known as the overflow technique. This technique improves the insertion algorithm by using local rotation to cause keys to overflow from a full node into a sibling node that is less than full. The effect is that every node is at least two-thirds full. Keeping the nodes this full uses the available space more efficiently and reduces the number of nodes necessary to hold a fixed number of keys. This improves the search time for the tree. "B*-trees have the following characteristics:

1) They are m-way search trees that are either empty or have a height >= 1.
2) The root node has at least two children, at least one value and a maximum of 2(2m - 2)/3 + 1 children.
3) All non-terminal nodes other than the root node have at least (2m-1)/3 children.
4) All terminal nodes are at the same level.
5) All non-terminal nodes with k sub-trees have k-1 keys." (Miller,1987)

A diagram of a B*-tree follows. (see Fig. 4.3)

The rules for insertion and deletion for B*-trees are quite complex. They depend on how full the node in question, and its siblings, are. To find out more about search trees and B*-trees in particular, a good place to start is with any file processing text. Nancy Miller (1987) gives a nice description of information storage and retrieval techniques.

4.5. Triangulation

Triangulation of a set of points is a process that is done regularly in computer graphics. Many different triangulations are possible for a particular set of points. An optimal solution is to carry out a triangulation so that the points inside a triangle are closer to its own vertices than to the vertices of any other triangle (Wolberg, 1990). This is called an optimal triangulation and it avoids generating triangles with sharp angles and long edges. This is important for image processing applications as it ensures that only nearby data points will be used in the surface patch computations that will follow.

Lawson (1977) describes how to optimise an arbitrary triangulation of some data points. He gives the following three criteria for optimality:

1) Max-min criterion: For each quadrilateral in the set of triangles, choose the triangulation that maximises the minimum interior angle of the two resultant triangles. This tends to bias the triangulation against long, thin triangles. In the diagram, figure (a) shows triangle ABC being chosen over triangle BCD using this

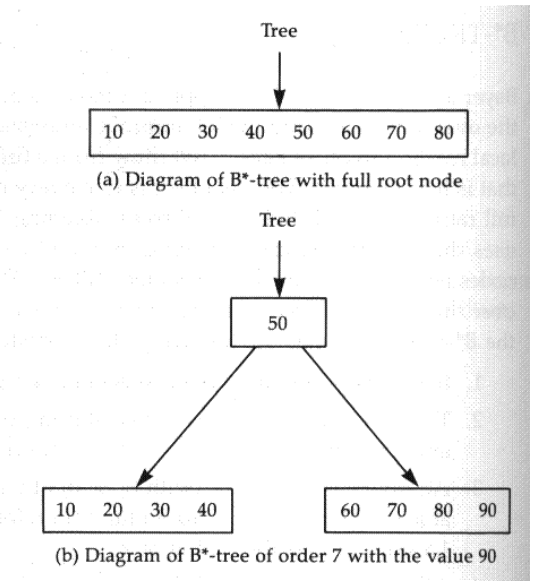


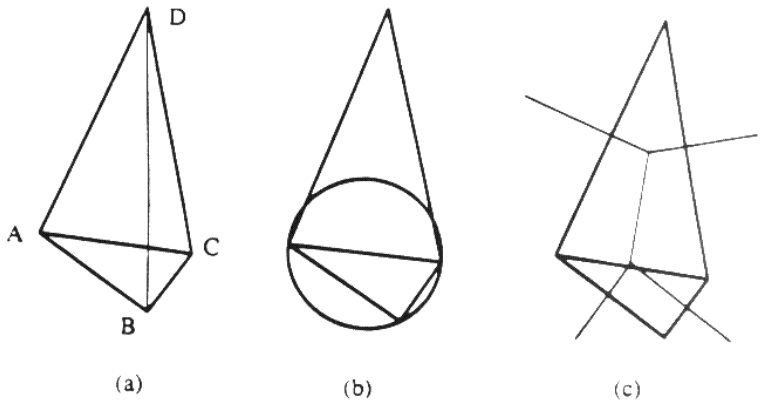
Fig. 4.3 : B*-tree example, Miller (1987)

criterion. This technique has a computational complexity of O(N**4/3).

