

Southern Illinois University at Carbondale
Department of Computer Science

Technical Report
April 17 1997

Morphing Between Multiple Images

Todor Georgiev and Michael Wainer

Abstract

We propose a new, Affine Space treatment of the theory of Morphing. Instead of the standard approach, where morphing is a transition between two images, we propose a natural way of generalizing the theory to an unlimited number of initial images. Any curve in the Affine Space defined by these images represents, in our approach, a movie. Applications in Animation and Image Based 3-D are discussed.

1. Introduction and basic notations.

Morphing can be defined as a smooth transition between two images, that preserves features. In other words, it has to map each feature from the Source Image to the corresponding feature in the Destination Image. For example, if we want to create a morph between pictures of two different people, we need to map the nose of person A to the nose of person B, the left eye of A to the left eye of B, and so on. . . This makes morphing different, and more complicated, than cross-fading, where image A is faded out, while image B is simultaneously faded in. Morphing and crossfading are just two of the infinitely many possible smooth transitions between two images.

An image is a color-valued function, defined on (a subset of) the plane. In other words, it is a function that maps each point (u, v) of the plane into the color space. We will denote such functions by f, g, h, \dots and will write $f: (uv) \rightarrow \text{color}$.

Also, we will use mappings from the plane to itself. Wolberg (ref.1) calls them “spatial transformations”. They establish geometric relationship (correspondence) between each point of (the plane of) the Source Image and the corresponding point in the Destination Image. We will denote them by the Greek letters $\varphi, \psi, \theta, \dots$ and will write $\varphi: (uv) \rightarrow (xy)$ to denote a map from the (uv) plane into the (xy) plane.

Often we will use the composition of two or more mappings. Assume $\varphi: (uv) \rightarrow (xy)$ is a mapping from (uv) to (xy). Assume also that $h: (xy) \rightarrow \text{color}$ is a color valued function (picture) on (xy). Then if \vec{v} is any point in (uv), $(h \circ \varphi)(\vec{v})$ is its a color. We can talk about the picture (or, function) $g = h \circ \varphi$ and we don’t have to specify any point just as we do not specify any point when we talk about the function h. Note that $h \circ \varphi$ is a function on the (uv) plane. It is called the pull-back of the function h generated by φ because, in a sense, h is pulled back from (xy) to (uv). A pull-back is defined for any p-form; here we have the special case of functions, which are 0-forms (ref.2).

2. Structure of the morphing transform.

Morphing has two components: warping and crossfading. Warping stretches the first image as if it was on a rubber surface, while crossfading redefines the color of each pixel as a blend between the colors of the corresponding pixels in the Source and Destination images.

Warping.

In order to define a warping transformation, we first need to be given a mapping φ from the (uv) plane of the Source Image to another copy of the plane, on which the Destination Image lives.

$$\varphi: (uv) \rightarrow (xy) \quad (1)$$

This map induces the morphing by telling us exactly how we “change the shape” of the plane. Now, if the source image is

$$f: (uv) \rightarrow \text{color} \quad (2)$$

then

$$g = f \circ \varphi^{-1} \quad (3)$$

is a new image on (xy) , induced by the map φ^{-1} . This is the pull-back of f by φ^{-1} .

To continue with our construction of the warp, we will use the identity map $i: (uv) \rightarrow (xy)$, which is “doing nothing”, i.e. $i(u,v) = (x,y) = (u,v)$. Its inverse defines another pull-back of the image on (uv) to an image on (xy) , identical to itself.

The first pull-back (3) is a 100% warp, or warp with parameter 1.

The second pull-back is a 0% warp, or warp with parameter 0.

Now, we will define a warp g_t with parameter $0 < t < 1$. This is easy to do. Define the map $\psi_t: (uv) \rightarrow (xy)$ by

$$\psi_t = t\varphi + (1-t)i \quad (4)$$

The warp we need is the pull-back of f generated by ψ_t^{-1}

$$g_t = f \circ (t\varphi + (1-t)i)^{-1} \quad (5)$$

This is a set of images, defined on (xy) and parametrized by t . Also, if the inverse is not unique, we get several choices for g_t .

