



# Warps and Morphs

Mike Land and Tara Puzin  
Linear Algebra  
College of the Redwoods

December 2, 2002

## Abstract

There is an interesting technique in the movie industry called metamorphosis, this technique morphs one digital image into another. When this process is used effectively, the photograph or computer image can be transformed into anything a person wants with dramatic effects.

## Introduction

Morphing is probably used most often in the movie industry with incredible special effects. It has been used in movies such as Terminator and the Abyss in commercials, music videos, and in video games. Morphing is a powerful tool that can enhance many multimedia projects, presentations, education, computer based training. This paper discusses what morphing is and just one technique used to morph one image into another.

*Introduction*

*Objective*

*Mathematics behind . . .*

*Drawing Two Lines*

*Home Page*

*Title Page*



*Page 1 of 23*

*Go Back*

*Full Screen*

*Close*

*Quit*

[Introduction](#)[Objective](#)[Mathematics behind . . .](#)[Drawing Two Lines](#)[Home Page](#)[Title Page](#)[Page 2 of 23](#)[Go Back](#)[Full Screen](#)[Close](#)[Quit](#)

## What is Morphing

The word morph derives from the word metamorphosis meaning to change shape or form. Morphing is achieved by compiling several images that are gradually distorted and faded out while the destination image is faded in. The early images in the sequence are much like the first image. As this procedure progresses forward the middle image is about fifty percent of both images. The first image fades out the second image fades in gradually. When this process is shown in a movie it transforms one object into another.

## Objective

Our objective is to discover the the mathematics behind morphing. Using linear algebra we will transform Rush Limbaugh into the Squidman shown in Figure 1. We will draw a line in the source image and draw a line in the destination image and project  $X$  onto the line  $PQ$  to calculate  $u$  and  $v$ . Once we calculate  $u$  and  $v$  we use that to find  $X'$  in the source image. The color of  $X'$  is then determined and poured into the destination image. Our algorithm will then calculate every pixel in both images. This is, the overview of, how we will perform a morph between two images. (Refer to Figure 1)

## Mathematics behind Morphing

The linear algebra behind morphing involves finding unit vectors, projections, and matrices. In linear algebra, we learn how to project a vector  $\mathbf{b}$  onto another vector  $\mathbf{a}$  (see Figure 2(a)): Note that  $\mathbf{b} - \lambda\mathbf{a}$  is perpendicular to  $\mathbf{a}$  in Figure 2(a). Therefore, the dot product equals zero and we can write

$$\begin{aligned}\mathbf{a} \cdot (\mathbf{b} - \lambda\mathbf{a}) &= 0 \\ \mathbf{a} \cdot \mathbf{b} - \lambda\mathbf{a} \cdot \mathbf{a} &= 0.\end{aligned}$$



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



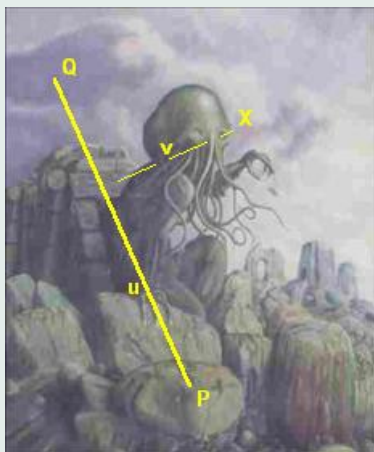
Page 3 of 23

Go Back

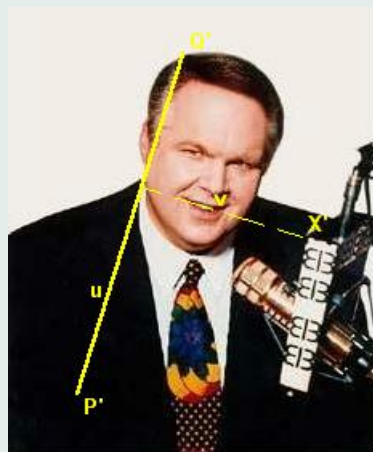
Full Screen

Close

Quit



(a) Destination Image



(b) Source Image

Figure 1:



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 4 of 23

Go Back

Full Screen

Close

Quit

Thus,

$$\lambda = \frac{\mathbf{a} \cdot \mathbf{b}}{\mathbf{a} \cdot \mathbf{a}}. \quad (1)$$

Note that in Figure 2(a),  $0 < \lambda < 1$ . There are two other possibilities (see Figures 2(b) and 2(c)).