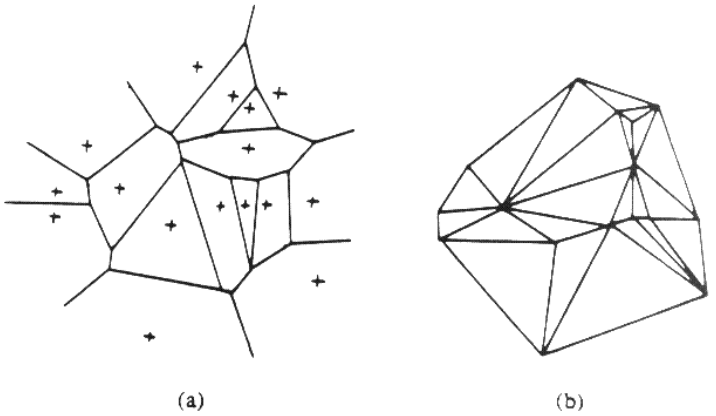
2) The circle criterion: For each quadrilateral in the set of triangles, pass a circle through three of its vertices. If the fourth vertex is outside the circle, then the quadrilateral should be split into two triangles using the diagonal that doesn't pass through the fourth vertex. Otherwise, use the diagonal through the fourth vertex. This is illustrated in figure (b).

3) Delaunay triangulation: Construct a Voronoi diagram, or Delaunay tesselation, for the data points. Voronoi diagrams are the result of collecting all the line segments that result from finding perpendicular bisector to all pairs of data points. A Delaunay triangulation is the result of joining adjacent regions on the Voronoi diagram (see Fig. 4.4) An O(Nlog2N) recursive algorithm for determining the optimal triangulation is given in Lee (1980).

Algorithms and code for triangulations are very common. It would be worthwhile to look around for suitable code to reuse rather than writing up your own code for triangulating data points.



Three criteria for optimal triangulation



(a) Delaunay tesselation; (b) Triangulation

Fig. 4.4 : The Delaunay triangulation, Goshtasby (1986)

5. EXAMPLES

Finding examples of morphing is a matter of watching commercials on television or going to a movie. Almost any film with Industrial Light and Magic or any other graphics studio's name in the credits is likely to include morphing. Special effects have always been attractive to the general public, but morphing seems to have really captured the viewers' imagination. Would Michael Jackson's "Black or White" video have sparked as much discussion without the morphing? Would people have seen "Terminator 2" ten times if they'd used old-fashioned cross-dissolves when the T-1000 changed shape? I don't think so. A lot of the following information comes from an article in Computer Graphics World by Peter Sorenson (1992).

Audiences have been spoilt with wonderful special effects for years. It's possible that they were taking them for granted and not really noticing and thinking about them until morphing emerged. These days computer graphics is not merely imitating reality, it is going beyond it. Here are a few notable examples of morphing in movies, commercials and demonstrations.

5.1. Movies

One of the most well known application areas for morphing is in films. A 70 second sequence in "The Abyss" is still being talked about in graphics circles. (see Fig. 5.1) This, of course, was the pseudopod scene. A lot of people are going to see movies just to see the special effects. For "Terminator 2" and "Lawnmower Man", the effects were a major drawcard. So, who did these movies, and how did they do it?

Willow

Company: Industrial Light and Magic. 1987

Scene:

Willow is using a wand to transform a sorceress back to her true form. The sorceress starts as a goat, becomes an ostrich, then a peacock, a turtle, a tiger and finally her true form.

Technique:

ILM used deformable puppets of each animal so that they could stretch them into the correct shape for each change. The actual change over between the puppets was done using 2-D morphing. This combination of puppetry and morphing earned ILM an Academy Award nomination.

The Abyss

Company: Industrial Light and Magic 1989

Scene:

The pseudopod comes into the the human habitat and explores. The pseudopod is made out of water and shaped like a worm. It attempts to communicate with the characters by mimicing their facial movements.

Fig. 5.1 : Pseudopod from The Abyss, Wolberg (1990)

Technique:

The body of the pseudopod was animated using traditional computer animation techniques. The face of the pod was morphed in 3-D using data from complete 3-D digitisations of the required facial expressions. The "morf" program written for "Willow" was used to do the interpolations between facial expressions.

Indiana Jones and the Last Crusade

Company: Industrial Light and Magic 1988

Scene:

The villain believes he has found the grail and drinks from the cup. He has chosen the wrong cup and proceeds to age rapidly until he dies and shrivels up like a mummy. (see Fig. 5.2)

Technique:

2-D morphing was used to blend between images of the villain's face as he died. A series of three increasingly grotesque masks were blended together in the sequence. The facial movements for the masks were made using a motion-control rig.