Note: Here we were using (uv) and (xy) as 2-dimensional vector spaces and **not** as general 2-dimensional manifolds. In the general case we do not have addition and multiplication by a number, as in (5). So, we need to define a 1-parameter set of functions ψ_t such that:

$$\psi_0 = i$$

$$\psi_1 = \varphi$$

There is no unique way of doing that.

Cross-fading after warping.

Given two images, Source and Destination, warping does not completely convert one to the other. Even if correspondence is set up perfectly and the parameter $t = 1$, the transformed image cannot exactly match the

Destination image. A simple example is a person with brown eyes (Destination) and a person with blue eyes (Source). After warping the eyes will remain blue.

This type of problems are solved by changing the color as we warp. This is done by redefining the colors of the image we use as Source. If the Destination Image is $h: (xy) \rightarrow \text{color}$, we define a new Source Image $c: (uv) \rightarrow \text{color}$ on the plane (uv) , which differs from f just by the color:

$$c = t h \circ \varphi + (1-t) f \quad (6)$$

Now, use in (5) image c instead of image f when warping by ψ_t^{-1}

$$g_t = (t h \circ \varphi + (1-t) f) \circ (t \varphi + (1-t) i)^{-1} \quad (7)$$

Notice that the color space needs to be (a subset of) a vector space, or we will have problems as discussed in the previous note.

3. Affine Space Theory of Morphing

Let's consider again formula (4). It looks like the parametric equation of a line that does not pass through the origin (fig. 1).

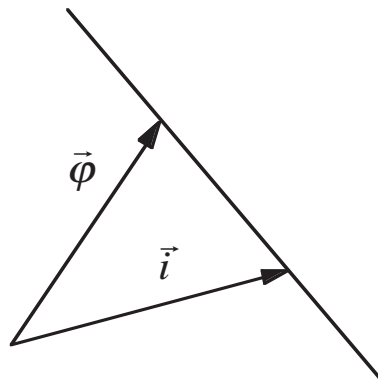


fig. 1

If \vec{i} and $\vec{\phi}$ are two linearly independent vectors, then any point on the line defined by them can be represented as $t \vec{\phi} + (1-t) \vec{i}$, where $-\infty < t < +\infty$. This is not a coincidence.

The mapping ϕ , considered in the previous chapter was taking points from one vector space to another. ϕ doesn't have to be linear, but applying it to a vector always produces a vector. Because the result is a vector, we can define linear combinations of such mappings. For example, the map $\phi + \psi$ is defined for a vector \vec{v} as $\phi(\vec{v}) + \psi(\vec{v})$. This is exactly the type of thing we did in equation (4). The point now is to notice that because we can do linear combinations of maps, the set of all such maps is a linear space. We will call it the space of all maps L. It is infinite dimensional.

Now, if ϕ and i are two such maps, $t\phi + (1-t)i$ would describe a line in the space of all maps. Notice that ϕ and i are linearly independent because ϕ is not the identity or proportional to it. Because of that, none of the maps collapses the whole plane into the point (0,0), and none of the morphings expand the plane to infinity.

This is our new geometric viewpoint on what morphing is. It is the pull-back generated by a map that lies on the line in L that connects ϕ and i .

4. Morphing between multiple images.

If we accept the above point of view, then restricting ourselves to just two maps is not natural. Let $\phi, \psi, \theta, \dots$ be maps from (uv) to (xy) as described above. Then we can think of them as vectors and write any linear combination

$$\alpha\phi + \beta\psi + \gamma\theta + \dots \tag{8}$$

This is another vector in the space L of all maps. If $\phi, \psi, \theta, \dots$ are linearly independent, this map does not collapse everything into the point (0,0) unless all coefficients happen to be zero.