## Calculating $u$

Returning to the destination image in Figure 3, ...

We use equation (1) to calculate  $u$ .

$$u = \frac{\overrightarrow{PX} \cdot \overrightarrow{PQ}}{\overrightarrow{PQ} \cdot \overrightarrow{PQ}} \quad (2)$$

Because  $\overrightarrow{PX} = X - P$  and  $\overrightarrow{PQ} = Q - P$ , we can also write

$$u = \frac{(X - P) \cdot (Q - P)}{(Q - P) \cdot (Q - P)}. \quad (3)$$

## Calculating $v$

To find the length of the projection, we project  $\mathbf{a}$  onto the unit vector. If  $\mathbf{a}$  and  $\mathbf{u}$  are nonzero vectors in  $\mathbb{R}^2$  and if  $\theta$  is the angle between these vectors then,

$$\cos \theta = \frac{\mathbf{a} \cdot \mathbf{u}}{|\mathbf{a}| |\mathbf{u}|}.$$

Thus,

$$\mathbf{a} \cdot \mathbf{u} = |\mathbf{a}| |\mathbf{u}| \cos \theta.$$



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 5 of 23

Go Back

Full Screen

Close

Quit

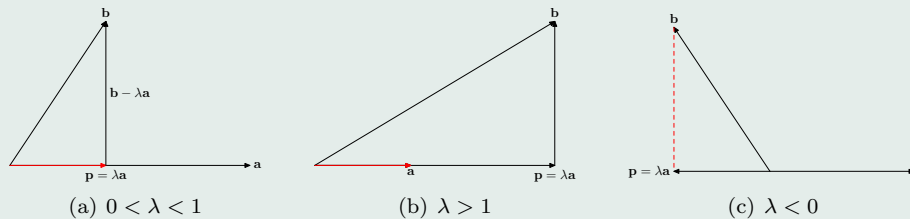


Figure 2: Projecting  $\mathbf{a}$  onto  $\mathbf{u}$

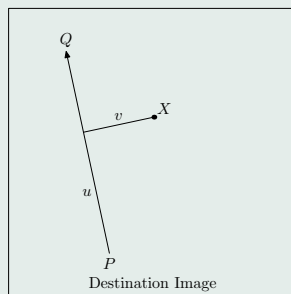


Figure 3: Calculating  $u$  and  $v$



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 6 of 23

Go Back

Full Screen

Close

Quit

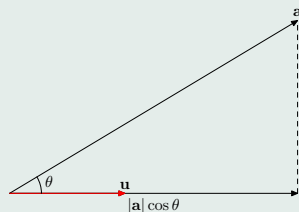


Figure 4: Project  $\mathbf{a}$  onto  $\mathbf{u}$ .

Now  $\mathbf{u}$  is a unit vector so  $|\mathbf{u}| = 1$ . Thus,  $\mathbf{a} \cdot \mathbf{u}$  becomes

$$\mathbf{a} \cdot \mathbf{u} = |\mathbf{a}| \cos \theta. \quad (4)$$

Therefore, the length of the projection is  $\mathbf{a}$  dotted with  $\mathbf{u}$ , as seen in Figure 4. In Figure 5 we are projecting  $\overrightarrow{PX}$  onto the unit vector that is perpendicular to the vector  $\overrightarrow{PQ}$ . We use equation (4) to calculate  $v$ .

$$v = \overrightarrow{PX} \cdot \frac{\text{perp} \overrightarrow{PQ}}{|\overrightarrow{PQ}|} \quad (5)$$

Note that  $\text{perp}(Q - P)$  is a vector perpendicular to  $\overrightarrow{PQ}$  that has the same length as  $\overrightarrow{PQ}$  (see Figure 5). Because  $\overrightarrow{PX} = X - P$  and  $\overrightarrow{PQ} = Q - P$  we can write our equation as

$$v = (X - P) \cdot \frac{\text{perp}(Q - P)}{|Q - P|}.$$



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 7 of 23

Go Back

Full Screen

Close

Quit

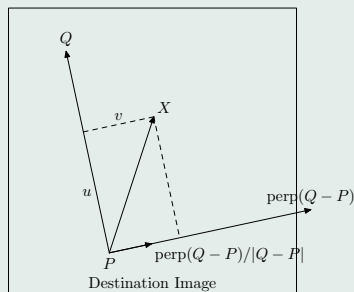


Figure 5: Project  $\overrightarrow{PX}$  onto  $\text{perp}(Q - P)/|Q - P|$ .