Terminator 2

Company: Industrial Light and Magic 1991

Scene:

Nearly every scene included morphing! More specifically, scenes where the T-1000 changed shape - from Sarah Connor; from floor to prison/asylum guard then back to himself; turning around within himself in a fight scene and parts of his body becoming weapons.

Technique:

Both 2-D and 3-D morphing was used in Terminator 2. For scenes with relatively little movement and fairly close initial and final images, 2-D morphing was used. The change from Sarah to the T-1000 is an example of the 2-D morphing that took place. To distract the viewer and make the scene look more realistic, different parts of the actors were morphed at different times. In other scenes, a 3-D morph took place.

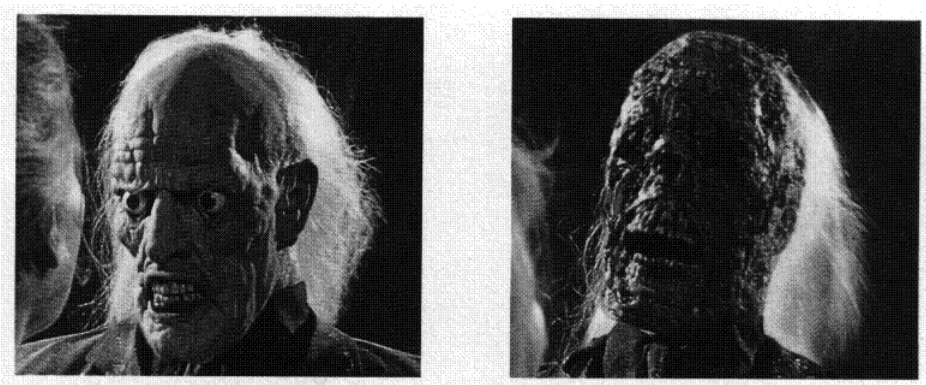


Fig. 5.2 : Morphing scene from Indiana Jones, Wolberg (1990)

Examples of where 3-D morphs would have been used are the scene where the T-1000 comes up out of the floor and the scene where he slips into the helicopter. The animators worked with full 3-D digitised data of the actors along with complete texture maps for each actor.

Star Trek 6 - The Undiscovered Country

Company: Industrial Light and Magic 1991 (probably)

Scene:

A few places in the film. The shape-changer, played by Iman, changed into many different forms. Some of these were: adult female, young girl, monster (her true form) and Captain Kirk.

Technique:

All of these changes would have been done using a 2-D morph. The scenes tended to have the actors standing fairly still for the morphs, making the job a bit easier. The difference in sizes between the actors increased the difficulty of the morphing, requiring extreme warps between the images.

Freddy's Dead: The Final Nightmare

Company: Sidley/Wright

Scene:

3 second scene where the backyard falls away to reveal the whole Earth.

Technique:

Matt paintings of the backyard and the Earth were 2-D morphed and then conventionally composited with a miniature of a house. As the neighbourhood painting falls away, it is twisted to blend with a small part of the Earth picture.

5.2. Commercials and Demonstrations

Along with films, commercials are the main application of morphing. The aim of a commercial is to get the attention of the viewer for long enough to deliver a message. When advertising their own expertise, animation companies are tending to use morphing to show just how skilled they really are. A selection of morphing examples follows...

Plymouth Voyager

Company: Pacific Data Images 1991

Effect:

The 1990 Plymouth Voyager van is morphed into the 1991 model.

Technique:

PDI used a 2-D morph which worked on different parts of the car at different times. To reduce distortion of the background during the morph, parts of the vehicle were photographed separately for easy manipulation. For other elements, painting software was used to separate the elements and generate mattes.

"G-Force" - Nintendo

Company: Sidley/Wright 1991

Effect:

The boy's face is stretched and contorted as if it has been affected by gravity.

Technique:

A mesh was put over the boy's face and then pulled around by the animator until the facial position was what the animator wanted. 2-D morphing was done between these key-frames to get the final sequence. (see Fig. 5.3)

Exxon

Company: Pacific Data Images 1991

Effect:

A moving car turns into a tiger.

Technique:

The tiger was filmed on a stage while the car was filmed on a mountain. Both were filmed using motion control systems to allow exact retakes. The 2-D morph was done with flair; the car ripples as it morphs, the ripples becoming the stripes of the tiger.