Because L is very high dimensional, we have no limitation on the number of maps in (8). For n -dimensional space we have only n independent vectors, but here n is huge. (We refer to the n -dimensional space of all maps, L , not the 2-dimensional space of our source images.) In this way, we can create a valid morphing transformation starting from any number of Source Images.

There is only one condition that (8) has to satisfy:

$$\alpha + \beta + \gamma + \dots = 1 \quad (9)$$

(9) is true if the pictures we want to start with contain at least one point, \vec{v} , which appears on the same place in the morph. This is always the case because we want the edges of the initial pictures to go to the edges of the morph; also, all pixels of the background usually do not change during morphing. We can write the above condition as:

$$\alpha\varphi(\vec{v}) + \beta\psi(\vec{v}) + \gamma\theta(\vec{v}) + \dots = \vec{v} \quad (10)$$

We want (10) to be true for all values of $\alpha, \beta, \gamma, \dots$.

A special case is $\alpha = 1, 0 = \beta = \gamma = \dots$. This would be a morphing into the first image. Another special case is $\beta = 1, 0 = \alpha = \gamma = \dots$ and so on. In this way we get

$$\varphi(\vec{v}) = \vec{v}, \quad \psi(\vec{v}) = \vec{v}, \quad \dots \quad (11)$$

Substituting (11) into (10), we get (9).

In this way, (8) and (9) define morphing between n pictures. In order to actually create such a morph, we need to set up n correspondence maps $\varphi, \psi, \theta, \dots$ between pairs of images (including the identity), using an appropriate GUI. Usually we set up n sparse correspondence meshes. Then we need to calculate the map (8), satisfying condition (9). The pull-back (of the “cross-faded” color function) generated by this map is the morph.

Notice that now the morph is a set of images depending on $n-1$ parameters. These are the n coefficients $\alpha, \beta, \gamma, \dots$, constrained by condition (9). What’s normally considered a morph is dependent on a single parameter, whereas our concept of morph is much richer than that. A single movie cannot adequately sample this morph space.

Any curve in the above $n-1$ dimensional manifold represents a movie. We have so much freedom in choosing different curves that we can think of creating different movies starting from the same set of just a few pictures. Setting up the correspondence is done just once.

5. The Affine Space

It turns out the freedom we have in morphing is too much. There are too many parameters to choose and too little understanding as to what the result is going to be. Trying to imagine what will happen during morphing is difficult even in the case of 3 initial pictures. In order to reduce this uncertainty, we'll try to use the general mathematical properties of the objects we are dealing with.

In the case of 3 pictures we have 3 linearly independent maps, φ, ψ, i . Then (8) and (9) are the parametric equation of a plane (fig. 2),

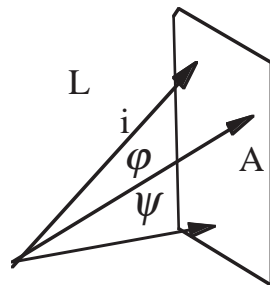


fig. 2

that is embedded in the infinite dimensional space of all maps, L . The origin does not belong to it. This plane is called the two-dimensional affine subspace of L . It is a particular implementation of a 2-dimensional affine space, with underlying vector space of all differences of points in the plane. Note that any point on this plane defines a morphing by its pull-back.

In the general case we have n pictures, n maps, and a $n-1$ dimensional affine subspace, A , of the space of all maps, L . It could be called “the morph space” because any morphing is produced by the pull-back of a point in A . Equations (8) and (9) are called “affine combination” (ref. 3).

The most general properties of morphing come from what is already known about affine spaces.

The affine space is not a vector space because $\bar{0}$ does not belong to it. Because of this fact, we cannot use vector space addition to add elements of the affine space. (Here we have in mind our implementation. An abstract affine space is not a vector space and there is no addition of the elements by definition.) In other words, if we restrict ourselves to stay only in A , and ϕ and ψ belong to A , then $\phi + \psi$ is undefined. It is defined in the big space L and we can check that the result is a new vector that does not belong to A . That’s why addition and, in general, linear combination is undefined in affine spaces.