## Calculating $X'$

In Figure 6, the location of  $X'$  is in the source image. The location of  $u$  is a percentage up  $PQ$  and  $v$  is a distance perpendicular to  $u$ . Now that we have found  $u$  and  $v$  we can calculate  $X'$  using the following formula.

$$X' = P' + u \cdot (Q' - P') + \frac{v \cdot \text{perp}(Q' - P')}{\sqrt{(Q' - P') \cdot (Q' - P')}} \cdot \text{perp}(Q' - P')$$

Start at  $P'$  move along  $\overrightarrow{P'Q'}$  the same percentage  $u$  that we moved along  $\overrightarrow{PQ}$ . Move perpendicular to  $\overrightarrow{P'Q'}$  a distance  $v$ .

These algorithms transforms each pixel coordinate from the destination image to the source image by rotating, shifting, and or scaling the image to transform the whole image pixel by pixel to end up with a totally new image.



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 8 of 23

Go Back

Full Screen

Close

Quit

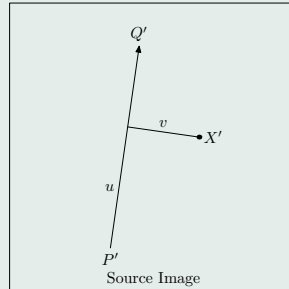


Figure 6: Locating  $X'$ .

## Drawing Two Lines

### Calculating the Position of the Pixel $X$ on the Destination Image

Warping images with many lines is similar to warping images with one line; however, the process becomes much more complex. On the destination image  $X$  is chosen to represent an arbitrary pixel, just as it was when warping with one line. In Figure 7 we can see that now that there are two lines drawn on the destination image. One can clearly see that there are now two  $u$ 's and two  $v$ 's that need to be calculated. The formulas for  $u_1$ ,  $u_2$ ,  $v_1$ , and  $v_2$  are the same formulas used previously for warping an



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 9 of 23

Go Back

Full Screen

Close

Quit

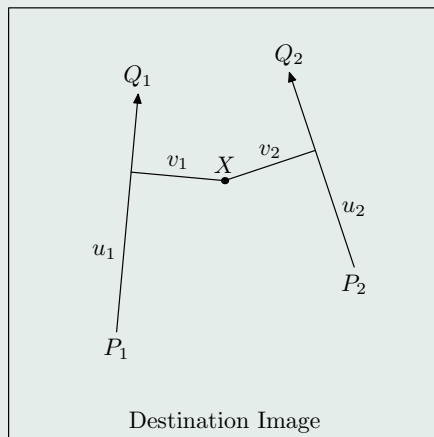


Figure 7: Calculating the position of the pixel  $X$ .



<a href="#">Introduction</a>
<a href="#">Objective</a>
<a href="#">Mathematics behind...</a>
<a href="#">Drawing Two Lines</a>

[Home Page](#)

[Title Page](#)

◀◀ ▶▶

◀ ▶

Page 10 of 23

[Go Back](#)

[Full Screen](#)

[Close](#)

[Quit](#)

image with one line.

$$u_1 = \frac{(X - P_1) \cdot (Q_1 - P_1)}{|Q_1 - P_1|^2}$$
$$u_2 = \frac{(X - P_2) \cdot (Q_2 - P_2)}{|Q_2 - P_2|^2}$$
$$v_1 = \frac{(X - P_1) \cdot \text{perp}(Q_1 - P_1)}{|Q_1 - P_1|}$$
$$v_2 = \frac{(X - P_2) \cdot \text{perp}(Q_2 - P_2)}{|Q_2 - P_2|}$$

The position of each pixel in the destination image must be found by using these formulas.

In order to find the position of the pixel  $X$  in the destination image,  $u_1$ ,  $u_2$ ,  $v_1$ , and  $v_2$  should be calculated first in the destination image. The variable  $u$  will be a percentage up the line  $PQ$ , and the variable  $v$  will be a set distance. The next step is to calculate the positions of  $X'_1$  and  $X'_2$  in the source image. As you see in Figure 8 this can be accomplished by going up the line  $P'Q'$  the same percentage  $u$ , as done in the destination image, and out the set distance  $v$  also calculated in the destination image. Notice that  $X'_1$  and  $X'_2$  are the pixels at the endpoints of  $v_1$ , and  $v_2$ .