Rosendahl

Company: Pacific Data Images 1991

Effect:

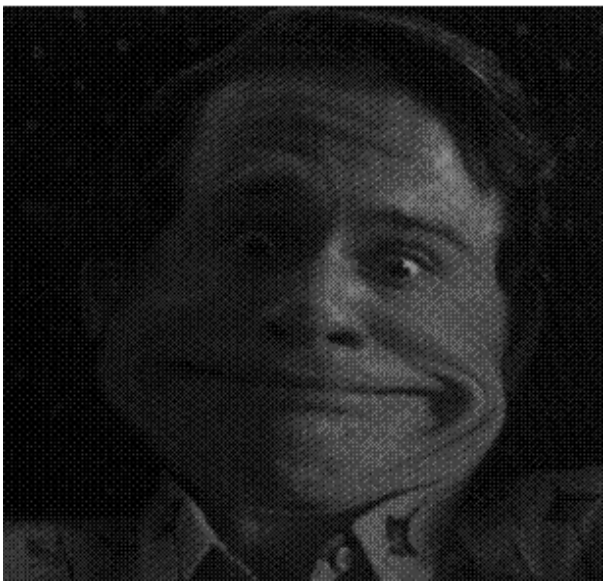


Fig. 5.3 : G-Force commercial, Sorenson (1992)

Three year old Eric Rosendahl morphs into his father, Carl Rosendahl, president of PDI.

Technique:

This in-house demonstration uses 2-D morphing. The boy's head warps to make it the same size and structure as that of the father. At the same time, the other facial features (eyes, nose, teeth etc.) morph into those of the father.

5.3. Music Videos

"Black or White" - Michael Jackson

Company: Pacific Data Images 1991

Effect:

A series of dancers are morphed into each other, while dancing. Jackson turns into and out of a black panther while walking.

Technique:

Both examples of morphing were 2-D morphs. The series of dancers were edited in many orders to get a sequence that flowed nicely. Then the end of each dancer's part was morphed onto the beginning of the next. The morphs were staggered so that different parts of each dancer changed at different times, for example, one of the girl's pony tail appears before her face even begins to morph in. (see Fig. 5.4)

For Jackson to change to and from the black panther, a similar technique to the Exxon clip was used. Care was taken to keep the panther and Jackson walking the same path with the same timing of steps. The camera was controlled to ensure the same angle was used for each take. Jackson had to be careful to have his body in a good position for smooth shape changing.

"Remember" the Time - Michael Jackson

Company: Pacific Data Images (probably) 1992

Effect:

Jackson appears out of a pile of dust, morphing into a metallic man before becoming his human form.

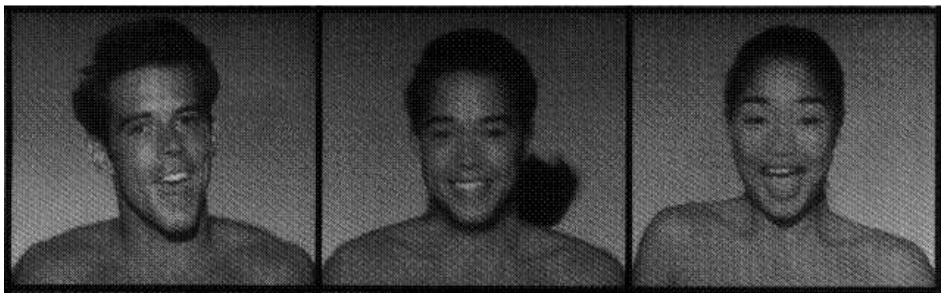


Fig. 5.4 : Black or White, Sorenson (1992)

Technique:

This clip probably used both 2-D and 3-D morphing. The metallic (gold) man was more than likely a 3-D model that morphed from the dust, into Jackson's shape. 2-D morphing would have taken place between the 3-D model and Jackson's real body.

6. COMPANIES AND PRODUCTS

There are a lot of companies who have developed a reputation for being exceptionally good at morphing. The following list gives names of a few companies and their products for those who are interested in looking into the background of the morphing examples in the previous section.

Cyberware Laboratory Inc.

Carried out the digitising for "The Abyss" and "Terminator 2". They use their Cyberware 3D video laser input system to save and store 3-D data in cylindrical space. Data sampling for the actors' faces in the pseudopod scene from "The Abyss" took approximately 15 seconds to scan and about two minutes to download (Anderson, 1990).

