Morphing for Intelligent Motion Picture and Video Interpolation

Bounding Distortion and Autonomous Implementation

by

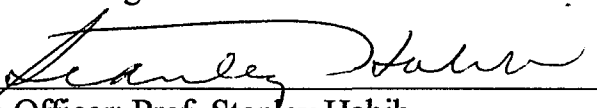
Xiaotie Zhang

A dissertation submitted to the Graduate Faculty in Computer Science in Partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

1995

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Date — Chair of Examining Committee: Prof. Charles R. Giardina



Date — Executive Officer: Prof. Stanley Habib

Prof. Michael Kress

Mr. Kenneth Davis

Prof. Dasarat Misir

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

To my wife

Hwai-yu Mary Zhang

Abstract

Morphing for Intelligent Motion Picture and Video Interpolation

by

Xiaotie Zhang

Adviser: Professor Charles R. Giardina

When morphing is used, distortion of the image is introduced by the averaging caused by multiple control lines. The deformation of a straight line segment is used as a barometer in measuring the distortion caused by the morphing process. Theorems are proved that describe the deformation on a straight line segment. Additionally in the two control lines case, a theorem is proved describing the type of curvature deformation. Sufficient conditions are also given for control line placement in order to remove the distortion effects.

A new image compression method, Morphing for Motion Interpolation(MMI) is proposed to apply morphing into motion picture interpolation with the advantages of both morphing and motion compensation. MMI method also keeps the MPEG features such as Random Access, etc. Two versions of MMI encoder and decoder are also given.

Acknowledgement

I would like to express my sincere gratitude and appreciation to my advisor Prof. Charles R. Giardina for his support, encouragement, and guidance throughout my graduate study.

I like to thank my supervisor committee members, Prof. Charles R. Giardina, Prof. Stanley Habib, Prof. Michael Kress, Mr. Kenneth Davis and Prof. Dasarat Misir, for their valuable suggestion and generous support.

I also like to thank the Department of Computer Science at the College of Staten Island/CUNY for the graduate assistantship, the graduate school of the City University of New York for the university fellowship. Both of them provided me an important financial support.

Specially, I wish to express my appreciation to Prof. George Jochnowitz who made it possible for me to study in America and helped me when I was new in this country.

Finally, I want to thank my parents and my wife, Hwai-yu, for their understanding and encouragement.

Table of Contents

Table of Figures

Chapter 1 Introduction

In recent years, several image processing techniques, among them morphing has achieved widespread use in the entertainment industry. Morphing involves the transformation of one 2D image into another 2D image. The idea is to specify a warp that distorts the first image into the second image. As the morphing proceeds, the first image is gradually distorted and is faded out, while the second image starts out totally distorted relative to the first and is faded in. Thus, the early images in the sequence are much like the first image. Middle images of the sequence are weighted averages of the first and last image. For morphing between human faces, the middle image often looks strikingly life-like, like a real person, but it is neither the image of the person in the first nor the second images.

Morphing techniques involve first specifying some function that maps points from one image onto points of the other image, then simultaneously interpolating the color and the position of corresponding points to

generate intermediate images. When viewed in sequence, these intermediate images produce an animation of the first image changing into the second. Variations of these techniques have been used to create astonishing special effects for commercials, music videos, and movies.

In telecommunications, visual communication is becoming a rapidly growing strategic management tool. Since economical long-distance video conferencing is feasible only with digital transmission, the video codec plays an essential role in determining service quality and compatibility among different system. To provide a full motion video signals at a reasonable transmission cost an efficient data compression technique is necessary. For lossy video compression algorithms, Motion Picture Expert Group(MPEG) is underway with efforts to standardize a video-coding algorithm for interactive applications with digital storage media. The bit-rate for the compressed video, which is limited by the medium speed, is currently set by MPEG at 1.5Mbps (bits per sec).

The MPEG algorithm is based on H.261 model, which is video codec for audiovisual services at Px64 kb/s, p is a number between 1 and 30, developed for the video conferencing application by a CCITT subgroup this model is referred to as the CCITT_RM [52]. The CCITT-RM is essentially a motion-compensated inter-frame predictive coding technique.

Image compression and film making have similar methods for motion picture processing. Seamless transition between defined aspects of pictures can be achieved in a number of ways. Rapid presentation of frames is usually sufficient to fool the visual system into perceiving continuous, smooth movement. As the discrepancy between individual frames increases, so must the sampling frequency and the subsequent representation rates. Morphing methods exploit artificially creating intervening frames, the number of which dictates the final transition's apparent fluidity. Using either film recording or computer blending, one image is gradually faded out while the presence of the final frame becomes more apparent. The rate at which one key frame

disappears and another is incorporated is normally determined in a linear manner.

The MPEG compression algorithm is derived in so far as it satisfied the required quality and random access requirements in motion videos. The algorithm is based on two types of coding, namely intra-frame and inter-frame, corresponding to the statistical and visual characteristics of a single image and an image sequence respectively. Pure intra-frame coding could satisfy the demand of random access, but the achievable compression rate is limited. Inter-frame coding does result in much higher compression rate, but at an expense of a more restricted degree of random access and quality. The major approach of the MPEG algorithm is to achieve a delicate balance between intra- and inter-frame coding, between recursive and non-recursive temporal redundancy reduction.[47]

Inter-frame Coding

Much of the information in a picture within a video

sequence is similar to information in a previous or subsequent picture. The MPEG algorithm takes advantage of this temporal redundancy to represent some pictures in terms of their difference from a reference. Two inter-frame coding techniques, predictive and interpolative, have been proposed by MPEG. Video frames are coded as three types of pictures: (a) Intra Picture(I), coded using only information present in the picture itself; (b) Predicted Pictures (P), coded with respect to the nearest previous I- or P-pictures. This technique is called forward prediction and is shown in Figure 1.1(A); and (c) Interpolated Picture (B-Bidirectional Prediction) coded with both past and future pictures as reference. This technique is called bidirectional prediction and is shown in Figure 1.1(B).[47]

A typical arrangement of I-, P- and B-pictures is shown in Figure 1.2 in the order they are displayed.

Motion compensation is also used in MPEG for enhancing the compression of P- and B-pictures by eliminating temporal redundancy. Motion compensation

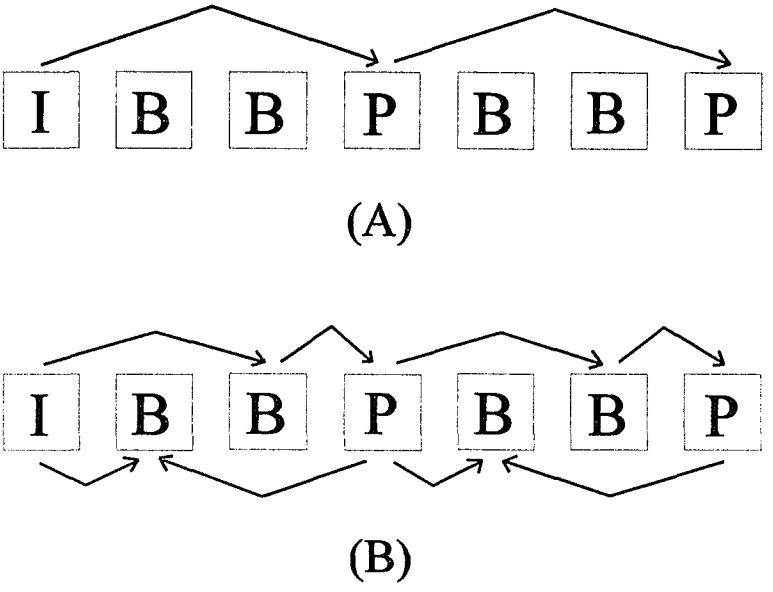


Figure 1.1 (A) Forward Prediction
(B) Bidirectional Prediction

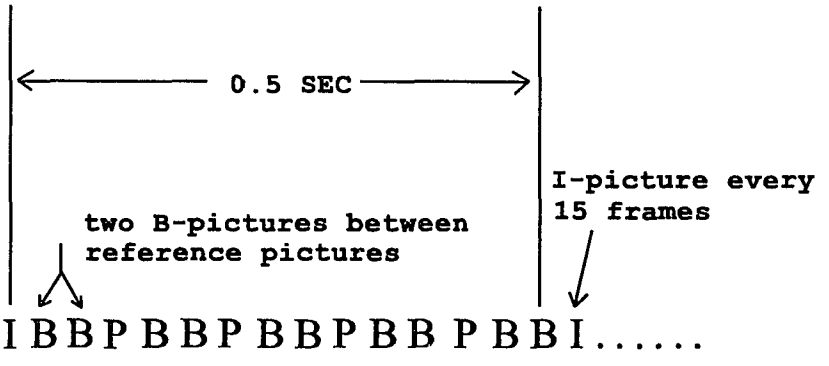


Figure 1.2 Typical Display Order of Picture Types

improves compression by about a factor of three compared to intra-picture coding.

For P-pictures, the amplitude and the direction of the displacement at some previous frames are coded as part of the necessary information to recover the picture.

For B-pictures, motion information is coded using either a previous or future reference picture as a reference. Full-resolution signal is reconstructed by interpolation of the past and future low-resolution signal with a correction term.

Intra-frame Coding

With the experience of still image coding in JPEG, the 8x8 DCT is chosen to perform further compression to both the remaining still image (intra-pictures) and prediction error signals which have very high spatial redundancy. Similar to JPEG, the encoding process essentially consists of three stages: (a) DCT transform, (b) quantization and (c) entropy coding.

In morphing, the special effect that one image gradually change into another, is achieved by interpolating a sequence of image frames. If the second image contains the same objects as the first one but moved, morphing interpolates the intermediate image frames that accomplish the motion when all the image frames playback sequentially. The interpolated image frames are generated by mathematical equations with positions of control lines and the first image, or the first and second images. No intermediate image information is needed. In MPEG, one group of frames(GOF) contains one intra-coded frame, four predicted frame, and ten bidirectional interpolated frames. Fourteen frames out of fifteen (P and B frames) are motion compensation(MC) coded interpolation frames. In MC-coded frames, the information of the motion-field is needed in order to do the coding. For fast motion video, there is a huge amount of information that MPEG needs to code and decode.

In this study, a new image compression method, morphing for motion interpolation(MMI) is presented with

the image interpolation advantages of both morphing and motion compensation. MMI methods have the advantage of motion compensation in which the non-motion fields of images are not coded to reduce redundancy. Here the DCT transform, quantization and entropy coding are needed only for I(intra-frames). The Block Matching Algorithm(BMA), which is needed for estimating the motion vector, is omitted in this method. With the MMI method, none of P and B frames need to be coded, only control line information utilized in morphing is needed in each I frame. The MMI methods do not alter MPEG features such as Random Access, and Reverse Playback, Fast Forward/Reverse Searches, etc.

When morphing is applied in image interpolation, distortion of the image is introduced by the averaging caused by multiple control lines. Thus a straight line may bend due to the morphing process. In this study, the deformation of a straight line segment is used as a barometer in measuring the distortion caused by the morphing process. Sufficient conditions are also given for control line placement in order to remove these

distortion effects.

Chapter 2 introduces the concept of morphing, it illustrates applications within entertainment and commercial areas, and illustrates different morphing methods. Chapter 3 presents the mathematical background underlying the morphing transform. Here a detail description is given showing how different morphing methods generate intermediate images. It also provides the general averaging equations for morphing. Chapter 4 defines the morphing transform of using one control line and multiple control lines, several theorems on morphing distortion are proved in the one control line case and in the multiple control line case. In the one control line case, it is shown that the morphing transform has no distortion based on the straight line segment, moreover it preserves parallel straight lines. In the multiple control line case, theorems are proved that describe the deformation on a straight line segment. Here a metric is introduced to estimate and bound the curvature of the deformed line segment. Additionally in the two control line case, a theorem is proved providing conditions for

the morphing transform to preserve straight line segments. Another theorem is proved describing the type of curvature deformation of a morphed line segment.

In chapter 5, issues related to the compression of motion video are described. MPEG requirements and its implications are given along with the standard MPEG compression algorithms which also provide the basis for the MMI method. This includes motion compensated prediction, and motion compensated interpolation, DCT transform, quantization, and entropy coding. Chapter 6 describes the MMI method and considers its advantages as well as how its methodology complies with the MPEG requirement. The structure of the encoder and decoder are presented for the proposed method.

Chapter 2 Introduction to Morphing

Morphing is short for metamorphosing. The word is derived from biology, where it describes nature's wonderful magic trick of turning a caterpillar into butterfly or a tadpole into a frog. Morphing depicts objects that smoothly change shape.

A typical application of morphing can be found in the movie Terminator 2: Judgment Day. In this movie there is a well-known scene in which the T1000 terminator is transformed into a puddle of liquid metal on a checkerboard floor. The puddle suddenly begins oozing upward, defying gravity, and forms the general outline of a man. The liquid metal then solidifies to a humanoid shape, and "grows" skin to become the clearly defined image of man. Morphing as a computer graphics technique plays astonishing visual tricks to our eyes. Police officers become people they touch. Arms turn into swords and vise versa. A car changes into a running tiger. Morphing smoothly dissolves between two or more images, so that one appears to become the other.

Morphing has made a profound impact on public visual media, and has become an important computer graphics technique at Industrial Light and Magic(ILM). Morphing was first introduced in the 1988 film Willow. When director Ron Howard wanted a sorceress to change into different animal forms, he turned to ILM special effects studio. Since then, he has performed morphing for several commercials and the movie Terminator 2: Judgment Day.[20]

More recently, it has been possible to capture the reference frames on a digital computer and numerically dissolve to the next frame using pixel point-to-point interpolation. The transition sequence must then be transferred to film and spliced into the correct position in the sequence. The results are artistically pleasing, and can be entertaining provided care is taken.

In image processing, morphing algorithms gradually change the first image into the second image, with interpolating a number of intermediate image frames. The first image is called the source image; the second image is called the target image. Morphing is a term

describing the procedure that makes the source image continuously "transform" into the target image. Morphing uses a computer to calculate and draw the image frames intermediately between the source and the target images. When all of the image frames playback, the source image will(gradually change) morph into the target image, like a man into a girl.

The following is an simple example showing a horizontal line segment morphing into a vertical line segment. Figure 2.1 shows the source image frame and target image frame.

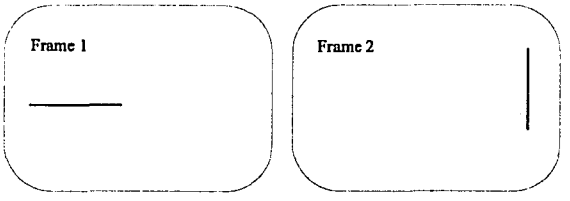


Figure 2.1 A horizontal and a vertical line segments.

Figure 2.2 shows how a straight line segment is interpolated.

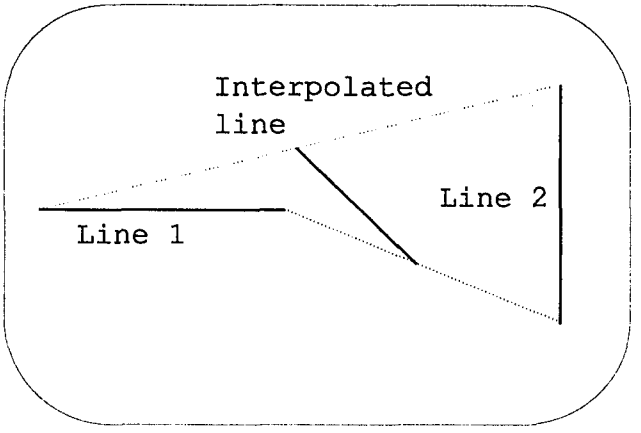


Figure 2.2 One interpolated line segment connecting the mid-points.