## Calculating the color of the Pixel $X'$ in the Source Image

When one digital image is morphed into another, the color of every pixel in the source image has to be calculated and poured into a pixel in the destination image. In this document  $X'$  represents the color of a pixel in the source image and  $X$  represents the



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 11 of 23

Go Back

Full Screen

Close

Quit

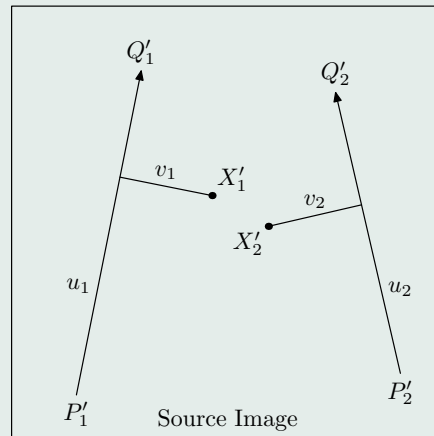


Figure 8: Calculating the position of the pixels  $X'_1$  and  $X'_2$ .



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 12 of 23

Go Back

Full Screen

Close

Quit

position of a pixel in the destination image. To calculate the color of  $X'$  in the source image the two displacements from  $X$  to  $X'_1$  and  $X$  to  $X'_2$  need to be calculated. The formulas for the two displacements are:

$$D_1 = X'_1 - X$$

$$D_2 = X'_2 - X$$

As you can see in Figure 9 the displacements are simply the distance from the pixel  $X$  (which is the position of the pixel  $X$  in the destination image, but put into the source image) to the pixels  $X'_1$  and  $X'_2$

The next step is to compute the weight of the displacements. The equation for the weight of the displacement is:

$$W = \left( \frac{\text{length}^p}{a + \text{dist}} \right)^b$$

In the weight equation, the *length* is the length of the line  $P'_1Q'_1$ . The *dist* is the distance from the pixel to the line. If  $0 \leq u \leq 1$  the distance is simply  $|v|$ . However, if  $u < 0$  or  $u > 1$  then the percentage  $u$  is above or below the line  $P_iQ_i$ . When this happens the distance is no longer  $|v|$ , but instead the distance from the pixel to the appropriate endpoint of the line  $P_iQ_i$ .

The parameters  $a$ ,  $b$ , and  $c$ , in the weight equation, are constants that can be used to change the relative effect of the lines. The constant  $a$  determines the strength of the line. When  $a$  is chosen to be very close to zero, and the distance from the line to the pixel is zero then the strength of the line is nearly infinite. Choosing this value of  $a$  gives very precise control of the warp; however, a warp with a small value of  $a$  is not smooth and fluid.

The constant  $b$  controls the warping effect of each line drawn on the image. When



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 13 of 23

Go Back

Full Screen

Close

Quit

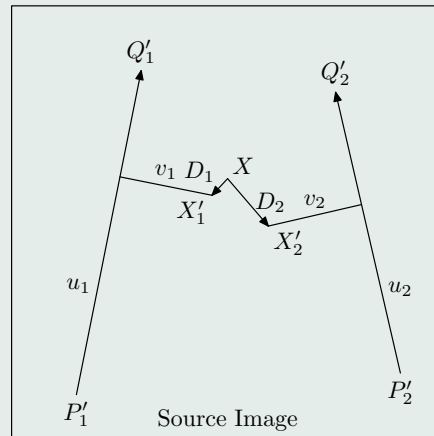


Figure 9: Calculating the Displacement



[Introduction](#)

[Objective](#)

[Mathematics behind...](#)

[Drawing Two Lines](#)

[Home Page](#)

[Title Page](#)



Page 14 of 23

[Go Back](#)

[Full Screen](#)

[Close](#)

[Quit](#)

$b = 0$  the pixel,  $X$ , will experience the same warping effect from every line in the picture. When  $b$  is large the pixel  $X$  will experience the warping effect only from the line closest to it. The best values for  $b$  are between  $[0.5 - 2]$ .

The constant  $p$  relates shorter lines to longer lines. When  $p = 0$  the lines have the same weight no matter what the length. When  $p = 1$  the longer lines have more of an effect on the warp than the shorter ones. A good value for  $p$  is 0.5.