Location:

Cyberware Laboratory Inc.
Monterey, California.

Digital Animation Labs

Animators from DAL used morphing in their contribution to the PBS "Astronomers" series. One of their animators, Karl Sims, presented a short film at SIGGRAPH 1989 in which he brought to life old drawings of "The Deluge" by Leonardo Da Vinci. Location:

Digital Animation Labs
Hollywood, California

Discrete Logic

Produce a software product, "Eddie", for doing morphing. It works with several existing animation packages.

Location:

Discrete Logic
5505 Boulevard St-Laurent
Montreal, Quebec
H2T 1S6
Phone: (514) 272-0525

Industrial Light and Magic

Possibly the most famous company for morphing. It is a division of LucasArts Entertainment, which is why it is involved in so many movies. The movies include: "Willow", "The Abyss", "Indiana Jones and the Last Crusade", and "Terminator 2".

Location:

Industrial Light and Magic
P.O. Box 2459
San Rafael, California
94117

Nova Design / Great Valley Products

Are producing a commercial image processing system for the Amiga which includes a separate program for full image morphing. This product, tentatively called "Montage", will run under AmigaDOS 1.3, 2.04 and up. Head programmer for the project is Thomas Krehbiel.

Contact:

Kermit Woodall
kermit@cup.portal.com

Pacific Data Images

Responsible for the morphing in Michael Jackson's film-clips. Also well known for their commercials for the 1991 Plymouth Voyager and the Exxon car to tiger morph. (see Fig. 6.1)

Location:

Pacific Data Images
Los Angeles, California

Pixar

Pixar's "RenderMan" package was used by ILM for "Terminator 2", "The Abyss" and "Young Sherlock Holmes"; by Hanna Barbera Productions for "Funtastic World of Hanna Barbera" and by Pixar for "Tin Toy". Artists at these companies use various tools to "create movie magic", then use Pixar's "RenderMan" for rendering.

Fig. 6.1 : Exxon - car to tiger morph, Sorenson (1992)

Location:

Pixar
1001 West Cutting Boulevard
Richmond, California
94804
Phone: (510) 236-4000

Sidley/Wright

Created the "G-Force" commercial for Nintendo as well as working on "Freddy's Dead: The Final Nightmare". Sidley/Wright have also done a commercial for Target with a bulging, read to explode purse, and another with animated pots, pans and antique stove. One of their in-house demos is a morph from their president's son into the president.

Location:

Sidley/Wright
Hollywood, California

Softimage

Tout Industrial Light and Magic and Digital Pictures as users of their software. Their system can do complex 3D character animation and special effects. Discrete Logic's "Eddie" works with Softimage. Have their own morphing package available commercially.

Location:

Softimage Inc.
3510 Boulevard St-Laurent
Montreal, Quebec
H2X 2V2
Phone: (514) 845-1636

7. PUBLIC DOMAIN SOFTWARE

The only true Public Domain morphing software I have found is "Morphine" by Mark Hall (1992). He has made it available to be used and modified so that he, and other people, can get a taste of morphing for themselves. He admits that there may be deficiencies in the code and asks that those who see anything wrong should let him know at:

foo@cs.rice.edu OR
hall@siggraph.org OR
Mark.Hall@eng.sun.com

In the Morphine file is a Makefile and a README file. The following information comes from the README file and use of the program.

To run the program, type:

morphine <filename>

The program uses GIF files and X-windows to show the images. It can run on SGI's, Sun4's, Vax's and rs6000 ("-Drs6000" needs to be added to the CFLAGS in the Makefile). Triangle drawing routines are taken from the Graphics Gems library. Files included are:

CP.gif	ConcaveScan.c
ConcaveScan.o	GraphicsGems.h
Makefile	README
blankgif	gifs.tar
globals.h	main.c
main.o	marko.gif
morphine	oldmill.gif
posting.tar	readgifs.c
readgifs.o	readmesh.c
readmesh.o	srcs.gif
test0	test1
test2	test3
test4	test5
util.c	util.o
window.gif	xv.h
xvgifwr.c	xvgifwr.o

Files "test0"-"test5" are examples of morphing done with the program. The format of the input files follows:

Format:

starttexture <giffile1>
endtexture <giffile2>
backgroundtexture <giffile3>
numtris <number of triangular regions>
steps <number of morphing steps>
colordiff <number between 0 and 768, to define "close enough" colors>
< <outputgifs> <filename> > <-optional

<starting texture coordinates, at beginning of sequence>

<starting texture coordinates, at end of sequence>
<final texture coordinates, at beginning of sequence>
<final texture coordinates, at end of sequence>
<output texture coordinates, at beginning of sequence>
<output texture coordinates, at end of sequence>

The coordinates are groups of three XY coordinates within the texture file. They define triangular regions of texture. There is a starting and ending coordinate for each corner, so the input texture can change over time. So far only the position of the output texture has been changed in the examples.

This program is well-worth looking at if you are planning on writing your own system. Pick up on tricks, and learn from mistakes and you save yourself a lot of time.

Another strategy is to use "archie" to find ftp-able morphing code. Using "archie morph" I found:

Host: ftp.cse.ucsc.edu
Location: /pub
Contents: morph (directory)

Host: uts.mcc.ac.uk
Location: /pub/riscos
Contents: morph (file)

8. CONCLUSION

In this report, I have given an overview of morphing. Morphing is a different process, depending on whether the morph is 2-D or 3-D. A 2-D morph involves warping and blending texture maps from an initial to a final position. 3-D morphing works on 3-D models, changing them from one shape to another. Rendering and texture mapping is separate from the morphing process when doing 3-D morphing. The type of morphing and the algorithm used depend on the application. Often the choice is simple, if you are working with existing images then you will use a 2-D morph, if you are manipulating 3-D models, a 3-D morph would be more suitable.

There are many ways to improve results of morphing. Using a large number of reference points, matching up the corresponding areas on the images, staggering the morph and doing touch-ups after doing the morph are widely used methods of improving quality. Reading up on related material can be very helpful. Related areas include digital image warping, search trees and triangulation.

It is always helpful, and inspiring, to see examples of other people's work. Looking closely at quality morphing can give the animator something to aim for, as well as some ideas for things to try. Most of the examples come from movies, advertisements and short films for animation exhibitions at conferences. Before developing a system for morphing from scratch, it is best to look at what other people are using. Most animators work with an established animation system, for example Alias, and use their morphing module after the animation is done. The pseudopod in The Abyss is an example of this type of approach. Once a solid basis of morphing information has been formed it is time to experiment. You'll soon get the hang of it and then anything is possible!

9. REFERENCES

Anderson S.E. (1990) *Making a pseudopod: an application of computer graphics imagery.* Proc. Ausgraph '90: 303-311.

Bayer R. and E. McCreight (1972) *Organization and maintenance of large ordered indexes.* Acta Informatica 1(3): 173-189.

Benson P. and D. Perret (1992) *Face to face with the perfect image.* New Scientist, 22nd February 1992: 26-29.

Bergland G.D. (1969) *A guided tour of the fast fourier transform.* IEEE Spectrum, July 1969: 41-52.

Bethel W. (1992) Email correspondence and news postings.

Bier E.A. (1986) *Two-part texture mappings.* IEEE Computer Graphics and Applications, September 1986: 40-53.

Blinn J.F. and M.E. Newell (1976) *Texture and reflection in computer generated images.* Communications of the ACM, October 1976, Vol 19(10): 542-547.

Bock D. (1992) Email correspondence and news postings.

Brigham E. (1988) *The Fast Fourier Transform and its Applications.* Prentice-Hall, Englewood Cliffs: NJ.

Crow F.C. (1981) *A comparison of antialiasing techniques.* IEEE Computer Graphics and Applications, January 1981: 40-48.

Dovey D. (1992) Email correspondence and news postings.

Fiume E., A. Fournier and V. Canale (1983) *Conformal texture mapping.* Eurographics '87: 53-64.

Frederick C. and E.L. Schwartz (1990) *Conformal image warping.* IEEE Computer Graphics and Applications. March 1990, Vol 10(2): 54-61.

Goshtasby A. (1986) *Piecewise linear mapping functions for image registration.* Pattern Recognition, Vol 19(6): 459-466.

Hall M. (1992) Email correspondence and news postings.

Heckbert P.S. (1991) *Digital Image Warping - a review.* IEEE Computer Graphics and Applications. January 1991: 114-116.