Subdividing the line again in Figure 2.2, another line segment can be interpolated in each side of the middle line segment.

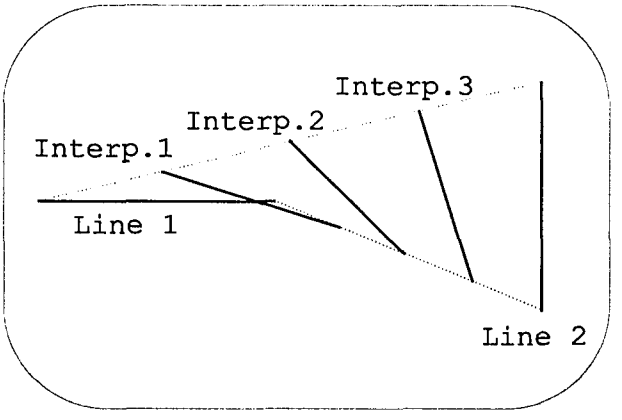


Figure 2.3 Two more interpolated line segments.

Now there are three line segments between frame one and frame two. More line segments can be interpolated along the path to assure smooth motion. If the frames are played one by one, from source image to target image, it appears that a horizontal line segment morphs into a vertical line segment.

Morphing can also be described as a shape change applied to one or more images. There are two types of morphing, distortion morphing and transition morphing. Distortion morphing distorts a single image into a new image using positional control information, the control points or control lines. It is also called warp morphing, or warping. In distortion morphing, the shape of the original image changes without fading to a second image. Figure 2.4 shows the original image, Figure 2.5 shows the new warped image. Distortion morph generate the new image according to the movement of control lines.

Transition morphing combines distortions of the source image and the target image to form a new image by using positional control information, as done in warping.

Figure 2.4 Original image

Figure 2.5 Warped image

This is the same as distortion morphing, along with the usage of color information from both the source image and the target image. Transition morphing is the classical morphing method, it warps from the source image to the target image. This is also the same as distortion morphing in which the control lines are automatically determined by their positions in the source and the target images. Transition morphing then distorts the images with color information of the source and the target images. This is also called image dissolving. If the source image and the target image contain different objects, transition morphing changes one object to the other. Figure 2.6 - 2.9 show how transition morphing

Figure 2.6 Source image.

Figure 2.7 Morphed image 1.

Figure 2.8 Morphed image 2.

Figure 2.9 Target image.

changes a clown into a cheetah.

One way to understand the warping (distortion morphing) is to imagine an image printed on a sheet of rubber. Image warping has the same effect on an image as stretching and pinching the rubber sheet at interesting places. For example, a map of the world is printed on a rubber sheet, and New York is pulled south. Computers can perform this operation in a simple way. The computer

screen consists of a map of colored pixels, which can be manipulated or transformed using mathematical equations. The process in general is called a "mapping". When the image stays connected through the use of the transformation, it is then called warping. In most warping algorithms, the warping is local, so if New York is pulled south, Newark being close to New York moves with it, but Chicago barely budges. The amount of movement is specified and controlled using designated points.

The simplest method is to indicate points on the screen, which should be moved in a certain way, and to drag these points to these final locations.[2] These points are called control points. Although relatively simple to program, this method poses difficulties for the user. Hundreds of points may be needed for effective warping. These points must also be kept track of. If a single point is misplaced, the warp will be wrong.

A better method for warping is to use line segments (here in called lines or control lines). With just the

two end-points, a line automatically specifies a string of points in between. Another benefit of this implementation is not as obvious: with lines, you can specify global changes. With one line, you can rotate or scale the entire image. This technique is much more powerful than local warping.[1]

When there is more than one line, they each compete for influence, and the effect becomes more local.

In a control line warping algorithm, such as the ones discussed in the following chapters, the same number of control line segment will be placed in the source and the target images. In each pair of control line segments, the control line in the source image will be transformed into a corresponding control line in the target image. For pixel-mapped images, the pixels on each control lines are preserved from the source image to the target image. The other pixels in the plane are formed by averaging the influence of each control line.

The transformation of control line segments from the

source to the target image may include translating, rotating and scaling of the control line segment. The control line segments in the intermediate image frames change gradually from the position in source image to the position in target image.

In general, morphing is an image processing technique typically used as an animation tool for the metamorphosis of one image into another.

The method for changing one digital image into another is to use dissolved method. The color of each pixel is interpolated over time from the source image value to the corresponding target image value. while this method is more flexible than the traditional optical approach (for example, different dissolve rates in different image areas), it is still often ineffective for suggesting the actual morphing from one subject to another. This may be partially due to the fact that we are accustomed to seeing this visual device used for another purpose: the linking of two shots, usually signifying a lapse of time and a change in place.

Chapter 3 Mathematical Fundamentals of Morphing

In this chapter, the fundamentals of the mathematics underlying morphing of images are studied by considering the representation and transformation of points and straight line segments.

There are two parts involved in the morphing procedure, image warping and image dissolving. Distortion morphing has only image warping, transition morphing has both image warping and image dissolving.

In analysis of image warping, an oriented line segment coordinate system, shown in Figure 3.1, is

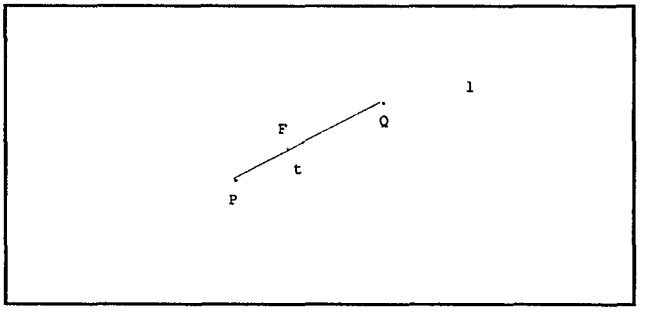


Figure 3.1 A straight line PQ.

applied to represent the relation between a point to a certain line segment. Given line segment PQ in RxR with direction from P to Q. Extending the line PQ to line l, given by

$$F(t) = tQ + (1-t)P, \qquad t \text{ is real.}$$

W is a point on the plane. Find the distance d between W and line PQ, WH $\perp$ PQ. Find the fractional distance *a* of W on PQ, $a = \frac{PH}{PQ}$. Figure 3.2 shows that W is uniquely determined by *a* and *d*. *d* is the distance from the line, and *a* is the fractional distance along the line. The value *a* goes from 0 to 1 as the point move from P to Q, and is less than 0 or greater than 1 outside of that range. The value for d is the perpendicular distance in pixels from the line.

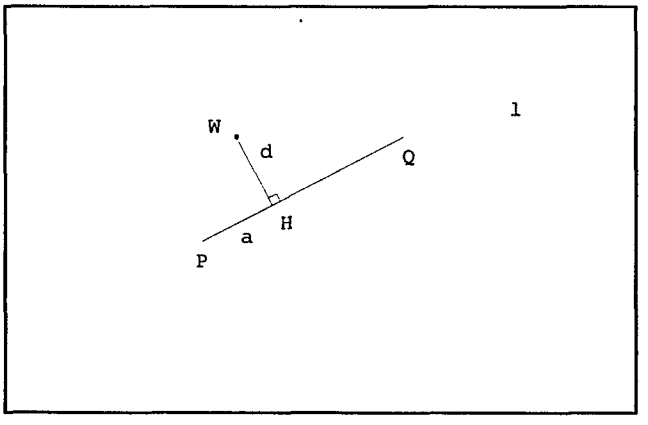


Figure 3.2 Any point can be uniquely determined by PQ.

There are six distinct regions for W, shown in Figure 3.3 depending upon values of *a* and *d*.

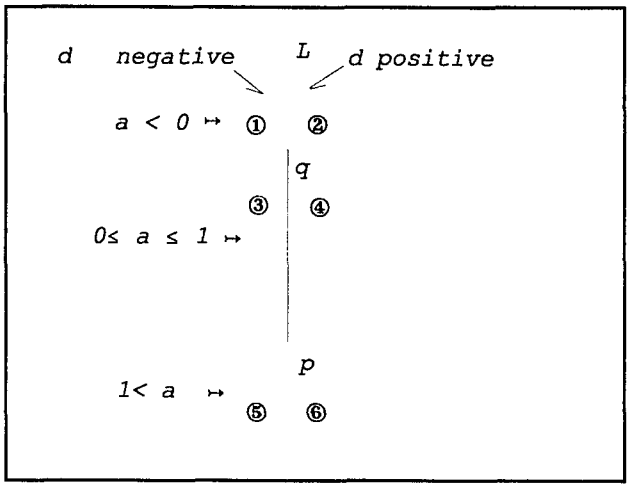


Figure 3.3 six distinct regions

This reference line segment PQ will be placed in a image. Every pixel of the image has an unique value pair, *a* and *d*. This line segment is called a control

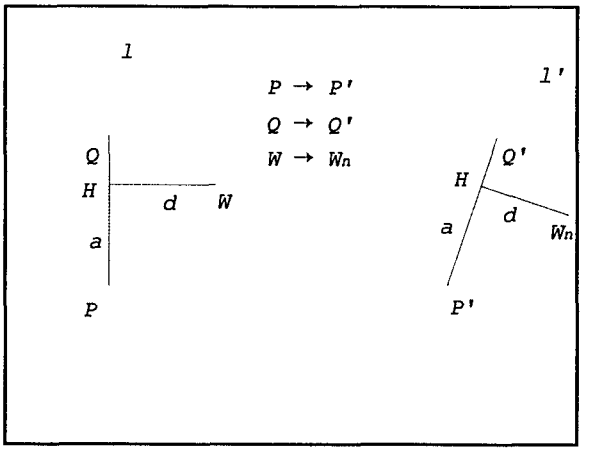


Figure 3.4 One control line transformation

line. When a morphing transformation, shown in figure 3.4, mapping the control line *l* into *l'* is employed, the values *a* and *d* stay fixed for all points W in RxR.

If there is just one control line in the image, the transformation of the image proceeds as follows.

Position the control line in the target image.

For each pixel W in the source image

find the corresponding value pair *a* and *d*

find the position of W_n in the target image

using the same pair *a* and *d*.

targetImage(W_n) = *sourceImage(W)*.

In Figure 3.4, W_n is the location in the target image for the pixel at W in the source image. The location is at a distance *d* (the distance from the pixel to the control line in the source image)to the line P'Q', and at a proportion *a* along that control line.

The algorithm transforms each pixel coordinate by translation, rotation, and scale, thereby transforming the whole image. All of the pixels along the control

line in the source image are copied on top of the control line in the target image. The *a* coordinate is normalized by the length of the control line, however, the *d* coordinate is not (since it is always specified as a distance in pixels). Consequently the resulting image is scaled along the direction of the given control lines by the ratio of the lengths of the control lines. The scaling is only along the direction of the line.[5]

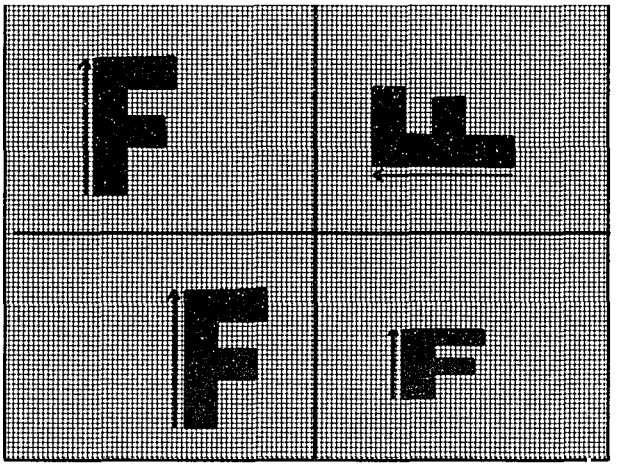


Figure 3.5 one control line example.

In figure 3.5, the figure on the upper left is the original image. The control line is translated in the lower left image, rotated in the upper right image, and scaled in the lower right image, performing the corresponding transformations to the image.

When multiple control lines are applied to the image, for each point W, multiple sets of a_i (fractional distance of the control line P_iQ_i), and d_i (the distance to the control line P_iQ_i) are needed. These various pairs represent the point W relative to the distinct control lines. Figure 3.6 shows three control lines influencing the point W.

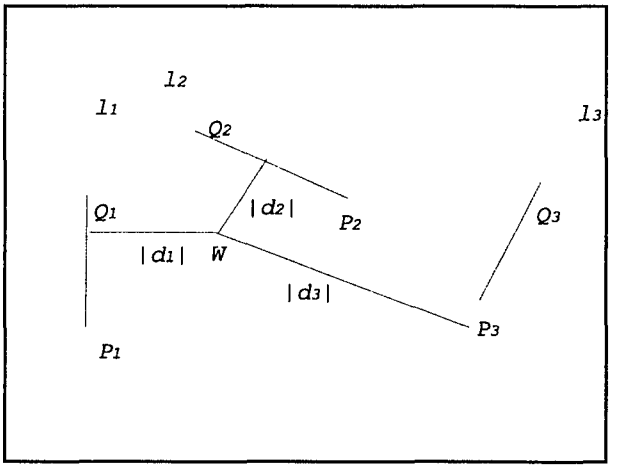


Figure 3.6 Three control lines influencing W.

Multiple control lines specify more complex transformations. Figure 3.7 shows two control lines applied in an image.

If there are *m* control lines, a weighting of the coordinate transformations for each control line is performed. A position W_{ni} is calculated for each control

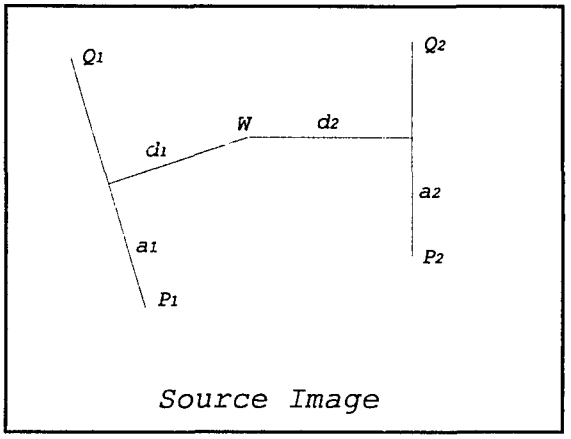


Figure 3.7 Source image with multiple control lines.

line. W_{ni} is the new calculated image point for control line *i*, and there are *m* image point for W. The displacement

$$D_i = W_{ni} - W \qquad i = 1, 2, \ldots, m \qquad (3.1)$$

is the difference between the pixel location in the source and target images, and a weighted average of those displacements is calculated. The weight is determined by the distance from W to the control line. This average displacement is added to the current pixel location W to determine the position W_n in the target image. The single control line case is a special case of the multiple line case, assuming the weights never go to zero. In general, the weight assigned to each control line is strongest when the pixel is exactly on that line,

and weaker the further the pixel is from the line. The equation for weighting[5] is

$$weight = M_i = \left(\frac{length_i^{\,p}}{(a + dist_i)} \right)^b \qquad (3.2)$$

where $length_i$ is the length of control line i, $dist_i$ is the distance from the pixel to the control line, here $dist_i = d$, and a, b, and p are constants that can be used to change the relative effect of the control lines.