Now, in order to get the color of the pixel  $X'$  in the source image the weighted average must be calculated and subtracted from  $X$ . Equation (6) shows this calculation. In Figure 10 you can see how the displacements play a role in finding the position of the pixel  $X'$ . This calculation must be done for every pixel in the source image ( $X'$ ) that has a corresponding pixel the destination image ( $X$ ).

$$X' = X + \frac{W_1 D_1 + W_2 D_2}{W_1 + W_2} \quad (6)$$

By applying the mathematics mentioned above we were able to write code in Matlab to produce a program that can warp images with many lines drawn on them. See Figure 11(a), Figure 11(b), Figure 12(a), and Figure 12(b) for an example of this. In this example we will draw two lines on the destination image and two lines on the source image. Notice how warping is just pixel transformation calculations. Also notice the error in pixel placement in the result of the destination image which occurs in Figure 12(a).



Introduction

Objective

Mathematics behind...

Drawing Two Lines

Home Page

Title Page



Page 15 of 23

Go Back

Full Screen

Close

Quit

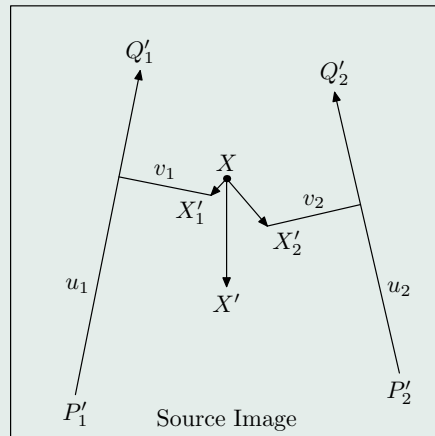


Figure 10: The position of the Pixel  $X'$



*Introduction*

*Objective*

*Mathematics behind...*

*Drawing Two Lines*

*Home Page*

*Title Page*



*Page 16 of 23*

*Go Back*

*Full Screen*

*Close*

*Quit*



(a) Destination Image

(b) Source Image

Figure 11: Warping with two lines.



*Introduction*

*Objective*

*Mathematics behind...*

*Drawing Two Lines*

*Home Page*

*Title Page*



*Page 17 of 23*

*Go Back*

*Full Screen*

*Close*

*Quit*



(a) Destination Image

(b) Source Image

Figure 12: Warping with two lines.



[Introduction](#)

[Objective](#)

[Mathematics behind...](#)

[Drawing Two Lines](#)

[Home Page](#)

[Title Page](#)



Page 18 of 23

[Go Back](#)

[Full Screen](#)

[Close](#)

[Quit](#)

## Introducing a Second Image to Produce a Morph

### Drawing Lines on Both Images

The general process to morph one digital image into another includes many steps. First a source image and a destination image must be chosen. The possibilities here are endless. We chose Rush and Squid Man as our images, however any two jpg images will do. The next step is to draw lines on the source image and the destination image. It is a good idea to frame both images with the first lines you draw. This prevents the color of any of the pixels in the source image to be calculated outside of the picture. Next lines should be drawn on the distinct features of each image as done in Figure 13(a) and Figure 13(b). The last step is to blend the two images to produce a morph.

The significance of outlining the features on each images to have the source image lines warp to the destination image lines, and to have the destination image lines warp to the source image lines. Rush's features are going to shrink down to the size of the outlined features in Squid Man's face. It also works the other way; Squid Man's features are going to grow to the outlined features in Rush's face. This should happen slowly, frame by frame, forming a sequence of images. Figure 14 shows the frame by frame sequence of Rushes original features shrinking to represent Squid Man's outlined features, while Figure 15 shows Squid Man's original features growing in order to fit Rushes original features.

To successfully morph one image into another the two sequences of images must be blended together. The forward sequence of the source image must be blended with the backwards sequence of the destination image. The middle image of the sequence must be blended so that it is 50% percent of the source image and 50% of the destination image. In Figure 16 there is no middle image, but you can see that frame number 4 is approximately 57% Rush, and 43% Squid Man; also, frame number 5 is approximately



*Introduction*

*Objective*

*Mathematics behind...*

*Drawing Two Lines*



(a) Destination Image

(b) Source Image

Figure 13: Outlining the Features.