Interpreted in terms of morphing, there is no “central morphing image”, just as there is no $\bar{0}$ vector; if two maps, ϕ and ψ , define morphs, their sum, $\phi + \psi$ does not define a morph.

The set of all differences of points that belong to A forms a vector space, called “the underlying space” (Ref. 4), or “the associated space” (Ref. 3). It is obvious that adding a vector from this space to a point in A produces another point in A . Sometimes this property is used as a definition of affine spaces (Ref. 4).

The above construction gives us a way of deriving one point in A from another. Starting from ϕ we can add any vector χ to it and get another point in A . χ can be represented as $\chi = \alpha(\psi - \theta)$, where ψ and θ are any two points in A . In this way we can build curves step by step, using small increments with vectors like χ , and create movies. Also, now we have understanding as to what the “direction of the change” is and how to create desired changes.

At the end we will point out that (8) and (9) can be written as

$$\phi + \beta(\psi - \phi) + \gamma(\theta - \phi) + \dots \quad (12)$$

Only the first element is from A , the others are vectors. So, this is just a transformation from φ to another point in A , as described above.

6. Morphisms of Affine Spaces.

Here we use mathematical ideas and terminology from (Ref. 4).

Given a movie, we want to be able to replace the characters in it with other characters, and get another movie. For example, if a person is performing some actions, we want to be able to “transport” his actions (not him) to another movie, where a different person will be doing the same things.

We represent a movie as a curve in the affine space of all mappings, A . The main tool for building the curve is the affine structure of the space which allows us to add a vector to a point. So, in terms of affine spaces we want to map (transform) A and the mathematical structures in it to another affine space, B , in such a way that structures are preserved.

A mapping from one object to another, which preserves structures is called a morphism. For example, a morphism from one vector space to another is a transformation T such that $T(\vec{v} + \vec{w}) = T(\vec{v}) + T(\vec{w})$. This preserves the vector space structure, i.e. the linear combinations. If a morphism is invertable, it is called isomorphism. Well-known facts are that if two vector spaces have the same dimension, then isomorphisms exist; they are linear transformations; the two spaces are called isomorphic.

Definition: Let A and B be affine spaces with underlying vector spaces V and W . The mapping $\tau : A \rightarrow B$ is called an affine morphism if there is a linear mapping $T : V \rightarrow W$ such that whenever $\varphi \in A$, $\chi \in V$,

$$\tau(\varphi + \chi) = \tau(\varphi) + T(\chi) \tag{13}$$

where the first $+$ is adding a vector in A , and the second $+$ is adding a vector in B .

It can be shown that given τ , T is unique.

$$\begin{aligned}\tau(\varphi) + T_\varphi(\chi) &= \tau(\varphi + \chi) = \tau(\psi + (\varphi - \psi) + \chi) \\ &= \tau(\psi) + T_\psi((\varphi - \psi) + \chi) = \tau(\psi) + T_\psi(\varphi - \psi) + T_\psi(\chi) \\ &= \tau(\psi + \varphi - \psi) + T_\psi(\chi) = \tau(\varphi) + T_\psi(\chi)\end{aligned}$$

So, $T_\varphi = T_\psi$.

If $V = W$ and T is the identity, τ is called a **translation**. A translation of the affine space of mappings produces much more than an ordinary translation of the characters in the movie. The new characters can have completely different shapes and sizes, but they move in the same way as the original ones.

A general morphism would almost always be more than what we need or can imagine. The only thing that is preserved would be the sequence and timing of the actions, but not the actions themselves. The directions and amplitudes can be changed in the sense that each new action is a linear combination of all actions from the first movie. For example, the transformation $T = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ would switch the motions of, say, arms and legs, so that when the person in the first movie spreads legs, the person in the second movie will raise hands.

Use of general morphisms is much more sophisticated and presumably requires a great deal of experience. A good application would be in modeling expressions of the face. We can produce very fine or very exaggerated combinations between smile, frown, joy, sadness, and so on. In this way we can create expressions that have never existed. Or, in the case of translation, transport the expression of one person to another.