The weighting factors are based on how far the original point is from each control line, and how long

$$W_n = W + \frac{\sum_{i=1}^{m} D_i M_i}{\sum_{i=1}^{m} M_i} \qquad (3.3)$$

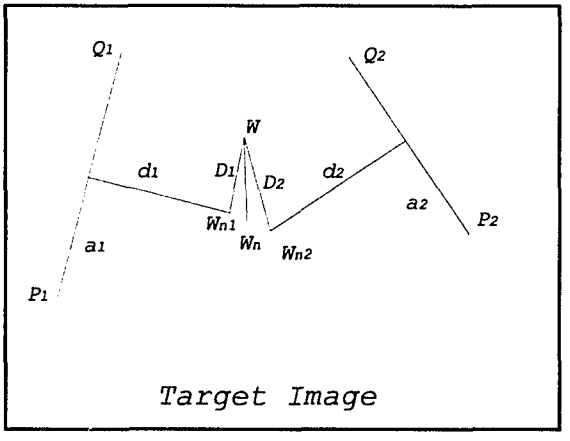


Figure 3.8 Target image with multiple control lines.

the control line is. Using this factor, the weighted average of m displacements is calculated to arrive at the final position W_n.

Figure 3.8 shows the averaged new point W_n

$$W_n = \frac{W\sum_{i=1}^{m} M_i + \sum_{i=1}^{m} D_i M_i}{\sum_{i=1}^{m} M_i} = \frac{\sum_{i=1}^{m} (W + D_i) M_i}{\sum_{i=1}^{m} M_i}$$

$$W_n = \frac{\sum_{i=1}^{m} W_{ni} M_i}{\sum_{i=1}^{m} M_i} \qquad (3.4)$$

Figure 3.9 shows transformed image point averaged by equation (3.4).

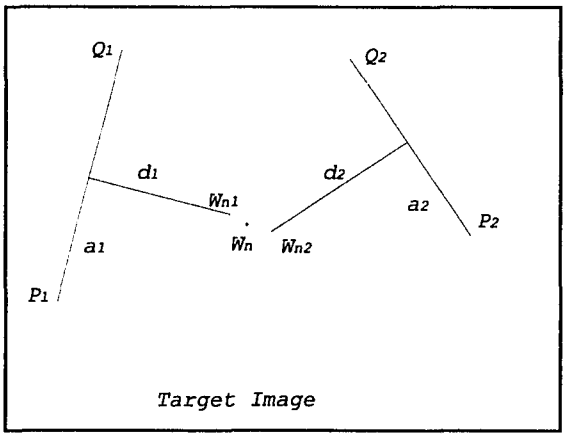


Figure 3.9 Target image with multiple control lines.

If *a* is barely greater than zero, then if the distance from the line to the pixel is zero, the strength is nearly infinite. With this value for *a*, the user will be assured that pixels on the control line will moves along with the control line. Larger values will yield a more smooth warping, but with less precise control. The variable *b* determines how the relative strength of different control lines falls off with distance. If it is large, then every pixel will be affected essentially only by this control line nearest it. If *b* is zero, then each pixel will be affected by all control lines equally. Values of *b* in the range [0.5, 2] are the most useful. The value of *p* is typically in the range [0, 1]; if it is zero, then all lines have the same weight, if it is one, then longer control lines have a greater relative weight than shorter control lines.[5]

The multiple line algorithm is as follows:

For each pixel W in the source image

DSUM = (0, 0)

weightsum = 0

For each control line P_iQ_i

calculate *a*, *d* based on P_iQ_i

calculate W_{ni} based on *a*, *d* and $P_i'Q_i'$

length = calculate the length of P_iQ_i

weight = $(length^p/(a+d))^b$

DSUM += W_{ni}* *weight*

weightsum += *weight*

W_n = *DSUM/weightsum*

targetImage(W_n) = *sourceImage*(*W*)

There are two ways to warp an image[5], forward mapping and reverse mapping. In forward mapping, warping scans through the source image pixel by pixel, and copies these pixels to the appropriate place in the target image. In this case, some pixels in the target image might get painted, and would have to be interpolated. In reverse mapping, warping goes through the target image pixel by pixel, and samples the correct pixel from the source image. The most important feature of inverse mapping is that every pixel in the target image is set to something appropriate. For reverse mapping, warping becomes such problem that which pixels in the source

image are sampled for the one pixel in the target image. The averaging equations for both forward and reverse mapping are the same. Next, the algorithms for reverse mapping are presented.

In one control line reverse mapping, shown in Figure 3.10, transformation of the image proceeds as follows.

For each pixel W_n in the target image

find the corresponding *a*, *d*

find the position of *W* in the source

image for that *a*, *d*.

targetImage(W_n) = *sourceImage*(*W*).

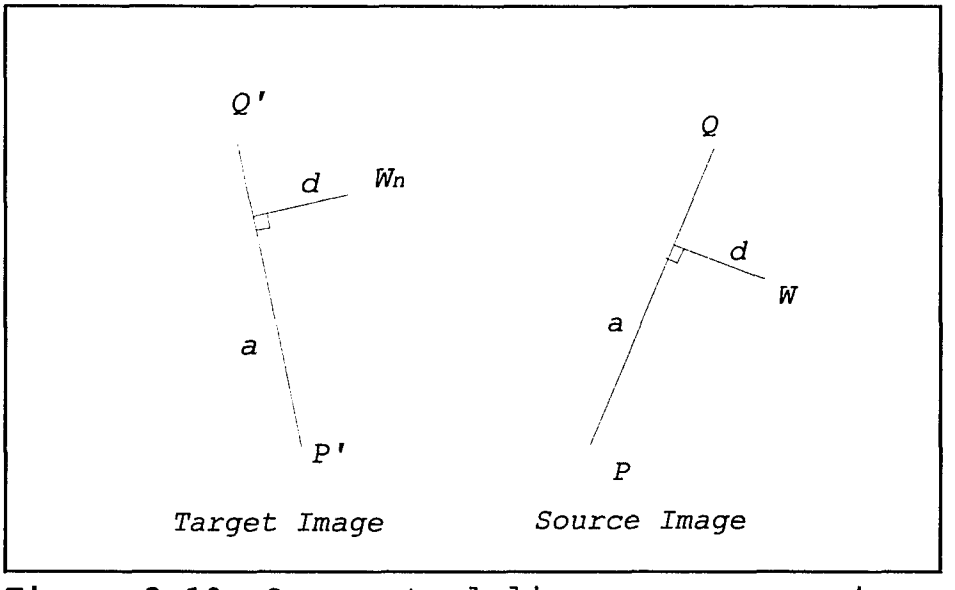


Figure 3.10 One control line reverse mapping

The multiple line reverse mapping algorithm, shown in Figure 3.11 and Figure 3.12 is as follows:

For each pixel W_n in the target image

DSUM = (0, 0)

weightsum = 0

For each control line $P_i'Q_i'$

calculate *a*, *d* based on $P_i'Q_i'$

calculate W_{ni} based on *a*, *d* and P_iQ_i

length = calculate the length of P_iQ_i

weight = $(length^p/(a+d))^b$

DSUM += W_{ni}* *weight*

weightsum += *weight*

W = *DSUM/weightsum*

targetImage(W_n) = *sourceImage*(*W*)

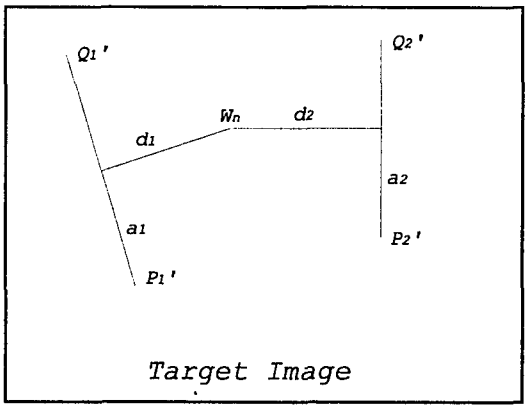


Figure 3.11 Target image with multiple control lines.

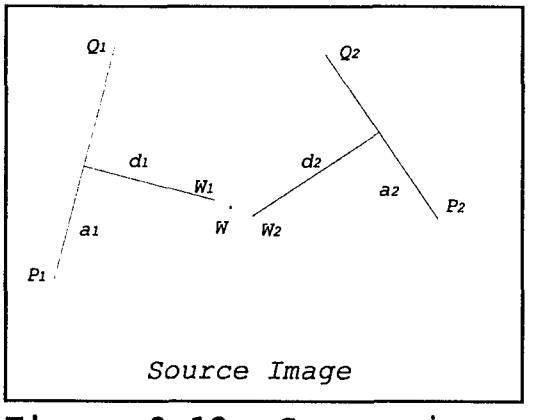


Figure 3.12 Source image with multiple control lines.

In the book [1], the weighting

$$weight_i = M_i = \frac{1}{d_i^2} \qquad (3.5)$$

is used in averaging. This is the same as in equation (3.2).

$a = 0, \quad b = 2, \quad$ and $p = 0$

Location of transformed point W_n will be weighted average of individual transformed points W_{ni} with weights M_i. If $d_i = 0$ use $W_n = W_{ni}$ otherwise use

$$M_i = \frac{1}{d_i^2} \qquad (3.7)$$

$$W_n = \frac{\sum_{i=0}^{m} M_i W_{ni}}{\sum_{i=0}^{m} M_i} \qquad (3.6)$$

These averaging formulas for multiple control lines are also utilized through out this study. The control lines do not cross each other is also assumed.

Image dissolving is also called cross-dissolving,

which means to blend colors together. The weighted averages of the primary color components are computed as follow.

$$r = \frac{m_1 r_1 + m_2 r_2}{m_1 + m_2} \qquad (3.8)$$

$$g = \frac{m_1 g_1 + m_2 g_2}{m_1 + m_2} \qquad (3.9)$$

$$b = \frac{m_1 b_1 + m_2 b_2}{m_1 + m_2} \qquad (3.10)$$

Where r_1 g_1 b_1 and r_2 g_2 b_2 are the primary colors for the first and second pixels and m_1 and m_2 are the weighting factors for each pixel. For example, with a color 20 percent of the way between two others, m_1 is 20, m_2 is 80 and their sum is 100.

If the weighting for the first pixel is zero, the color takes the value of the second pixel. Similarly, if the second weight factor is zero, the first pixel is selected.

The luminance Y and chrominance u, v(Cr, Cb) can

also be used as color components in dissolving. If this is done;

$$Y = \frac{m_1 Y_1 + m_2 Y_2}{m_1 + m_2} \qquad (3.11)$$

$$u = \frac{m_1 u_1 + m_2 u_2}{m_1 + m_2} \qquad (3.12)$$

$$v = \frac{m_1 v_1 + m_2 v_2}{m_1 + m_2} \qquad (3.13)$$

Chapter 4 Morphing Transform of Straight Lines

This chapter is dedicated to investigating how an arbitrary straight line segment is mapped by the morphing transform. In particular distort weighting using the averaging equations specified in the previous chapter a straight line often becomes "curved". In this chapter bounds are derived on the "curvature".

4.1. Representation of Line Segments

In the previous chapter, a point A can be determined by a control line (control line segment) and two parameters, the fractional distance a and the distance c.

Definition 4.1 PQ is a control line segment, A is a point on the plane. The length of PQ is l. Then the relation between the point A and the control line PQ can be represented as $A_{\rightarrow PQ}\{a, c\}$ or $A\{a, c\}$. a is the fractional distance of control line PQ, $|a| = \frac{PF}{l}$ (see Figure 4.1). $a < 0$ when F is outside of PQ on the

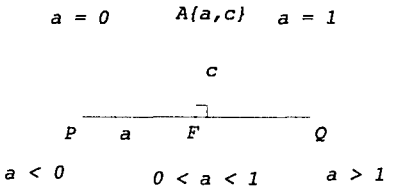


Figure 4.1 Point A's position to control line PQ.

P side; $0 < a < 1$, when *F* is between *PQ*; and $a > 1$ when *F* is outside of *PQ* on the *Q* side. $|c|$ is the distance between A and control line PQ. When the line segment PQ (from P to Q) points to right, *c* is positive if *c* is above PQ; *c* is negative if *c* is below PQ. When the line segment PQ (from P to Q) points to left, *c* is positive if *c* is below PQ; *c* is negative if *c* is above PQ.

Since any point can be uniquely determined by a control line, a straight line segment can also be determined by the control line shown in Figure 4.2. A{*a*, *c*}, B{*a*+*b*, *c*+*d*} are two end points of a line segment. Any point W on line segment AB can be represented by

$$W\{a+bt,\ c+dt\} \qquad 0 \le t \le 1$$

$W\{a+bt,\ c+dt\}$ represents a line segment AB.

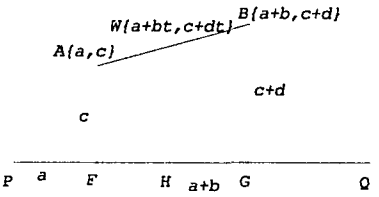


Figure 4.2 Straight line with single control line

If $-\infty < t < +\infty$, it represents a straight line. If W needs to specify to the control line PQ, the following representation may be used.

$$W_{\rightarrow PQ}\{a+bt,\ c+dt\} \qquad 0 \le t \le 1$$

In XY-coordinate system, PQ is a control line with $P(x_p,\ y_p)$, $Q(x_q,\ y_q)$. The length of PQ is l, $l=\sqrt{(x_q-x_p)^2+(y_q-y_p)^2}$. AB is a line segment with $A_{\rightarrow PQ}\{a,\ c\}$ and $B_{\rightarrow PQ}\{a+b,\ c+d\}$. The position of point A and B, in XY-coordinate system are illustrate below.

The position of point A:

$$x_A = x_p + a(x_q-x_p) - \frac{c}{l}(y_q-y_p)$$

$$y_A = y_p + a(y_q - y_p) + \frac{c}{l}(x_q - x_p) \qquad (4.1)$$

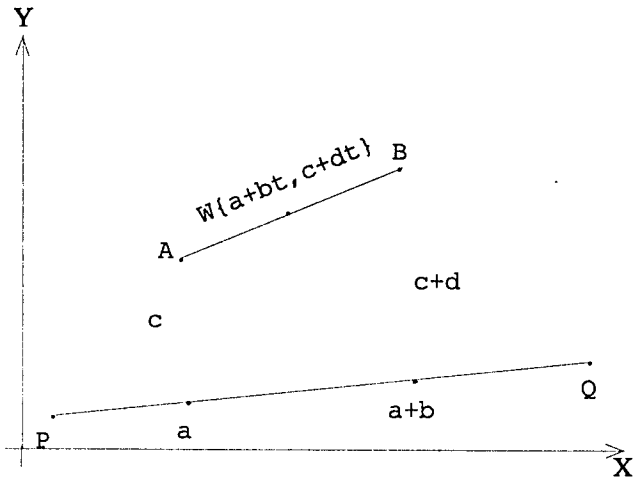


Figure 4.3 Line segment and control line in XY-coordinate system

The position of point B:

$$x_B = x_p + (a+b)(x_q - x_p) - \frac{c+d}{l}(y_q - y_p)$$

$$y_B = y_p + (a+b)(y_q - y_p) + \frac{c+d}{l}(x_q - x_p) \qquad (4.2)$$

Any point on AB, $W\{a+bt,\ c+dt\}$, $0 \le t \le 1$, can be expressed as

$$x_W = x_p + (a+bt)(x_q - x_p) - \frac{c+dt}{l}(y_q - y_p)$$

$$y_W = y_p + (a+bt)(y_q - y_p) + \frac{c+dt}{l}(x_q - x_p) \qquad (4.3)$$

W is function of t. When t changes from 0 to 1, W has a locus producing a straight line segment from A to B. The above representation will be used throughout the study to represent a line segment AB, when the position of the control line PQ and the relation between AB and PQ are known.