Heckbert P.S. (1986) *Survey of texture mapping.* IEEE Computer Graphics and Applications. Nov. 1986 Vol 6(11): 56-67.

Kaufman M. (1992) Email correspondence and news postings.

Kent J.R., R.E Parent and W.E. Carlson (1992) *Establishing correspondences by topological merging: a new approach to 3-D shape transformation.* Electronic copy.

Kent J.R. (1992) Email correspondence and news postings.

Kouhia J. (1992) Email correspondence and news postings.

Lawson C.L. (1977) *Software for C1 surface interpolation.* Mathematical Software III. Academic Press: London: 161-194.

Lee D.T. and B.J. Schachter (1980) *Two algorithms for constructing a Delaunay triangulation.* International Journal of Computer Information Science, Vol 9: 219-242.

Magnenat-Thalmann N., H.T. Minh, M. de Angelis and D. Thalmann (1988) *Human prototyping.* New Trends in Computer Graphics: Proceedings of CG International '88. : 74-82.

Miller N.E. (1987) *File Structures Using Pascal.* Benjamin/Cummings: California.

Morris D. (1982) *Manwatching.* Triad Paperbacks, London: Great Britain.

Parke F. (1982) *Parameterized models for facial animation.* IEEE Computer Graphics and Applications, 2(9), Nov 1982: 61-68.

Sorenson P. (1992) *Morphing magic.* Computer Graphics World, January 1992: 36-42.

Weiler K. (1980) *Polygon comparison using a graph representation.* In: Proceedings of SIGGRAPH, August 1980: 10-18.

Windley J. (1992) Email correspondence and news postings.

Wolberg G. (1990) *Digital Image Warping.* IEEE Computer Society Press: Los Alamitos, CA.

Wolberg G. (1988) *Image warping among planar shapes.* New Trends in Computer Graphics: Proc. CG International '88: 209-218.

Woodall K. (1992) Email correspondence and news postings.

LIST OF FIGURES

Fig. 2.1 Morphing process for Willow, Wolberg (1990) 2
Fig. 3.1 Convex and star-shaped objects. 4
Fig. 3.2 Facial Regions, Magnenat-Thalmann (1988) 5
Fig. 3.3 Marilyn to Bogart (early stages), Magnenat-Thalmann (1988) 5
Fig. 3.4 Marilyn to Bogart (later stages), Magnenat-Thalmann (1988) 6
Fig. 3.5 3-D transformation, Kent et. al. (1992) 7
Fig. 3.6 User control of morphing process, Kent et. al. (1992) 8
Fig. 3.7 The T-1000 morphs out of Sarah Connor, Sorenson (1992) 9
Fig. 3.8 Tracking the eye movement when looking at a face, Morris (1982) 9
Fig. 4.1 Filters, Heckbert (1986) 12
Fig. 4.2 A linear system with input, impulse response and output. 17
Fig. 4.3 B*-tree example, Miller (1987) 19
Fig. 4.4 The Delaunay triangulation, Goshtasby (1986) 20
Fig. 5.1 Pseudopod from The Abyss, Wolberg (1990) 21
Fig. 5.2 Morphing scene from Indiana Jones, Wolberg (1990) 22
Fig. 5.3 G-Force commercial, Sorenson (1992) 24
Fig. 5.4 Black or White, Sorenson (1992) 25
Fig. 6.1 Exxon - car to tiger morph, Sorenson (1992) 28

TABLE OF CONTENTS

Table of Contents ii
List of Figures iv
Abstract v
Acknowledgements vi
Introduction 1
What is Morphing? 2
Types of Morphing 3
2-D Morphing 3
3-D Morphing 4
Ways to Improve Results 8
Helpful Information 10
Digital Image Warping 10
Texture Mapping 10
Digital Image Warping - George Wolberg 11
Chapter 1 - Introduction 12
Chapter 2 - Preliminaries 13
Chapter 3 - Spatial Transformations 13
Chapter 4 - Sampling Theory 14
Chapter 5 - Image Resampling 14
Chapter 6 - Antialiasing 14
Chapter 7 - Scanline Algorithms 15
Chapter 8 - Epilogue 16
Appendices 16
The Fast Fourier Transform 17
B*-trees 18
Triangulation 18
Examples 20
Movies 21
Commercials and Demonstrations 23
Music Videos 25
Companies and Products 27
Public Domain Software 30
Conclusion 32

References 33

Appendices

APPENDIX A

Email and Usenet Contacts.