The practical implementation of an affine morphism is not very different from what we did before. We need one new picture, and we have to set up one more mesh that defines the correspondence map ξ between the new picture and the Source picture. Then calculate $\xi - \varphi$. It is not going to be

a vector from the underlying space if $\xi \notin A$, which is almost always the case. Now we just add $\xi - \varphi$ to each point on the curve representing the old movie, and we get a point representing a frame from the new movie.

We are doing an illegal operation in the old affine space. But it is perfectly legal in L. We are just shifting the n-1 dimensional hyperplane we used to call “affine space A” to a new n-1 dimensional hyperplane - the affine space B. This is a translation. It establishes the morphism and it has to go beyond the structures defined in A.

Also, we should notice that if $\xi \in A$, it can not be used above. This condition guarantees that $\bar{0}$ does not belong to B.

In order to set up the complete morphism we have to use T . T can be represented by a matrix in some basis of the n-1 dimensional underlying space. For a basis we can use n-1 independent vectors - for example the differences between φ and each of the maps ψ, θ, \dots . They can easily be calculated using the meshes we already have. Let's denote this basis by $\chi_1, \chi_2, \chi_3, \dots$. Any point in A can be written as $\varphi + \sum a_i \chi_i$. If it represents a frame from the movie, we map it to $\tau(\varphi) + \sum_{i,j} a_i T_{ij} \chi_j$.

The morphism τ is not just arbitrary. We want to control it. Above we assumed that φ goes to ξ , in other words $\tau(\varphi) = \xi$. So, the final result is $\xi + \sum_{i,j} a_i T_{ij} \chi_j$. If T is the identity, we get a translation of $\varphi + \sum a_i \chi_i$ by $\xi - \varphi$, as it should be.

Let's also notice that even when the two underlying spaces, V and W are isomorphic, they do not have to be the same space. If this is the case, we have the same final result, only the basis $\chi_1, \chi_2, \chi_3, \dots$ is different. It should be chosen to be any basis of the new underlying space, W.

The fact that there is a one to one correspondence between the morphism τ and the pair ξ, T (established above) tells us that the method we have described can cover all possible morphisms. We have also shown that a morphism is the most general way of transporting one movie into another. In other words, the above theory is **complete**. No matter what methods are being used, it is not possible that someone comes up with a transformation from one movie to another (or from one face expression to another), which does something that cannot be done in the way described above.

7. Folds and Holes.

If ψ_t is not one-to-one, ψ_t^{-1} may not be a function. In these cases we have to be careful to identify the reason for the problem and create a work around. Otherwise we get holes or folds in the morph. This is a problem noticed by everyone and widely discussed in the literature (ref. 1 and others). Here is a short account of what the problem is.

If two different points happen to be mapped by ψ_t to the same point in the Destination, then the color of that point is not well defined. Is it the color of the first, or the second point from the Source ? Or is it a “mixture” of the two ? . . . This situation is called a fold.

The other bad case is if ψ_t is not onto. Then some part of the Destination Image is not covered and we get a hole. There is no information about the color of these pixels.

One might think that problems like these could be avoided by appropriately choosing φ and θ , or some other way. This, however is not true. We will prove that there is no way to avoid this even if we impose the condition that ψ_t is always constructed as a combination of invertable maps.

In other words, if φ and θ are invertable and linearly independent, then there always are values of α, β , $\alpha + \beta = 1$, such that $\alpha\varphi + \beta\theta$ is not invertable.

Proof: Since φ is invertable,

$$\varphi(x) = \varphi(y) \quad (13)$$

is impossible when $x \neq y$. Exactly the same is true for

$$\theta(x) = \theta(y) \quad (14)$$

Assume $\alpha\varphi + \beta\theta$ is invertable, in other words assume $(\alpha\varphi + \beta\theta)(x) = (\alpha\varphi + \beta\theta)(y)$ is impossible when $x \neq y$. Then

$$\alpha(\varphi(x) - \varphi(y)) = \beta(\theta(y) - \theta(x)) \text{ is impossible.}$$

Since we are free to choose α and β , this means that we must have either $\varphi(x) = \varphi(y)$ and $\theta(y) \neq \theta(x)$

or $\varphi(x) \neq \varphi(y)$ and $\theta(y) = \theta(x)$.