4.2. One Control Line Case

The one control line case is simplest and provides a basic use of the morphing transform. It is also important for analyzing the multiple control line cases. In the one control line case, let PQ be the control line, then any point A on the plane can be uniquely determined by PQ and its relation with PQ. In Figure 4.1, this is $A_{PQ}\{a, c\}$.

Definition 4.2 One control line morphing transform is defined as when the single control line PQ is translated, rotated, or (nonzero)scaled to become a new control line P'Q'. However, point $A_{PQ}\{a, c\}$ on the plane will maintain its relative relation to the new

control line P'Q', that is $A_{n\to P'Q'}\{a, c\}$. A_n is the image of A after morphing transform.

One control line morphing transform changes the position, orientation and length of a control line, but it preserves the relation of any point on the plane to this control line. It also preserves the properties that a straight line segment maps into a straight line segment, and parallel straight lines maps into parallel straight lines. The next two simple theorems and their proofs are given for completeness and to establish notation used in subsequent theorems.

Theorem 4.3 One control line morphing transform maps a straight line into a straight line, and consequently maps a straight line segment into a straight line segment.

Proof:

Let PQ, P'Q' be the control lines before and after morphing transform, respectively. AB is a line segment on the plane. W is any point on AB, $W_{\to PQ}\{a+bt, c+dt\}$, $0 \le t \le 1$. A'B' is the line segment after morphing

transform the control line PQ to the new control line P'Q'. W_n is the image of W after the morphing transform. Since the morphing transform preserves the relation between W and PQ then W_n will have the same relation to the control line P'Q', $W_{n\to P'Q'}\{a+bt,\ c+dt\}$.

In Figure 4.4(a), PQ is the control line with $A\{a,c\}$ and $B\{a+b,\ c+d\}$. AB is a straight line segment. For any given t, $0 \le t \le 1$, W is a point on AB and $W\{a+bt,\ c+dt\}$.

After the morphing transform is applied, in Figure 4(b), P'Q' is the new control line with $P'(x_{p'},\ y_{p'})$ and $Q'(x_{q'},\ y_{q'})$. The length of P'Q' is $l'=\sqrt{(x_{q'}-x_{p'})^2+(y_{q'}-y_{p'})^2}$. W{a+bt, c+dt} maps to $W_{n\text{-}p'q'}\{a+bt,\ c+dt\}$. $a,\ b,\ c,\ d$ remain unchanged, according to the definition of the morphing transform.

The position of $W_n(x_n,\ y_n)$ is

$$x_n = x_{p'} + (a+bt)(x_{q'}-x_{p'}) - \frac{c+dt}{l'}(y_{q'}-y_{p'})$$

$$y_n = y_{p'} + (a+bt)(y_{q'}-y_{p'}) + \frac{c+dt}{l'}(x_{q'}-x_{p'}) \qquad (4.4)$$

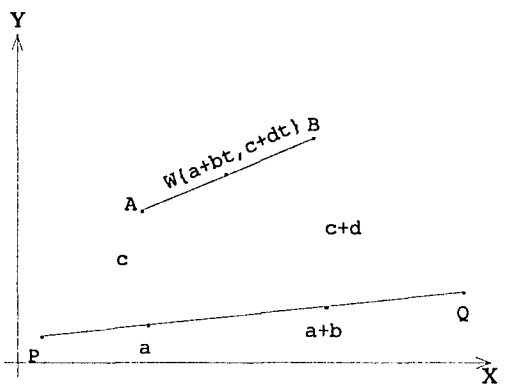


Figure 4.4(a) Line segment AB before transform

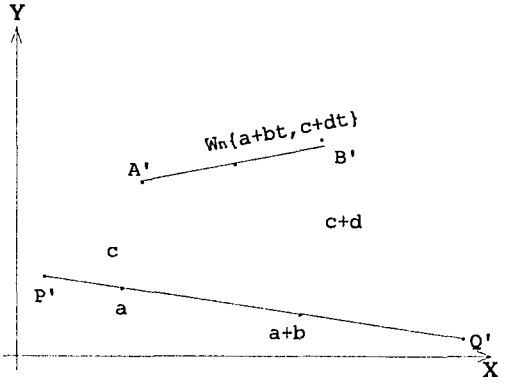


Figure 4.4(b) Line segment A'B' after transform

Since $x_{p'}$, $y_{p'}$, $x_{q'}$, $y_{q'}$, l' and a, b, c, d are all constant, the derivative of $(x_n)'_t$ and $(y_n)'_t$ are

$$\frac{dx_n}{dt} = b(x_{q'}-x_{p'}) - \frac{d}{l'}(y_{q'}-y_{p'})$$

$$\frac{dy_n}{dt} = b(y_{q'}-y_{p'}) + \frac{d}{l'}(x_{q'}-x_{p'}) \qquad (4.5)$$

The derivative $\frac{dy_n}{dx_n}\Big|_{t=t_0}$ is the slope of tangent line when $t = t_0$.

$$\frac{dy_n}{dx_n} = \frac{\frac{dy_n}{dt}}{\frac{dx_n}{dt}} = \frac{b(y_{q'} - y_{p'}) + \frac{d}{l'}(x_{q'} - x_{p'})}{b(x_{q'} - x_{p'}) - \frac{d}{l'}(y_{q'} - y_{p'})} \qquad (4.6)$$

The derivative $\frac{dy_n}{dx_n}(t)$ is a constant. Therefore,

A'B' is a straight line if $t \in \mathbb{R}$; and it is a line segment if $0 \leq t \leq 1$.

From the above description the one control line morphing transform maps a straight line into a straight line, and a straight line segment into a straight line segment. The one control line morphing transform is a linear transform. It causes images in the two-dimensional plane change only in translation, rotation and size(enlarging and shrinking), but it maintains the shape or the pattern of the image.

Theorem 4.4 The morphing transform for the one control line preserves the parallel property of straight line segments.

Proof:

Let PQ be the control line with length l, $P(x_p, y_p)$ and $Q(x_q, y_q)$. A_1B_1 and A_2B_2 are parallel line segments, and $A_1B_1 \parallel A_2B_2$. W_1 is any point on A_1B_1; W_2 is any point on A_2B_2. (See Figure 4.5(a)).

$$W_1(x_1, y_1) = W_{1\to PQ}\{a_1+b_1t, c_1+d_1t\}$$

$$W_2(x_2,\ y_2\} = W_{2\to PQ}\ \{a_2+b_2t,\ c_2+d_2t\} \qquad 0 \le t \le 1$$

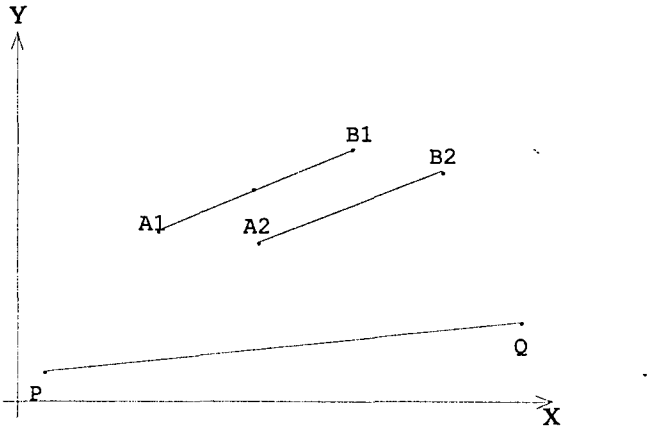


Figure 4.5 (a) Parallel line segment

Assume the slope of PQ is k, then $k = \frac{y_q - y_p}{x_q - x_p}$. Without loss of generality $x_q \neq x_p$, (otherwise, use $1/k$). Apply Theorem 4.3,

$$\frac{dy_1}{dx_1} = \frac{b_1(y_q - y_p) + \frac{d_1}{l}(x_q - x_p)}{b_1(x_q - x_p) - \frac{d_1}{l}(y_q - y_p)} = \frac{b_1k + \frac{d_1}{l}}{b_1 - \frac{d_1}{lk}} \qquad (4.7)$$

using same reasoning

$$\frac{dy_2}{dx_2} = \frac{b_1(y_q - y_p) + \frac{d_1}{l}(x_q - x_p)}{b_1(x_q - x_p) - \frac{d_1}{l}(y_q - y_p)} = \frac{b_2k + \frac{d_2}{l}}{b_2 - \frac{d_2}{lk}} \qquad (4.8)$$

Since $A_1B_1 \parallel A_2B_2$, then

$$\frac{dy_2}{dx_2} = \frac{dy_1}{dx_1}$$

This is

$$\frac{b_1 k + \frac{d_1}{l}}{b_1 - \frac{d_1}{lk}} = \frac{b_2 k + \frac{d_2}{l}}{b_2 - \frac{d_2}{lk}}$$

Therefore,

$$\frac{lk + \frac{d_1}{b_1}}{lk - \frac{d_1}{b_1}} = \frac{lk + \frac{d_2}{b_2}}{lk - \frac{d_2}{b_2}}$$

Add 1 to each side,

$$\frac{2lk}{lk - \frac{d_1}{b_1}} = \frac{2lk}{lk - \frac{d_2}{b_2}}$$

That is $lk - \frac{d_1}{b_1} = lk - \frac{d_2}{b_2}$, so that $\frac{b_1}{d_1} = \frac{b_2}{d_2}$. This is necessary condition for $A_1B_1 \parallel A_2B_2$. After morphing transform, P'Q' is the new control line with P'(x'_p, y'_p) and Q'(x'_q, y'_q), slope $k' = \frac{y'_q - y'_p}{x'_q - x'_p}$, and length l'.

W_{n1} is the image of W_1 on $A_1'B_1'$ and W_{n2} is the image of W_2 on $A_2'B_2'$. According to definition of morphing

transform,

$$W_{n1}(x_{n1},\ y_{n1}) = W_{n1\to P'Q'}\{a_1+b_1t,\ c_1+d_1t\}$$

$$W_{n2}(x_{n2},\ y_{n2}) = W_{n2\to P'Q'}\{a_2+b_2t,\ c_2+d_2t\}$$

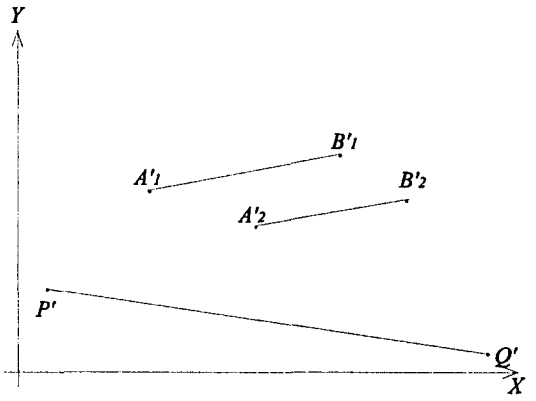


Figure 4.5(b) Parallel line segment after morphing transform.

From theorem 4.3, $A_1'B_1'$ and $A_2'B_2'$ are straight line segments, the slope of $A_1'B_1'$ and $A_2'B_2'$ are

$$\frac{dy_{n1}}{dx_{n1}} = \frac{b_1k' + \frac{d_1}{1}}{b_1 - \frac{d_1}{1k'}} = \frac{\frac{b_1}{d_1}k' + \frac{1}{1'}}{\frac{b_1}{d_1} - \frac{1}{1k'}} \qquad (4.9)$$

$$\frac{dy_{n2}}{dx_{n2}} = \frac{b_2k' + \frac{d_2}{1}}{b_2 - \frac{d_2}{1k'}} = \frac{\frac{b_2}{d_2}k' + \frac{1}{1'}}{\frac{b_2}{d_2} - \frac{1}{1k'}} \qquad (4.10)$$

Since $\frac{b_1}{d_1} = \frac{b_2}{d_2}$, then $\frac{dy_{n1}}{dx_{n1}} = \frac{dy_{n2}}{dx_{n2}}$, that means the line

segments $A'_1B'_1 \parallel A'_2B'_2$.

4.3. Two Control Lines Case

For images with two control lines, the morphing transform changes the positions of these control lines. One point on the plane will have two image points. The image point preserves its relation with each control line. These two image points will be averaged into one with certain rules. There are many ways of averaging, corresponding to the many types of morphing transforms. In this study, $1/c^2$ (c is the non-zero distance) is used for averaging.

Definition 4.5 For an image with m ($m \geq 1$) control lines, $l_1, l_2, \ldots, l_m$. The morphing transform occurs by translating, rotating, and scaling the control lines into other control lines, $l_1', l_2', \ldots, l_m'$. Any point W on the plane, $W = W_{-1i}\{a_i, c_i\}$, $i = 1, 2, \ldots, m$, will have m image points after morphing transform W_i, to the control line l_i, $i = 1, 2, \ldots, m$. Each image point preserves the relation with its transformed control line.

$$W_i = W_{i\to 1i}\{a_i,\ c_i\} \qquad i = 1,\ 2,\ \ldots,\ m.$$

The new image point W_n of W after morphing transform is then

$$W_n = \frac{\sum_{i=1}^{m} \frac{1}{c_i^2} W_i}{\sum_{i=1}^{m} \frac{1}{c_i^2}} \tag{4.11}$$

Figure 4.6(a) - (c) show the steps of morphing transform W with two control lines. In Figure 4.6(a), P_1Q_1 and P_2Q_2 are control lines. W is any point on XY plane. W has two sets of relations, one is to P_1Q_1, one is to P_2Q_2. W's relation to P_1Q_1 is denoted as $W_{\to P1Q1}\{a,\ c\}$. a, c have the same meanings as in one control line case. And W relation to P_2Q_2 is denoted as $W_{\to P2Q2}\{u,\ m\}$.

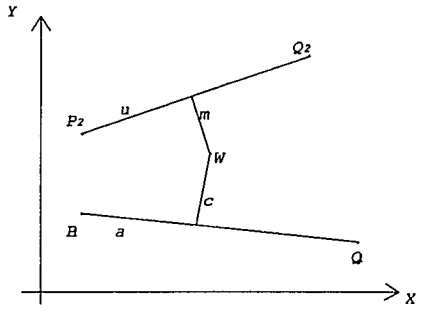


Figure 4.6(a) Two control lines and one point on the plane.

After the morphing transform is used, the control lines in Figure 4.6(b), control line P_1Q_1 and P_2Q_2 are transformed into the new positions $P'_1Q'_1$ and $P'_2Q'_2$.