The views expressed by these wonderful people have been of invaluable help to my research.

Jeremy Lee	- s047@sand.sics.bu.oz.au
Glenn Clapp	- gclapp@thunder.sim.es.com
Lance Norskog	- thinman@netcom.com
Eric A. Haines	- erich@eye.com
James Kent	- jkent@groucho.cgrg.ohio-state.edu
Wes Bethel	- wes@ux6.lbl.gov
Samuel Uselton	- uselton@nas.nasa.gov
Mark Hall	- mark.hall@eng.sun.com
Rich Neiswonger	- rudedog%dogpound@wupost.wustl.edu
Jay Windley	- jwindley@asylum.cs
Juhana Kouhia	- jk87377@cc.tut.fi
Kermit Woodall	- Kermit@cup.portal.com
Michael L. Kaufman	- kaufman@delta.eecs.nwu.edu
Eliot Feibush	- eliotf@holst.tamri.com
Thad Beier	- thad@lever.asd.sgi.com
Dave Bock	- dbock@doppler.ncsc.org
B. Moghaddam	- bmoghad@sitevax.gmu.edu
Bill McGonigle	- flowerpt@coos
Leo Kaplin	- kaplin@acsu.buffalo.edu
Bob Stevens	- bstevens@hal.gnu.ai.mit.edu
Mike McDonnell	- mike@zogwarg.etl.army.mil
Stuart Kreitman	- skk@wsl.dec.com
Brian Ekins	- ekins@ingr.com
Donald Dovey	- dovey@renoir.llnl.gov
Richard Ottolini	- stgprao@xing.unocal.com
Sean Schur	- schur@venera.isi.edu
Cindy Ferguson	- cindy@saturn.ucsc.edu
Dimitrios Valsamis	- U45561@uicvm.bitnet
Randy Rohrer	- rmrohre@afterlife.ncsc.mil

APPENDIX B

This is the message that was posted to Usenet and people who had shown an interest in morphing on Usenet in the first six months of 1992. All of the responses are available for anonymous ftp.

Subject: Morphing - What do you know about it??

Summary: information sought about morphing for a report

Keywords: morphing

Organization: Curtin University of Technology

Date: Tue, 16 Jun 1992 03:34:56 GMT

Hi,

I'm doing a project which is a survey/overview of techniques being used for morphing around the world. Wes Bethel posted his responses to my questions recently. I have quite a few other replies, but I don't mind getting a few more. I'll be making the report available via ftp when I've written it up. If you're interested in a copy, I'll post information on how to get it once it's finished.

I've sent this message to people who've posted about morphing this year. If you've already got the message, you don't need to post again. (I'm not that cruel) Some of my messages bounced, so don't be surprised if you've posted about morphing but didn't receive a message.

Here's my questions, you can follow them if you like. If you have a report or file of information that would be easier for you to send that would be fine.

Thanks for any information you can give. If you can't answer all of the questions, don't worry. Just answer those you can find time for.

I hope this isn't to much to ask, any input is appreciated.

Hope to hear from you soon,

Valerie Hall

Curtin University of Technology

Perth, Western Australia

val@marsh.cs.curtin.edu.au

****

1) What is your interest in morphing?

(eg. hobby, research thesis/project/assignment, professional animation.)

2) What morphing work have you done?
(eg. used a system, created a system, researched morphing)

3) Do you have references? If so, please list them.
(ftp sites included)

4) Do you have any programs, self made or proprietary that you use for morphing?
(give software title and source)

5) Does your morphing program rely on other software? If so, what software?

6) Could you list any companies that you know are doing morphing?

7) How would you describe morphing? What is your technique?

****

INTRODUCTION TO MORPHING

Computing Topics 591

by

Valerie Hall
875001H

July 1992

Supervisor

Andrew Marriott
Lecturer, School of Computing Science

ABSTRACT

The aim of this paper is to provide introductory information about morphing. It should serve as a starting point for others interested in the area. The paper provides background material and references for further reading. A section listing examples of morphing will help those who would like to see more morphing.

ACKNOWLEDGEMENTS

I'd like to thank all of the people who took part in my email survey on morphing. Special thanks goes to James Kent for sending me a copy of his paper, and to Mark Hall and Rich Neiswonger for the use of their morphing programs. I'd also like to thank my supervisor, Andrew Marriott, for his support and patience throughout this project.