This contradicts the initial condition that φ and θ are invertable (13), (14). So, our assumption that $\alpha\varphi + \beta\theta$ is invertable is wrong. End of proof.

ψ_t is not invertable for some values of the parameters. Then, the color of the corresponding point in the morph is not defined. It has to be given by $f \circ \psi_t^{-1}$, which is not unique.

We have proven that folds in the morph are **unavoidable**. Instead of trying to avoid them, we have to accept them as something natural. Next we will develop a method that makes use of a z-buffer and folds in the morph to create more realistic image-based 3-D graphics. This result, together with the method of morphing between 3 (or more) parallel views is extending the ideas of (Ref. 5, 6) one step closer to the goal.

$f \circ \psi_t^{-1}(\vec{x})$ is a set of colors, each one defined as $f(\vec{v}_i)$, where \vec{v}_i are the points for which $\psi_t(\vec{v}_i) = \vec{x}$. What to do if this is a set of more than one color? Which color to display?

The problem has a nice solution in the case of morphing between different views of a 3-D scene. Assume the camera is located on a plane and always directed perpendicular to it (fig. 3). If we take two pictures from different viewpoints A and B, it has been proven (ref. 6) that we can use morphing

to reconstruct any view along the line connecting these two points. As shown above, folds cannot be avoided.

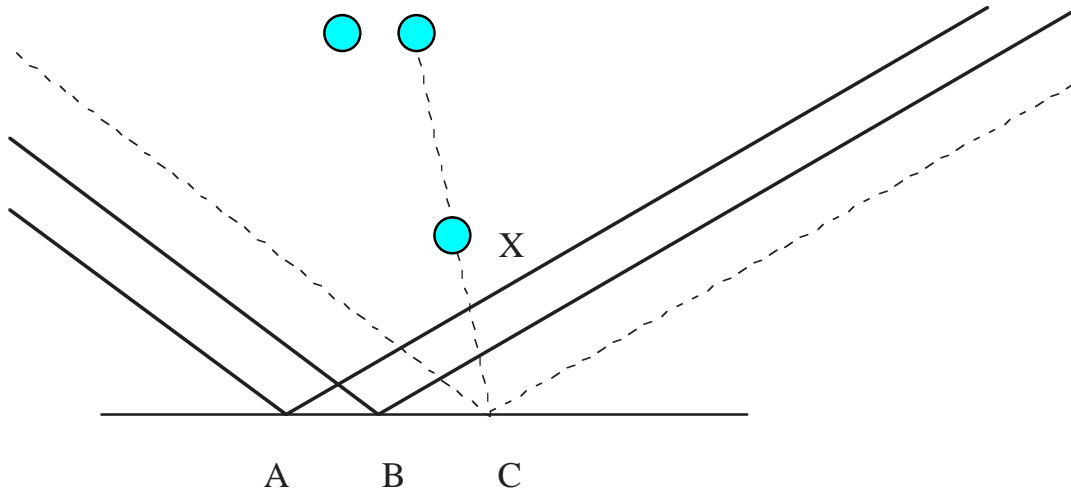


fig. 3

This is not surprising at all because an object X that is in front of other objects can come in the way of the light ray connecting the virtual camera C and the far away objects and make them invisible. Morphing does not “know” that we want to hide them and we get the fold. (We get two different colors and the common solution is to display a mixture.)

In our approach we also get a set of two colors for the pixel. But we also calculate two additional numbers which are, roughly speaking, the speed at which these “color spots” (that now happen to be on the same pixel) “move” on the image plane during morphing. Then we display the color that moves faster. It is in front of the other.

In more detail, we assign a fourth component, z to each RGB color. This component will be used in a z -buffer. Here it is how we calculate z .