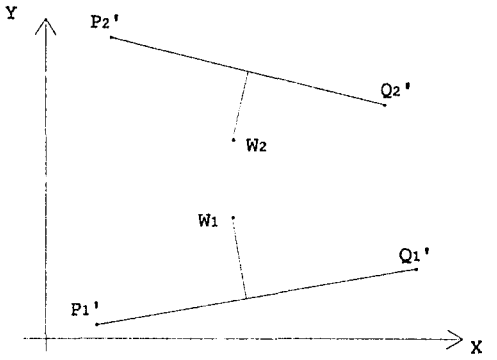


Figure 4.6(b) Morphing transform changes the control lines.

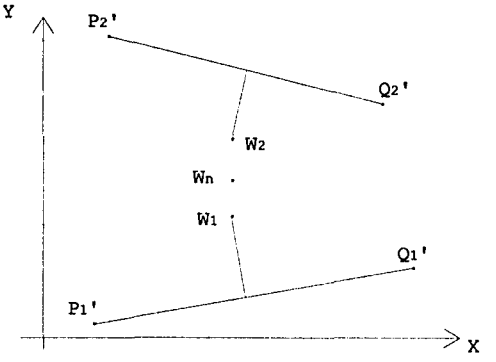


Figure 4.6(c) Morphing transform averages the two image points.

W_1 and W_2 are the images of W. W_1 has the same relation to $P_1'Q_1'$ as W relates to P_1Q_1, and W_2 has the same relation to $P_2'Q_2'$ as W relates to P_2Q_2. That is $W_{1\text{-}P1'Q1'}\{a, c\}$ and $W_{2\text{-}P2'Q2'}\{u, m\}$.

Then in Figure 4.6(c), the two image points, W_1 and W_2, are averaged into a single new point W_n. W_n is W after the morphing transform is used. For averaging image points, different mathematical formula may applied for different emphasis. Here harmonic averaging of power

2 is used for averaging.

$$W_n = \frac{\frac{1}{r_1^2}}{\frac{1}{r_1^2}+\frac{1}{r_1^2}} W_1 + \frac{\frac{1}{r_2^2}}{\frac{1}{r_2^2}+\frac{1}{r_2^2}} W_2 = \frac{r_2^2 W_1 + r_1^2 W_2}{r_1^2 + r_2^2} \qquad (4.12)$$

r_1 and r_2 are the distances of W to each control line P_1Q_1 and P_2Q_2, respectively.

This expression shows that the further a point is located from the control line, the less the point is affected (or controlled) by that control line. In Figure 4.6, $r_1 = c$ and $r_2 = m$.

If W is dependent on a variable t, such that $W_{\sim P1Q1}=\{a+bt,\ c+dt\}$ and $W_{\sim P2Q2}=\{m+nt,\ a+vt\}$, $0 \le t \le 1$, W provides a line segment. When W is a function of t, W_1 and W_2 are also functions of t, so also are r_1 and r_2. From Equation (4.12), the new image point of W after the morphing transform is used, W_n is also function of t. When t changes from 0 to 1, W_n forms a line segment.

Figure 4.7 shows a image of a line segment with two

control lines the two end points are A and B. P_1Q_1 and P_2Q_2 are control lines. A, B have the relation with P_1Q_1 and P_2Q_2 that $A_{\neg P1Q1}\{a, c\}$, $B_{\neg P1Q1}\{a+b, c+d\}$, and $A_{\neg P2Q2}\{u, m\}$, $B_{\neg P2Q2}\{u+v, m+n\}$. If each point W between A and B satisfies that $W_{\neg P1Q1}\{a+bt, c+dt\}$ and $W_{\neg P2Q2}\{u+vt, m+nt\}$, this line segment is a straight line segment, as in Figure 4.7.

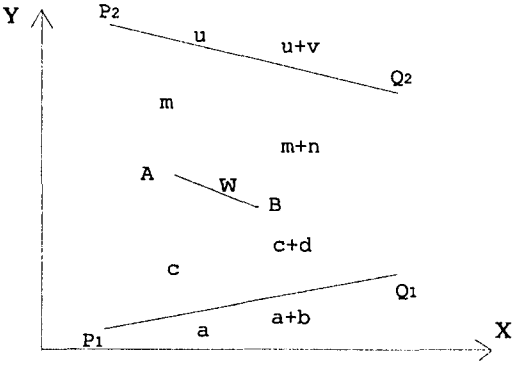


Figure 4.7 One line segment with two control lines.

A straight line segment maps into a straight linesegment after a one control line morphing transform is used because this transform does not need to average image points. In the two control line case, the morphing transform may not map straight lines into straight lines. So this transform may not be linear transform.

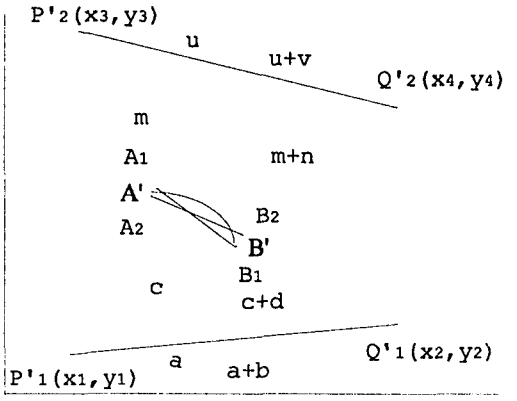


Figure 4.8 Two images of line segments after morphing transform.

Theorem 4.6: In Figure 4.7, let P_1Q_1 and P_2Q_2 be two control lines, and let AB be a straight line segment. After morphing transform, in Figure 4.8, control lines P_1Q_1 and P_2Q_2 are moved to the new positions $P_1'Q_1'$ and $P_2'Q_2'$. A_1B_1 and A_2B_2 are images of AB related to control line $P_1'Q_1'$ and $P_2'Q_2'$. A'B' is the morphing transformed line segment of AB. Then A'B' is usually not a straight line segment. It is a straight line segment if

(1) Extensions of the two control lines and the line segment can meet at one point, or

(2) Control lines $P_1'Q_1'$ and $P_2'Q_2'$ have the same geometric relationship as P_1Q_1 and P_2Q_2, that means AB has only one image line segment after morphing

transform.

Proof:

As in Figure 4.7, assume $A_{\sim P1Q1}\{a, c\}$, $A_{\sim P2Q2}\{u, m\}$, and $B_{\sim P1Q1}\{a+b, c+d\}$, $B_{\sim P2Q2}\{u+v, m+n\}$. Therefore, any point W between AB can be written as

$W_{\sim P1Q1}\{a+bt, c+dt\}$, $W_{\sim P2Q2}\{u+vt, m+nt\}$ $0 \le t \le 1$

$A=W|_{t=0}$ $B=W|_{t=1}$

From theorem 4.3, $A_1'B_1'$ and $A_2'B_2'$ are straight line segments. Assume the end points of the control lines P_1', Q_1', P_2', Q_2' have coordinate position as $P_1'(x_1, y_1)$, $Q_1'(x_2, y_2)$, $P_2'(x_3, y_3)$, and $Q_2'(x_4, y_4)$. The length of $P_1'Q_1'$ is $l_1 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$, the length of $P_2'Q_2'$ is $l_2 = \sqrt{(x_4 - x_3)^2 + (y_4 - y_3)^2}$. Any point $W(t)$ on AB is formed by two points, $W_1(t)$ and $W_2(t)$. $W_1(t)$ keeps the relation with $P_1'Q_1'$, such that $W_{1\sim P1'Q1'}\{a+bt, c+dt\}$, and $W_2(t)$ keeps the relation with $P_2'Q_2'$, such that $W_{2\sim P2'Q2'}\{u+vt, m+nt\}$. So we have,

$$x_{A_1} = x_1 + a(x_2-x_1) - \frac{b}{l_1}(y_2-y_1)$$

$$y_{A_1} = y_1 + a(y_2-y_1) + \frac{b}{l_1}(x_2-x_1)$$

$$x_{A_2} = x_3 + u(x_4-x_3) - \frac{n}{l_2}(y_4-y_3)$$

$$y_{A_2} = y_3 + u(y_4-y_3) + \frac{n}{l_2}(x_4-x_3) \qquad (4.13)$$

The increments are defined

$$\delta_{x_1} = b(x_2-x_1) - \frac{d}{l_1}(y_2-y_1)$$

$$\delta_{y_1} = b(y_2-y_1) + \frac{d}{l_1}(x_2-x_1)$$

$$\delta_{x_2} = v(x_4-x_3) - \frac{n}{l_2}(y_4-y_3)$$

$$\delta_{y_2} = v(y_4-y_3) + \frac{n}{l_2}(x_4-x_3) \qquad (4.14)$$

$W_1(t)$ position at XY - coordinate is $W_1(X_1, Y_1)$,

$$X_1 = x_1 + (a + bt)(x_2 - x_1) - \frac{c + dt}{l_1}(y_2 - y_1) = x_{A_1} + \delta_{x_1} t \qquad t \in [0,1]$$

$$Y_1 = y_1 + (a + bt)(y_2 - y_1) + \frac{c + dt}{l_1}(x_2 - x_1) = y_{A_1} + \delta_{y_1} t \qquad (4.15)$$

The distance from $W_1(t)$ to $P'_1Q'_1$ is

$$r_1 = c + dt \tag{4.16}$$

$W_2(t)$ position at XY - coordinate is $W_2(X_2, Y_2)$,

$$X_2 = x_3 + (u + vt)(x_4 - x_3) - \frac{m + nt}{l_2}(y_4 - y_3) = x_{A_2} + \delta_{x_2} t \qquad t \in [0,1]$$

$$Y_2 = y_3 + (u + vt)(y_4 - y_3) + \frac{m + nt}{l_2}(x_4 - x_3) = y_{A_2} + \delta_{y_2} t \tag{4.17}$$

The distance from W2(t) to $P_2'Q_2'$ is

$$r_2 = c + dt \tag{4.18}$$

From equation (4.12), the new transformed point of W is W_n, and $W_n(X_n, Y_n)$.

$$W_n = \frac{\frac{1}{r_1^2}}{\frac{1}{r_1^2}+\frac{1}{r_2^2}} W_1 + \frac{\frac{1}{r_2^2}}{\frac{1}{r_1^2}+\frac{1}{r_2^2}} W_2$$

$$= \frac{r_2^2}{r_1^2+r_2^2} W_1 + \frac{r_1^2}{r_1^2+r_2^2} W_2 \tag{4.19}$$

W_n is function of t, when t changes from 0 to 1, W' draw

a line A'B'. Equation (4.19) can be simplified as following, assume

$$s_1 = r_1^2 = (c + dt)^2$$

$$s_2 = r_2^2 = (m + nt)^2$$

so

$$W_n = \frac{s_2 W_1 + s_1 W_2}{s_1 + s_2} \tag{4.20}$$

So that

$$\frac{dW_n}{dt} = \left(\frac{s_2 W_1 + s_1 W_2}{s_1 + s_2}\right)'_t$$

$$= \frac{(s_2 W_1 + s_1 W_2)'(s_1 + s_2) - (s_2 W_1 + s_1 W_2)(s_1 + s_2)'}{(s_1 + s_2)^2}$$

$$= \frac{(s_2' W_1 + s_2 W_1' + s_1' W_2 + s_1 W_2')(s_1 + s_2) - (s_2 W_1 + s_1 W_2)(s_1' + s_2')}{(s_1 + s_2)^2}$$

$$= \frac{(s_2 W_1' + s_1 W_2')(s_1 + s_2) + s_1 s_2' W_1 + s_1 s_1' W_2 + s_2 s_2' W_1 + s_2 s_1' W_2 - (s_1' s_2 W_1 + s_1 s_1' W_2 + s_2 s_2' W_1 + s_1 s_2' W_2)}{(s_1 + s_2)^2}$$

$$= \frac{(s_2 W_1' + s_1 W_2')(s_1 + s_2) + s_1 s_2' W_1 - s_1 s_2' W_2 + s_2 s_1' W_2 - s_2 s_1' W_1}{(s_1 + s_2)^2}$$

$$= \frac{(s_1' s_2 - s_1 s_2')(W_2 - W_1) + (s_2 W_1' + s_1 W_2')(s_1 + s_2)}{(s_1 + s_2)^2} \tag{4.21}$$

Finally,

$$\frac{dW_n}{dt} = \frac{s_2^2\left(\frac{s_1}{s_2}\right)'(W_2-W_1)+(s_1+s_2)\,(s_2W_1'+s_1W_2')}{(s_1+s_2)^2} \qquad (4.22)$$

W_n may represent either X_n or Y_n. Therefore the slope of W_n at t, $0 \le t \le 1$, is

$$\frac{dY_n}{dX_n} = \frac{\dfrac{s_2^2\left(\frac{s_1}{s_2}\right)'(Y_2-Y_1)+(s_1+s_2)\,(s_2Y_1'+s_1Y_2')}{(s_1+s_2)^2}}{\dfrac{s_2^2\left(\frac{s_1}{s_2}\right)'(X_2-X_1)+(s_1+s_2)\,(s_2X_1'+s_1X_2')}{(s_1+s_2)^2}}$$

$$= \frac{s_2^2\left(\frac{s_1}{s_2}\right)'(Y_2-Y_1)+(s_1+s_2)\,(s_2Y_1'+s_1Y_2')}{s_2^2\left(\frac{s_1}{s_2}\right)'(X_2-X_1)+(s_1+s_2)\,(s_2X_1'+s_1X_2')} \qquad (4.23)$$

Since $\frac{dY_n}{dX_n}$ is a function of variable t, A'B' has changing slope at points between A' and B'.

Refer to Equation (4.15) and (4.17), X_1, Y_1, X_2 and Y_2 are linear function of t, so that $(X_1)'_t$, $(Y_1)'_t$, $(X_2)'_t$

and $(Y_2)'_t$ are all constants. If $\frac{s_1}{s_2} = h^2 = constant$, then $(\frac{s_1}{s_2})' = 0$

$$\frac{dY_n}{dX_n} = \frac{(s_1+s_2)(s_2Y_1'+s_1Y_2')}{(s_1+s_2)(s_2X_1'+s_1X_2')} = \frac{s_2(Y_1'+h^2Y_2')}{s_2(X_1'+h^2X_2')} \qquad (4.24)$$

$$\frac{dY_n}{dX_n} = \frac{Y_1'+h^2Y_2'}{X_1'+h^2X_2'} = constant \qquad (4.25)$$

$\frac{dY_n}{dX_n}$ is a constant, meaning that $W_n(t)$ forms a straight line segment A'B', $0 \le t \le 1$.

Figure 4.9 shows the situation that any point on AB has proportional distances to each control lines, and extension of the line segments P_1Q_1, P_2Q_2, and AB.

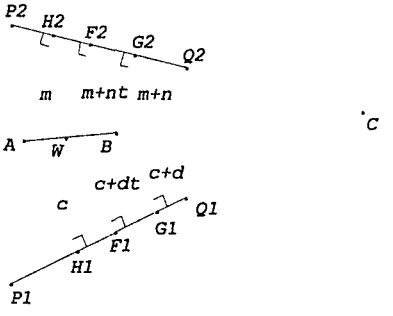


Figure 4.9 Any point on AB has proportion distances to each control lines.