*Home Page*

*Title Page*

◀ ▶

◀ ▶

Page 19 of 23

*Go Back*

*Full Screen*

*Close*

*Quit*



Figure 14: Rush Sequence

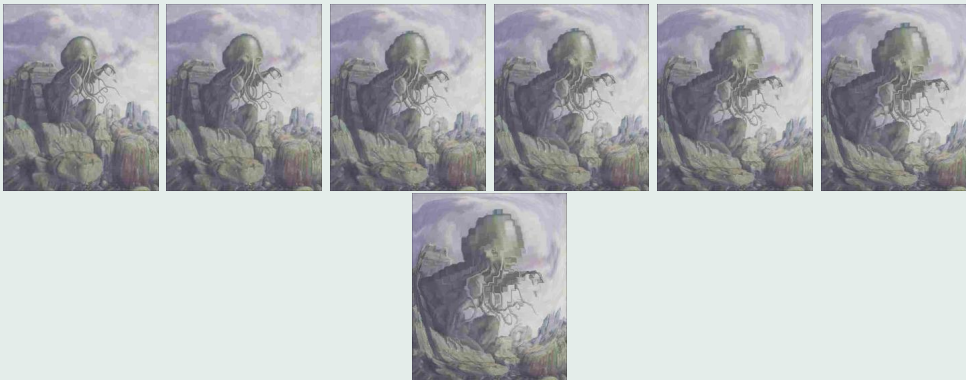


Figure 15: Squid Man Sequence



[Introduction](#)

[Objective](#)

[Mathematics behind...](#)

[Drawing Two Lines](#)

[Home Page](#)

[Title Page](#)



Page 20 of 23

[Go Back](#)

[Full Screen](#)

[Close](#)

[Quit](#)



Introduction

Objective

Mathematics behind . . .

Drawing Two Lines

Home Page

Title Page



Page 21 of 23

Go Back

Full Screen

Close

Quit

57% Squid Man, and 43% Rush. This is a successful morph of Rush to Squid Man.

## Conclusion

Now you know the Linear Algebra approach of morphing one digital image into another. This is only one of the many applications Linear Algebra has to offer. By using Linear Algebra we were able to produce a nice morph of Rush into Squid Man. Therefore, Linear Algebra proves to be a good approach for Metamorphosis. The next time you watch a movie that involves the special effect where one figure morphs into another, you will know the mathematical process involved to produce that morph.

## Appendix

When writing our Matlab code, one of the first steps we had to take was to make the source image and the destination image the same size. We did this by writing an m-file titled **MTresize**. We chose to make the larger image smaller to produce a shaper image, rather than blow up the smaller image in lose definition.

Next we wrote a function titled **MTcollectLines** in order to be able to draw lines on our source image and destination image. The ginput command in Matlab gets the  $x$ -coordinate and then the  $y$ -coordinate; however, we need the row in then the column. To do this we switched the  $x$  and the  $y$  coordinates.

In order to get the line perpendicular to the line  $\overrightarrow{PQ}$  we wrote a m-file titled **perpendicular.m**

Then we wrote the function **MTcomputePixel**. This function computes the color of  $X'$  for every pixel in the source image.

The **warpManyFrames** was our main m-file. It reads in the source image and the destination image, creates a forward sequence of the frames for the source image, and



Figure 16: Blending the Morph



*Introduction*

*Objective*

*Mathematics behind...*

*Drawing Two Lines*

*Home Page*

*Title Page*



*Page 22 of 23*

*Go Back*

*Full Screen*

*Close*

*Quit*



*Introduction*

*Objective*

*Mathematics behind . . .*

*Drawing Two Lines*

*Home Page*

*Title Page*



*Page 23 of 23*

*Go Back*

*Full Screen*

*Close*

*Quit*

creates a backwards sequence for the destination image.

Finally we wrote the function `blend.m`. This function blends the forward and backward sequences and produces the final morph.

## References

1. Arnold, David. For his Linear Algebra, Matlab, and Latex expertise.
2. Strang, Gilbert. Introduction to Linear Algebra. Wellesley-Cambridge Press, 1998.
3. Anton, Howard Rorres, Chris. Elementary Linear Algebra. John Wiley Sons, Inc., 2000.
4. Beier, Thaddeus. <http://www.hammerhead.com/thad/thad/.html>
5. Grady, Leo. <http://cas00.bu.edu/course/cs580/spring2001/lgrady/p1/>