For each point \vec{v} on the image plane we define a number $z(\vec{v})$ as $z = |\varphi(\vec{v}) - i(\vec{v})|$. When calculating $\psi_t^{-1}(\vec{x})$ we get a set of vectors $\vec{v}_1, \vec{v}_2, \dots$, each one with its own color $f(\vec{v}_i)$. We attach to that color

the fourth component $z(\vec{v}_i)$ calculated above. Then use it in a z-buffer when displaying the image. Only one of the set of colors will be displayed - the one with the largest z . The others are “hidden behind”, which makes the illusion of 3-D motion more realistic. There is no fold problem.

When morphing between more than two images, we get a set of maps $\varphi, \psi, \theta, \dots$ instead of just the two maps φ, i . Then we can use **any two** of them instead of φ, i to calculate z . But we have to use the same two everywhere. If some points are not visible for a given map, we should try to avoid that map and choose another one whenever possible. Just two maps are enough because they contain the complete information about depth. The results we would get from different pairs of maps are the same because depth is the same.

8. Conclusion and Future Work.

In this paper we have introduced the idea of Affine Space approach to Morphing. It leads in a natural way to a method for morphing between 3 or more initial images.

A very important application is in 3-D graphics. Recent work by (Ref. 8) focused on synthesizing new 3-D views of objects based upon 2-D images. Neuronets were used to generate the new image directly without first having to explicitly recreate a 3-D object model. Later works (Ref. 5, 6) simplify the process by using the idea of parallel views. Our work is developing the method further in that direction. Almost any parallel view of a virtual camera on the plane can be synthesized in one step by morphing between multiple images.

The result of a morphing (the morph) is a large set of images, described by more than one parameter. Any curve in this space could be treated as a movie. This immediately leads to possible applications in Animation. Starting from 3 or more pictures, we can create a very rich set of in-betweens, possibly a whole movie. This saves a lot of work. We can create several movies from the same pictures, and even convert one movie into another.

Another use of the method is as an alternative to (Ref. 7) way of modeling face expressions. If we start with n face expressions, any point in the $n-1$ dimensional Affine Space would be another expression. A morphism of Affine Spaces would transport expressions from one person to another.

Multiple-image morphing also has the potential to be used as a compression tool in video and animation applications. Applying the inverse process should allow us to factor the sequence of frames to create a basis set of images capable of recreating the original movie. Now the entire sequence can be recreated by specifying only the morph coefficients for each frame. Decomposition of frames into overlapping subframes responsible for imaging individual components may simplify this process and promote even greater reuse of basis frames (i.e. a single set of frames may correspond to each spokes-person on a news show, etc.). Fitting the coefficients to an interpolation function would also allow the movie to be synthesized at different frame rates than that of the original. Ways to factor a sequence of images are being explored.

References

1. G. Wolberg, "Digital Image Warping" (IEEE Computer Society Press, Los Alamitos, Ca, 1990).
2. D. Bleeker, "Gauge Theory and Variational Principles" (Addison-Wesley, Reading, Massachusetts, 1981).
3. J. Foley, A. van Dam, S. Feiner, J. Hughes, "Computer Graphics" (Addison Wesley, Reading Massachusetts, 1996).
4. D. Saunders, "The Geometry of Jet Bundles" (Cambridge Univ. Press, 1989).
5. S.Seitz, C. Dyer, View Morphing. In Proc. SIGGRAPH 96, pages 21-30, 1996.

6. S. Seitz, C. Dyer, Toward Image-Based Scene Representation Using View Morphing. Dept. of CS, Univ of Wisconsin, Technical Report #1298, May 1996.
7. Y. Lee, D. Terzopoulos, K. Waters, Realistic Modeling for Facial Animation. In Computer Graphics Proceedings, Annual Conference Series, 1995, pages 55-62.
8. T. Poggio, R. Brunelli, A Novel Approach to Graphics, A.I. Memo No. 1345, MIT and AI Laboratory, February 1992.