If three lines P_1Q_1, P_2Q_2 and AB meet at same point C, then we can derive the following relation from two triangle $\triangle AWF_1$ and $\triangle AWF_2$.

$$\frac{c}{c+dt} = \frac{m}{m+nt} = \frac{CA}{CW} \qquad (4.26)$$

so that

$$\frac{c+dt}{m+nt} = \frac{c}{m} = constant \qquad (4.27)$$

Therefore

$$\frac{s_1}{s_2} = \frac{r_1^2}{r_2^2} = \frac{(c+dt)^2}{(m+nt)^2} = \frac{c^2}{m^2} = constant \qquad (4.28)$$

If the line AB passes through the conjunction of extension line P_1Q_1 and P_2Q_2, then that straight line segment maps to a straight line segment.

In Figure 4.8, if $P_1'Q_1'$ and $P_2'Q_2'$ have the same geometrical relation as P_1Q_1 and P_2Q_2, in Figure 4.7, then any W on AB will have the same image point to each control line. Then

$$W_1(t) = W_2(t)$$

So that

$$Y_1 = Y_2 = Y \quad \text{and} \quad X_1 = X_2 = X$$

From (16), the slope of A'B' at $W_n(t)$ is

$$\frac{dY_n}{dX_n} = \frac{(s_1+s_2)(s_2Y_1'+s_1Y_2')}{(s_1+s_2)(s_2X_1'+s_1X_2')} = \frac{Y'}{X'} = constant \qquad (4.29)$$

Then A'B' is a straight line segment.

This theorem can also be understood in terms of how the position of new transformed point, W_n, is affected by the distances between the two control lines. If the distance to each control line changes proportionally when t changes, then the effect will be cancelled proportionally. The result makes W_n a straight line segment. But usually, the transform of a straight line segment is not straight line segment.

If A'B' is not a straight line segment, a metrics is established to bound the deformation of this line segment AB.

Theorem 4.7 In Figure 4.7, let P_1Q_1 and P_2Q_2 be two control lines, and let AB be a straight line segment. After applying the morphing transform, in Figure 4.8, control lines P_1Q_1 and P_2Q_2 are moved to the new positions $P_1'Q_1'$ and $P_2'Q_2'$. A_1B_1 and A_2B_2 are images of AB related to control line $P_1'Q_1'$ and $P_2'Q_2'$. From theorem 4.3, A_1B_1 and A_2B_2 are straight line segments. A'B' is the morphing transformed line segment of AB. Then A'B' is inside of $A_1B_1A_2B_2$ or $A_1A_2B_1B$ whichever forms a quadrilateral.

Proof:

Let us draw Figure 4.7 and Figure 4.8 again.

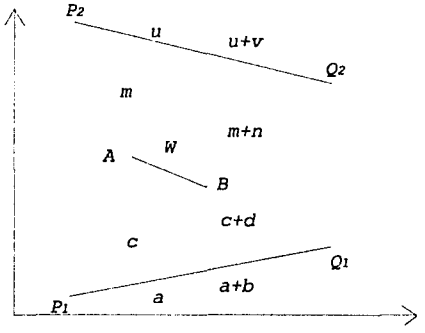


Figure 4.7 One line segment with two control lines.

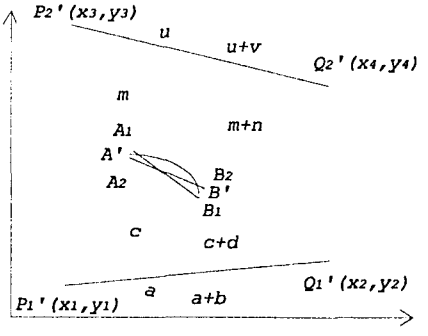


Figure 4.8 Two images of line segments after morphing transform.

From Theorem 4.3, A_1B_1 and A_2B_2 are straight lines. Assume W is any point on AB, $W_1(X_1, Y_1)$ is corresponding

point on A_1B_1, and $W_2(X_2, Y_2)$ is the corresponding point on A_2B_2. From equation (4.12), the morphing transformed W is $W_n(X_n, Y_n)$.

$$W_n = \frac{r_2^2}{r_1^2+r_2^2} W_1 + \frac{r_1^2}{r_1^2+r_2^2} W_2 \qquad (4.30)$$

r_1 and r_2 are distances from W to P_1Q_1 and P_2Q_2, respectively. W_1 and W_2 can either represent X_1 and X_2 or Y_1 and Y_2. Without loss of generality, let $W_1 = X_1$, $W_2 = X_2$ and $X_1 \le X_2$. Then the above equation (4.30) can be written as

$$X_n = \frac{r_2^2}{r_1^2+r_2^2} X_1 + \frac{r_1^2}{r_1^2+r_2^2} X_2$$

$$= X_1 + \frac{r_1^2}{r_1^2+r_2^2} (X_2-X_1)$$

$$= X_2 - \frac{r_2^2}{r_1^2+r_2^2} (X_2-X_1) \qquad (4.31)$$

Since $X_2 - X_1 \ge 0$ then $X_1 \le X_n \le X_2$.

Therefore W_n is in between W_1 and W_2. Because W is any point on AB, then A'B', the morphing transformed line segment A'B' lies between A_1B_1 and A_2B_2. So, A'B' lines in

$A_1B_1A_2B_2$ or $A_1A_2B_1B_2$ whichever forms a quadrilateral.

The morphing transform after causes a straight line segment to map into a deformed line segment. The deformed line segment can be bounded in a certain area indicated by Theorem 4.7. The next theorem will show that morphing transform deforms the straight line into a convex curve.

Theorem 4.8 Let AB be a straight line segment in the two control lines system. Let A'B' be the morphing transformed AB. Then A'B' is a convex line segment, that means if A'($x_{A'}$, $y_{A'}$) and B'($x_{B'}$, $y_{B'}$) are two end points of A'B', and $G_y = y_{B'} - y_{A'}$, $G_x = x_{B'} - x_{A'}$, and if $W_n(X_n, Y_n)$ is any point on A'B', then there exist only one t_0 that satisfy

$$\left.\frac{dY_n}{dX_n}\right|_{t=t_0} = \frac{G_y}{G_x} \qquad (4.32)$$

and

$$t_0 = \frac{G_x(y_{A_2} - y_{A_1}) - G_y(x_{A_2} - xA_1)}{G_y(\delta_{x_2} - \delta_{x_1}) - G_x(\delta_{y_2} - \delta_{y_1})} \qquad (4.33)$$

δ_{x_2}, δ_{x_2}, δ_{y_1}, δ_{y_2} are represented in equation (4.14).

Proof:

From equation (4.23),

$$\frac{dY_n}{dX_n} = \frac{s_2^2\left(\frac{s_1}{s_2}\right)'(Y_2-Y_1)+(s_1+s_2)(s_2Y_1'+s_1Y_2')}{s_2^2\left(\frac{s_1}{s_2}\right)'(X_2-X_1)+(s_1+s_2)(s_2X_1'+s_1X_2')} \qquad (4.23)$$

Assume when $t = t_0$,

This is the slope of secant A'B', t_0 is where the

$$\left.\frac{dY_n}{dX_n}\right|_{t=t_0} = \frac{G_y}{G_x} \qquad (4.32)$$

deformed line A'B' has parallel tangent line to the secant. When t_0 is fixed(t_0 is not assumed to be either unique or finite).

$$\left.s_2Y_1'+s_1Y_2'\right|_{t=t_0} = \lim_{\Delta t\to 0}\left[s_2\left(\frac{Y_1(t_0+\Delta t)-Y_1(t_0)}{\Delta t}\right)+s_1\left(\frac{Y_2(t_0+\Delta t)-Y_2(t_0)}{\Delta t}\right)\right]$$

$$= (s_1+s_2)\lim_{\Delta t\to 0}\frac{1}{\Delta t}\left[\frac{s_2Y_1(t_0+\Delta t)+s_1Y_2(t_0+\Delta t)}{s_1+s_2} - \frac{s_2Y_1(t_0)+s_1Y_2(t_0)}{s_1+s_2}\right]$$

$$= \lim_{\Delta t\to 0}\frac{1}{\Delta t}\left[s_2Y_1(t_0+\Delta t)+s_1Y_2(t_0+\Delta t)-s_2Y_1(t_0)-s_1Y_2(t_0)\right]$$

$$= (s_1+s_2)\lim_{\Delta t\to 0}\frac{Y_n(t_0+\Delta t)-Y_n(t_0)}{\Delta t}$$

$$= (s_1+s_2)Y_n'(t_0) \qquad (4.34a)$$

The same reason

$$s_2 X_1' + s_1 X_2' = (s_1 + s_2) X_n'(t_0) \qquad (4.34b)$$

Therefore

$$\left.\frac{dY_n}{dX_n}\right|_{t=t_0} = \left.\frac{s_2^2 (\frac{s_1}{s_2})' (Y_2 - Y_1) + (s_1 + s_2)^2 Y_n'}{s_2^2 (\frac{s_1}{s_2})' (X_2 - X_1) + (s_1 + s_2)^2 X_n'}\right|_{t=t_0} = \frac{G_y}{G_x} \qquad (4.35)$$

Since

$$\left.\frac{Y_n'}{X_n'}\right|_{t=t_0} = \left.\frac{dY_n}{dX_n}\right|_{t=t_0} = \frac{G_y}{G_x} = constant$$

so

$$\left.\frac{dY_n}{dX_n}\right|_{t=t_0} = \frac{s_2^2 (\frac{s_1}{s_2})' (Y_2 - Y_1)}{s_2^2 (\frac{s_1}{s_2})' (X_2 - X_1)} = \frac{G_y}{G_x} \qquad (4.36)$$

From Theorem 4.6, if $\frac{s_1}{s_2} = constant$, or if $\frac{s_1}{s_2} \neq constant$, but the control lines have the same geometrical configuration before and after the morphing transform, each point has only one image after the transform, this means $X_1 = X_2$, and $Y_1 = Y_2$, then, morphing transform maps the straight line to a straight line, from equation (4.29), every point on A'B' has the same slope

$\frac{G_y}{G_x}$, otherwise

$$\left.\frac{dY_n}{dX_n}\right|_{t=t_0} = \left.\frac{Y_2-Y_1}{X_2-X_1}\right|_{t=t_0} = \frac{G_y}{G_x} \qquad (4.37)$$

Since G_y, G_x are constant. From (4.15) and (4.17)

$$\left.\frac{Y_2-Y_1}{X_2-X_1}\right|_{t=t_0} = \frac{y_{A_2}+\delta_{y_2}t_0-(y_{A_1}+\delta_{y_1}t_0)}{x_{A_2}+\delta_{x_2}t_0-(x_{A_1}+\delta_{x_1}t_0)} = \frac{G_y}{G_x} \qquad (4.38)$$

Then we can have

$$[G_y(\delta_{x_2}-\delta_{x_1})-G_x(\delta_{y_2}-\delta_{y_1})]t_0 = G_x(y_{A_2}-y_{A_1})-G_y(x_{A_2}-x_{A_1}) \qquad (4.39)$$

If $G_y(\delta_{x_2}-\delta_{x_1}) \neq G_x(\delta_{y_2}-\delta_{y_1})$, then

$$t_0 = \frac{G_x(y_{A_2}-y_{A_1})-G_y(x_{A_2}-x_{A_1})}{G_y(\delta_{x_2}-\delta_{x_1})-G_x(\delta_{y_2}-\delta_{y_1})} \qquad (4.36)$$

When $t = t_0$, $W(x_w, y_w)$ position is

$$x_w = \frac{(m+nt_0)^2(x_{A_1}+\delta_{x_1}t_0)+(c+dt_0)^2(x_{A_2}+\delta_{x_2}t_0)}{(m+nt_0)^2+(c+dt_0)^2}$$

$$y_w = \frac{(m+nt_0)^2(y_{A_1}+\delta_{y_1}t_0)+(c+dt_0)^2(y_{A_2}+\delta_{y_2}t_0)}{(m+nt_0)^2+(c+dt_0)^2} \qquad (4.40)$$

In the two control line morphing transform, a straight line segment generally maps into a convex line segment. If the control lines meet the condition in Theorem 4.6, the transformed convex line segment becomes a special convex line segment --- straight line segment.

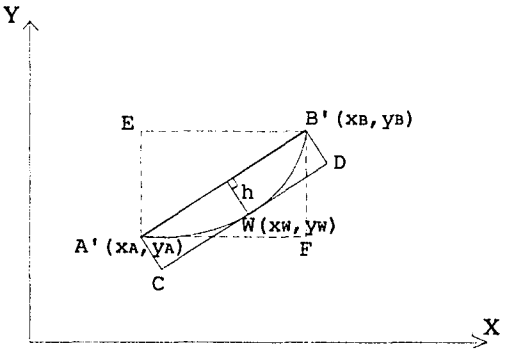


Figure 4.10 Minimum rectangle contains the transformed convex line segment.

See from Figure 4.10, let A'B'D C represents the minimum rectangle contains the transformed convex line segment, let A'E B'F represents the rectangle which the four sides are parallel to the coordinate axis. When comparing the area of A'B'DC and the area of A'EB'F, a simple form is derived.

Corollary 4.9: Let AB be a straight line segment. Arc A'B' is the convex line segment after two control

line morphing transform, where A'(x_A, y_A), B'(x_B, y_B). Let A'B'DC be the minimum rectangle containing arc A'B'. W(x_W, y_W) is the tangent point on arc A'B', as in Figure 4.10. The minimum rectangle which contains the arc A'B' is A'B'DC, then the area of A'B'DC is

$$Area_{A'B'DC} = \left| (x_B - x_A)(y_W - y_A) - (y_B - y_A)(x_W - x_A) \right| \qquad (4.41)$$

The rectangle A'EB'F has all of its side parallel to the coordinate axis. The area of A'EB'F is

$$Area_{A'EB'F} = \left| (y_B - y_A)(x_B - x_A) \right| \qquad (4.42)$$

then

$$\frac{Area_{A'B'DC}}{Area_{A'EB'F}} = |Y_W - X_W| \qquad (4.43)$$

where $W_n(X_W, Y_W)$ is the new coordinate after translating A' to origin and rescaling B' to (1,1). As in Figure 4.11.

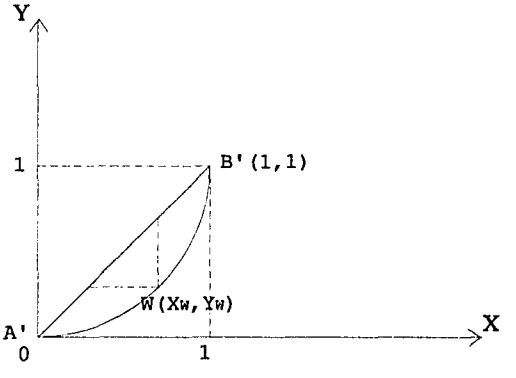


Figure 4.11 A' translate to origin B' translate to (1,1).

Proof:

Let $G_y = y_B - y_A$, $G_x = x_B - x_A$,

$$Area_{A'EB'F} = |G_x G_y| \tag{4.44}$$

Then any point (x,y) on secant A'B',

$$x = x_A + G_x t, \qquad y = y_A + G_y t, \qquad t \in [0, 1] \tag{4.45}$$

h is the length from W to the point (x, y) on A'B', so

$$h^2 = (x_A + G_x t - x_W)^2 + (y_A + G_y t - y_W)^2 \tag{4.46}$$

h_{min} is the height A'C, so

$$\frac{dh^2}{dt} = 2(x_A + G_x t - x_W) G_x + 2(y_A + G_y t - y_W) G_y \tag{4.47}$$

Let $\frac{dh^2}{dt} = 0$, we have

$$t = \frac{G_x(x_W - x_A) + G_y(y_W - y_A)}{G_x^2 + G_y^2} \tag{4.48}$$

Therefore, the height , h_{min}

$$h_{min}^2 = [G_x \frac{G_x(x_W - x_A) + G_y(y_W - y_A)}{G_x^2 + G_y^2} - (x_W - x_A)]^2$$

$$+ [G_y \frac{G_x(x_W - x_A) + G_y(y_W - y_A)}{G_x^2 + G_y^2} - (y_W - y_A)]^2$$

So,

$$h_{min}^2 = \frac{[G_x(y_W - y_A) - G_y(x_W - x_A)]^2}{G_x^2 + G_y^2} \qquad (4.49)$$

Let $l_{A'B'}$ be the length of A'B'. Then

$$l_{A'B'}^2 = G_x^2 + G_y^2 \qquad (4.50)$$

Then the area of A'B'DC is

$$(Area_{A'B'DC})^2 = l_{A'B'}^2 h_{min}^2 = [G_x(y_w - y_A) - G_y(x_W - x_A)]^2 \qquad (4.51)$$

So

$$Area_{A'B'DC} = |G_x(y_W - y_A) - G_y(x_W - x_A)| = |(x_B - x_A)(y_W - y_B) - (y_B - y_A)(x_W - x_A)| \qquad (4.52)$$

If $G_xG_y \neq 0$, that is, if the *area* A'EB'F is not zero. The comparison between the area of the rectangles A'B'CD and A'EB'F is

$$\frac{Area_{A'B'DC}}{Area_{A'EB'F}} = \frac{Area_{A'B'DC}}{|G_xG_y|} = \left|\frac{y_W - y_A}{G_y} - \frac{x_w - x_A}{G_x}\right|$$

$$\frac{Area_{A'B'DC}}{Area_{A'EB'F}} = \left|\frac{y_W - y_A}{y_B - y_A} - \frac{x_W - x_A}{x_W - x_A}\right| \qquad (4.53)$$

If translate A' to (0,0), B' to (1,1), then the unified

area $Area_{A'EB'F} = 1$ (4.54)

$$Area_{A'B'DC} = |Y_W - X_W| \quad (4.55)$$

where $W_n(x_W, y_W)$ is the image of W after translation.

4.4. Multiple Control Lines Case

In the multiple control line system, say m control lines are used, the deformation of morphing transformed straight line segment is more complicated. First, the straight line segment will have m image line segments after the morphing transform. Each image line segment preserves its relation to its corresponding control line. These image line segments may change orientations, sizes and positions, but they are still straight line segments. If some control lines do not change their geometric relation, their images will remain as one image. But they are still counted as different line segments.

Second, when we use equation (4.11) to combine or average the image line segments to get the morphing transformed line segments, that line segment very

possibly is not a straight line segment, it may not even be a convex line segment like in two control line case. The deformed line segment is bounded using the following theorem.

Theorem 4.10: In m control line morphing transform. Let $l_1, l_2, \ldots, l_m$ be the original control lines, and let $l'_1, l'_2, \ldots, l'_m$ be the control lines after the morphing transform. If AB is a straight line segment, and $A_1B_1, A_2B_2, \ldots, A_mB_m$ are the images of AB according to morphing transform the control lines, $l_1, l_2, \ldots, l_m$. And A'B' is the line segment of the morphing transformed AB. Then A'B' lies inside the maximum polygon formed by $A_1, A_2, \ldots, A_m, B_1, B_2, \ldots, B_m$.

Proof:

Mathematical induction is used to prove this theorem. First, let $m = 2$, this is in two control line system. From Theorem 5, this is true because A'B' is line segment lied inside $A_1A_2B_1B_2$ or $A_1B_1A_2B_2$ Whichever formed a quadrilateral.

The averaging equation (4.11), for m control lines

$$W_n = \frac{a_1 W_1 + a_2 W_2 + \ldots + a_n W_n}{a_1 + a_2 + \ldots + a_n} \qquad (4.56)$$

W is any point on line segment AB, W_1, W_2,..., W_n are images of W on according to l_1, l_2,...,l_m. W_n is the new position on transformed line segment. $a_i = \frac{1}{d_i^2}$, d_i is the distance between W and control line l_i. So, a_i is a function of t, and $a_i > 0$.

Assume when $m = k$, in k control line system, this theorem is true. That is morphing transformed line segment lies inside the maximum polygon formed by A_1, A_2,..., A_k, B_1, B_2,..., B_k, Figure 4.12.

Suppose now $m = k+1$ control lines, l_1, l_2,...,l_{k+1}, A_1B_1, A_2B_2,..., A_kB_k, $A_{k+1}B_{k+1}$ are images of original line segment AB by l_1, l_2,...,l_k, l_{k+1}.

If we just consider first k control lines, the morphing transformed line segment lies inside the polygon formed by A_1, A_2,..., A_k, B_1, B_2,..., B_k. This is shown in Figure 4.13.

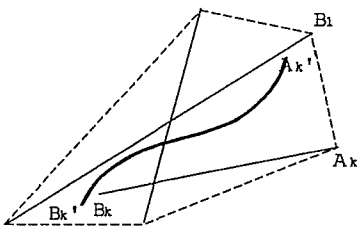


Figure 4.12 k control lines.

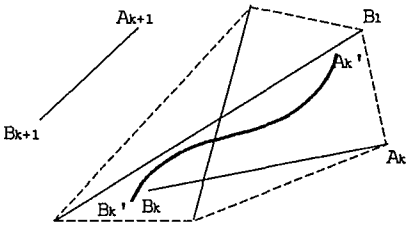


Figure 4.13 k+1 control lines.

The morphing transformed line segment by k control lines is $A'_kB'_k$, W_{nk} is the morphing transformed W.

$$W_{nk} = \frac{a_1W_1 + a_2W_2 + \ldots + a_kW_k}{a_1 + a_2 + \ldots + a_k} \qquad (4.57)$$

W_{k+1} is the image of W on $A_{k+1}B_{k+1}$. W_n is the morphing transformed W by *k+1* control lines.

$$W_n = \frac{a_1W_1 + a_2W_2 + \ldots + a_kW_k + a_{k+1}W_{k+1}}{a_1 + a_2 + \ldots + a_k + a_{k+1}} \qquad (4.58)$$

W can either represent *X* coordinate or *Y* coordinate.

Without loss of generality, assume $W_{k+1} \leq W_{nk}$

So,

$$W_n - W_{k+1} = \frac{a_1W_1 + a_2W_2 + \ldots + a_kW_k + a_{k+1}W_{k+1}}{a_1 + a_2 + \ldots + a_k + a_{k+1}} - W_{k+1}$$

$$= \frac{a_1W_1+a_2W_2+\ldots+a_kW_k}{a_1+a_2+\ldots+a_k+a_{k+1}} - \frac{a_1+a_2+\ldots+a_k}{a_1+a_2+\ldots+a_k+a_{k+1}} W_{k+1}$$

$$= \left(\frac{a_1+a_2+\ldots+a_k}{a_1+a_2+\ldots+a_k+a_{k+1}} \right) \left(\frac{a_1W_1+a_2W_2+\ldots+a_kW_k}{a_1+a_2+\ldots+a_k} - W_{k+1} \right)$$

$$= \left(\frac{a_1+a_2+\ldots+a_k}{a_1+a_2+\ldots+a_k+a_{k+1}} \right) \left(W_{nk} - W_{k+1} \right) \geq 0 \qquad (4.59)$$

And,

$$W_n - W_{nk} = \frac{a_1W_1 + a_2W_2 + \ldots + a_kW_k + a_{k+1}W_{k+1}}{a_1 + a_2 + \ldots + a_k + a_{k+1}} - \frac{a_1W_1+a_2W_2+\ldots+a_kW_k}{a_1+a_2+\ldots+a_k}$$

$$= \frac{[(a_1W_1+a_2W_2+\ldots+a_kW_k)+a_{k+1}W_{k+1}](a_1+a_2+\ldots+a_k)}{(a_1+a_2+\ldots+a_k+a_{k+1})(a_1+a_2+\ldots+a_k)}$$

$$- \frac{[(a_1+a_2+\ldots+a_k)+a_{k+1}](a_1W_1+a_2W_2+\ldots+a_kW_k)}{(a_1+a_2+\ldots+a_k+a_{k+1})(a_1+a_2+\ldots+a_k)}$$

$$= \frac{a_{k+1}W_{k+1}(a_1+a_2+\ldots+a_k) - a_{k+1}(a_1W_1+a_2W_2+\ldots+a_kW_k)}{(a_1+a_2+\ldots+a_k+a_{k+1})(a_1+a_2+\ldots+a_k)}$$

$$W_n - W_{nk} = \left(\frac{a_{k+1}}{a_1+a_2+\ldots+a_k+a_{k+1}} \right) (W_{k+1} - W_{nk}) \leq 0 \qquad (4.60)$$

That is $W_{k+1} \leq W_n \leq W_{nk}$. If assume $W_{nk} \leq W_{k+1}$, the result will be $W_{nk} \leq W_n \leq W_{k+1}$.

This means the morphing transformed W lies in the area formed by $A'_kB'_kB_{k+1}A_{k+1}$ or $A'_kB_{k+1}B'_kA_{k+1}$ whichever forms largest sub-quadrilateral, this sub-quadrilateral has three straight line sides and one curved line side.

And this sub-quadrilateral is inside the largest polygon formed by A_1, A_2, ..., A_{k+1}, B_1, B_2, ..., B_{k+1}.

Chapter 5 Video Compression Basics

Real video images and sound are powerful tools to provide a more natural interface when presenting many different kinds of information in applications such as education, training, entertainment, advertising and so on. Since late 1987, when the David Sarnoff Research Center presented its implementation of a universal all-digital medium, the expectation of low-cost, compatible, high-performance systems has increased daily.

Unlike the digital audio information, video requires a huge amount of data for representation. Thus most applications of digital video depend on compression. Image compression techniques seek to exploit various redundancies present in the signal and the perceptual limitations of the human vision in order to obtain a compact representation. This is used for both transmission and storage purposes.

Compression methods are built on both redundancies and nonlinearities in the data and the nonlinearities of

human vision. These methods exploit correlation in space for still images and in both space and time for video signals. Compression in space is known as intra-frame compression, while compression in time is called inter-frame compression. Generally, methods that achieve high compression ratios(10:1 to 100:1) are lossy in that the reconstructed images are not identical to the original.

The lossy algorithms also generally exploit aspects of the human visual system. For example, the eye is much more receptive to fine detail in the luminance (brightness) signal than in the chrominance(color) signals. So, the luminance signal is usually sampled at a higher spatial resolution.

In order to provide a standard efficient compression method, the CCITT Specialists Group on Coding for Visual Telephony, computer simulated various algorithms to code moving picture at different requirements arising from the telecommunications field. They proposed a Reference Model (CCITT-RM8)[52]. This algorithm is aimed at the bit-rate range of 64kbps to 2Mbps. Each frame of video

consists of the luminance (Y) and the chrominance (Cb and Cr) components. The luminance signal has a (noninterlaced) resolution of 352x288, and the chrominance signals are subsampled 2:1 in both spatial directions with respect to the luminance (they are thus 176x144). CCITT later converted the RM8 model into Recommendation H.261.

The International Organization for Standardization (ISO)- MPEG Group works for the definition of the standard coding algorithm for moving images based on CCITT Recommendation H.261. This coding algorithm is prepared to provide not only the Normal Video Playback but also the Reverse Video Playback, the Fast Forward Video Playback, the Fast Reverse Video Playback, Random Access and High Quality Sill Mode.

MPEG REQUIREMENTS AND THEIR IMPLICATIONS

- Normal playback: This is the standard playback mode where video is decoded and displayed in the forward direction. As is typical with other compression

systems, the picture quality in this mode is of prime importance.

- Random access: This requirement concerns the fast access and reconstruction of an arbitrary video frame from the bit-stream. This feature requires "intra-frames" be inserted into every so many frames in order to provide random access points.
- Reverse playback: This feature enables the frames to be reconstructed and displayed in reverse order, at normal speed.
- Fast search: This is the capability of displaying the video at 8 to 10 times the normal speed in both forward and reverse directions.
- High-resolution still frame: Certain frames are required to be encoded at much higher spatial resolution than normal, and reconstructed on demand.

The current proposed MPEG compression algorithm has been described in[58]. The MPEG algorithm breaks a video sequence into a series of short segments, which are processed independently. Each segment, which typically contains fifteen frames, is known as a group of frames

(GOF) contains three types of frames; intra-frame, predicted and bidirectionally predicted (also known as interpolated frames).

Basic MPEG Algorithm

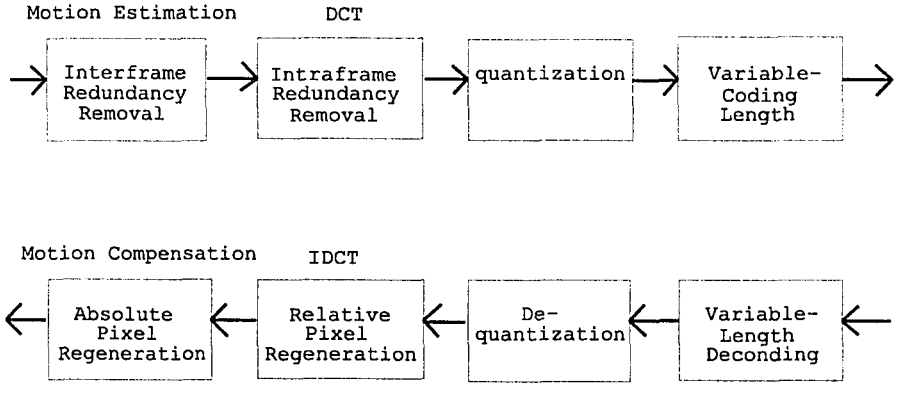


Figure 5.1 Basic MPEG Algorithm

The first frame in a GOF is coded in intra-frame (I) mode, i.e. independent of any other frames. The coding of an I-frame begins by subdividing the luminance component into 16x16 blocks and chrominance components into 8x8 blocks. Taken together, the 16x16 luminance block and two 8x8 chrominance blocks are known as a macroblock(MB). The luminance portion of a MB is then

further partitioned to four 8x8 blocks, and all six (four luminance and two chrominance) 8x8 blocks in the MB are transformed using the two-dimensional Discrete Cosine Transform(DCT). The DCT coefficients of each block are ordered according to a zig-zag scan and quantized. For the DC coefficient, a fixed quantization to 8 bits is performed. Each DC coefficient is then predicted using the value of the previous DC coefficient, and the prediction error is entropy coded. The AC coefficients are grouped together and then entropy coded.

Following the I-frame, every M-the frame is coded in predictive (P) mode (typically, M=3). The first P-frame is predicted using the I-frame, and subsequent P-frames predicted using the previous P-frame. Each MB in a P-frame can be coded in one of several modes: predictively, predictively with motion-compensation, or, if the prediction is not accurate enough, in intra-frame mode. In the predictive case, a prediction error macroblock is formed and the DCT coefficients of each 8x8 block quantized. The coding of intra-frame macroblocks is the same as that for I-frames. For each MB, a parameter

called the MB Type indicates which mode is used. Motion vector data MV1 is coded for a particular MB only if the motion-compensated prediction mode is selected. The MB type can also convey information relating to MB specific quantization parameters and other special cases. If the motion vector for a given MB is zero, and if all of the DCT coefficients are zero after quantization, the MB is skipped. In many cases, the DCT coefficients of some, but not all, of the blocks within a macroblock are entirely zero. To handle these cases, a coded block pattern (CBP) parameter is used to tell which of the six blocks have nonzero data, and only those blocks are actually coded.

Between P-frames, M-1 frames are coded using a bidirectional predictive mode (B). Coding modes allowed for B-frames include motion compensation using prediction from either the previous P-frame (forward prediction) or the next P-frame (backward); motion compensated interpolation using a weighted average of the previous and next P-frame (interpolative); and intraframe mode. Macroblock type and address and CBP parameters serves

functions similar to that in the P-frame case. If motion compensation is used, the MB type indicates whether the prediction is forward, backward, or interpolative. Only those motion vectors necessary to reconstruct the frame are actually coded. Note that in interpolative mode, up to two sets of motion vectors, MV1 and MV2, may have to be coded. In B-frames, a MB is skipped if all DCT coefficients are zero and the MB type and motion vectors are the same as those of the previously coded frame.[44]

Even though the coding of the Intra-frames is conceptually and computationally the simplest type of coding operation performed in the system, it turns out that the quality of the intra-frames critically effects the overall quality of the reconstructed video.

5.1 DCT and IDCT

Source image samples are grouped into 8x8 blocks; each sample has a value in the range of $[-2^{p-1}, 2^{p-1}-1]$, where p is usually 8. Each block is an input to the Forward DCT(FDCT). The output of FDCT can be input to Inverse DCT(IDCT) to obtain the original blocks to

reconstruct the original source image samples. The equations of 8x8 FDCT and 8x8 IDCT are as follows:

$$F(u,v)=\frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7}f(x,y)*\cos\frac{(2x+1)u\pi}{16}\cos\frac{2y+1v\pi}{16}\right] \qquad (5.1)$$

$$f(x,y)=\frac{1}{4}C(u)C(v)\left[\sum_{u=0}^{7}\sum_{v=0}^{7}C(u)C(v)F(u,v)*\cos\frac{(2x+1)u\pi}{16}\cos\frac{2y+1v\pi}{16}\right] \qquad (5.2)$$

where C(u), C(v)=1/$\sqrt{2}$ for u, v=0; C(u), C(v)=1 otherwise.

5.2 Block Matching Algorithm(BMA)

Block Matching Algorithms are used to determine the motion vector. In BMA, the block of pixels of size (MxN) is compared with a corresponding block within the same search area of size (M+2p x N+2p) in the previous frame (Figure 5.2) and the best match is found based on cost function such as minimum MSE. In MPEG, M N are set to be 16, and p is 7.

There are a number of fast algorithms developed for Block Matching for the estimation of motion on a block by block basis. In the paper titled "The cross-search algorithm for motion estimation", Ghanbari offers a fast

block matching algorithm.

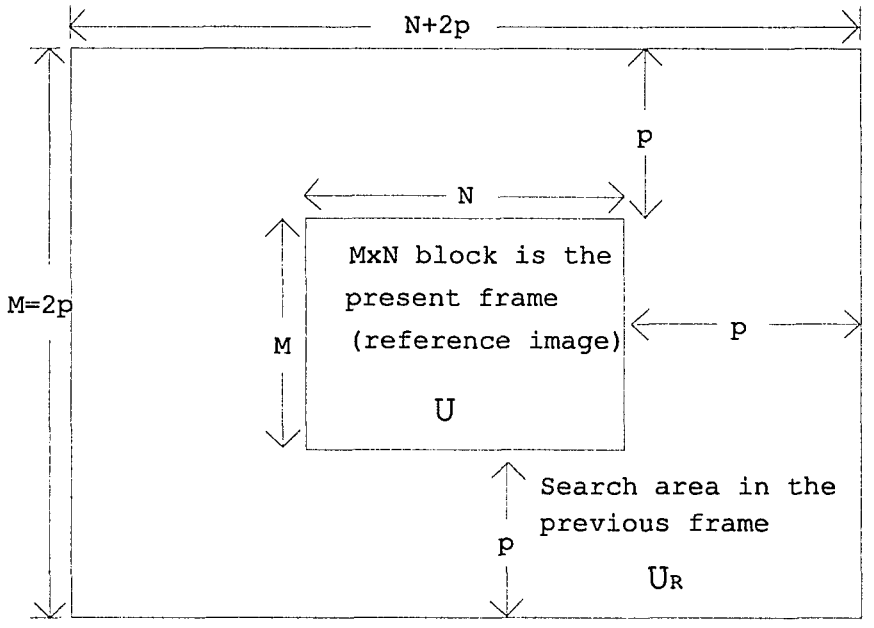


Figure 5.2 Geometry for manipulation in MxN reference image U with (M+2p)x(N+2p)image U_R(search area) in the previous frame.

To locate the best match by full search, $(2w+1)^2$ evaluations of the matching criterion are required. To reduce the computational complexity, Jain and Jain [54] use a two-dimensional logarithmic search method (TDL) to track the direction of a minimum mean squared error distortion measure.

Koga et al [49] use a three-step motion vector direction search (TSS) to compute displacements up to 6 pixels/frame. In Kappagantula and Rao's [53] modified motion estimation algorithm (MMEA), prior to halving the step sizes, two more positions are also searched. Puri et al [51] have introduced the orthogonal search algorithm (OSA) where with a logarithmic step size, at each iteration 4 new locations are searched.

5.3 Quantization

MPEG quantizes the DCT coefficients after the DCT transform. Quantization combined with run-length coding contribute to the overall compression efficiency of MPEG.

Intra- and inter-frame are quantized differently. Intraframe blocks contain high energy in all frequencies and are likely to produce "blocking effects" if too coarsely quantized; on the other hand predictive frame blocks mostly contain high frequency blocks can be subject to a much coarser quantization.

Three forms of quantization can be used for this

purpose.

1) Zero-memory quantization

2) Block quantization

3) Sequential quantization

5.4 Entropy coding

After quantization, quantized values are coded using a lossless redundancy reduction method. Most MPEG applications use Huffman coding with probability distributions embedded in the Huffman code itself, with more probable codes associated with shorter codewords. Dynamic Huffman coding is avoided to reduce the complexity of the overall algorithm even though certain improvement could have achieved by its use.

In a paper titled "Encoding of motion video sequences for the MPEG environment using arithmetic coding" [44], Vescito and Gonzales offers the use of arithmetic coding to achieve somewhat better results. In their solution they take advantage of the fact that arithmetic coding can easily be made adaptive and perform the probability estimation in parallel with the coding.

This eliminates the need to estimate the probability distributions ahead of time by collecting data on the statistics of either the sequence to be encoded or a representative source and adaptive approach offers more precise probability distribution values thus resulting in shorter codewords.

Chapter 6 Morphing for Motion Interpolation

Motion compensated image compression takes advantage of the similarities between consecutive image frames and provides a means to only store the blocks of pixels that have changed their value since the previous frame. Because of operations like DCT the transform, quantization and entropy coding as well as the inverse of these operations, motion compensation based MPEG is a lossy image compression algorithm. In this chapter another image compression method for motion pictures, morphing for motion interpolation (MMI) is introduced. The MMI method makes use of image distortion to achieve the compression of the motion fields of the image, and does not "touch" the non-motion fields. The pixels of the motion fields are reconstructed by mathematical equations and control line positioning. The pixels values are weighted averages. The MMI is an intelligent image compression.

6.1 Principle of MMI

High image compression of motion pictures usually make use of interpolation techniques. The purpose of interpolation is to determine skipped frames from neighboring frames. Morphing interpolation compares two "end" frames, and interpolates only motion fields of the interpolated image frames. Different from linear interpolation and motion compensated interpolation, morphing interpolation first linearly interpolates the corresponding control lines in the interpolated frames over time fraction between the first "end" to the second "end" frames. Then it takes the motion fields in the interpolated frames as they are in the "end" frames. In the interpolated frames, pixels of the motion fields are generated by computing and averaging a few (based on the number of control lines) pixels in each "end" frame to find out the position of this pixel in both "end" frames according to control line changes. The morphing interpolation dissolves the color information of the two pixels in the "end" frames over time fraction from the first "end" to second "end" frames. Compared with MPEG, the interpolated frames (B frame) and the predicted frames (P frame) are generated in a different way in MMI

using the two reference frames (I frames of this GOF and next GOF), motion field, and control line positions.

Similarities to MPEG

The I frames are coded and decoded in the same way as in MPEG to store or transmit. The DCT transform, the IDCT transform quantization, dequantization, entropy coding, and entropy decoding are used to remove spatial redundancy.

Differences to MPEG

In MMI with each I frame, control line positions are also coded, stored or transmitted. The B frames and P frames are not used. Instead of B and P frames, the MMI method interpolates M frames between adjacent I frames. M stands for morphing. M frames are generated computationally between two adjacent I frames using only the information from these two I frames.

An illustration of how the pixels in a motion field

are generated by the morphing interpolation procedure, assuming two control lines are used is given next.

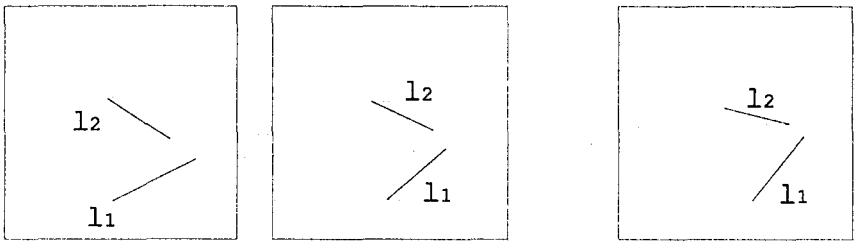


Figure 6.1 Interpolation of control lines in M frames.

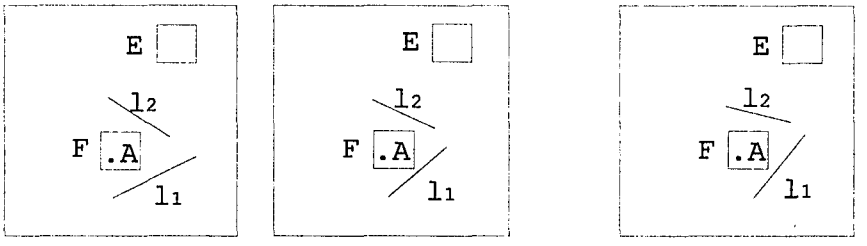
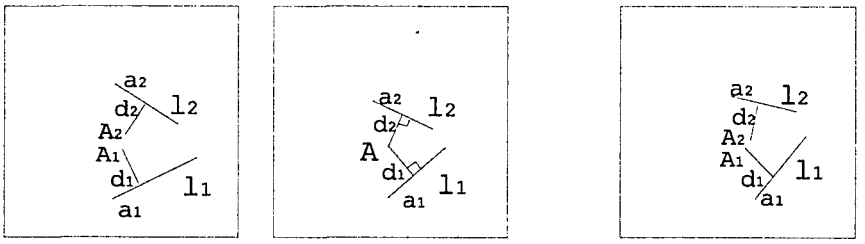


Figure 6.2 Determination of motion status of MBs.

Figure 6.1 shows two adjacent I frames (a), (c), and one morphing interpolated M frame (b). Since the control line position l_1 and l_2 are coded with each I frame, (a) and (c) contain their own control lines. In the M frame (b) control lines l_1 and l_2 are generated automatically according to the position of this M frame in the sequence of M frames interpolated between the I frames.

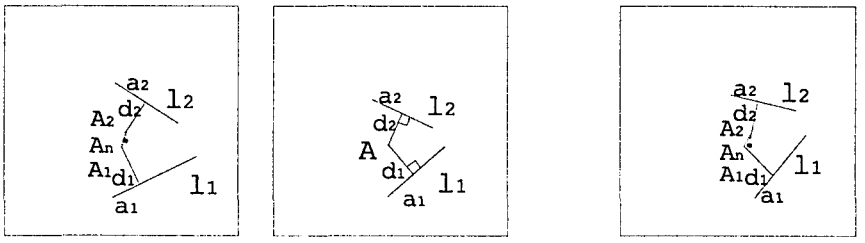
If microblock(MB) E is marked as non-motion field in

the I frame (a) and (c) recover E in (b) from the E of either (a) or (c) or average of (a) and(c), see Figure 6.2. Even though E is marked non-motion field, it does not mean E in (a) and (c) are exactly identical. It means only within a certain error tolerance, they are regarded as the same.



(a) (b) (c)

Figure 6.3 Morphing step 1 using reverse warping.



(a) (b) (c)

Figure 6.4 Morphing step 2 using reverse warping.

If an MB F is marked as a motion field, the pixels in F will be generated by the morphing(MMI). If A is a pixel of F, find A's relation to each control line. a_1 a_2 are the fractional distances along the control line l_1 l_2, and d_1 d_2 are the distance to the control line l_1 l_2.

Then find the pixel A_1 of l_1 and A_2 of l_2 in (a) and (c) which keep the same relation as A to l_1 and l_2 . Then average A_1 and A_2 in (a) and (c) using the equation suggested in Chapter 3 to obtain in the morphing transformed pixel A_n.

A_n is the morphing transformed pixel in (a) and (c). A_n contains the position information and the color information of A.

A_n does not have to be in microblock F. It may be in F or in another adjacent motion field microblocks. For instance if a ball moves, the control lines move with it. The MB contains the ball may be in left MB of F in (a) and in right MB of F in (b). This is how morphing works for motion field instead of motion compensation. It is called intelligent interpolation.

The next step, linearly average (dissolve) the color information of A_n in (a) and (c), using the equation (3.8)-(3.10) or (3.11)-(3.13). The weighting factors are percentages of time distance of the current M frame

position each referenced I frames. If M is close to (a), A is weight more color information of A_n in (a), and vice versa.

The advantage of MMI method is in its simplicity. If one frame is coded every N frames, N-1 frames are totally "skipped", so in the MMI method by itself implies an order of 1/N compression. All the transmitted/stored frame are coded the same way, intra-frame coded with control line positions information. Second, it has less computation, because there is no DCT, IDCT, quantization, dequantization, entropy coding, entropy decoding in generating interpolated M frames. The motion MB pixels are reconstructed by averaging a few (the number of control line) pixels from two reference I frames.

Finally, there are alternate way in determining the MB status, motion field or non-motion field. It can be done on either the encoder side or decoder side. If encoding rate is important, the MB status is determined on decoder side, otherwise it is done on the encoder side.

This is an example how MB status is encoded. Assume status=0 stands for non-motion field status, status=1 stands for motion status. Two bits are needed to represent its comparison to the previous I frame and later I frame. One MB contains on luminance MB 16x16 and two chrominance MB 8x8. The luminance portion of a MB is further partitioned into four 8x8 block. All six (four luminance and two chrominance) 8x8 MB are DCT transformed, quantized, and entropy coded with other two bits as follow.

x=0 no motion, compare with previous I frame
x=1 motion field, compare with previous I frame
y=0 no motion, compare with later I frame
y=1 motion field, compare with later I frame

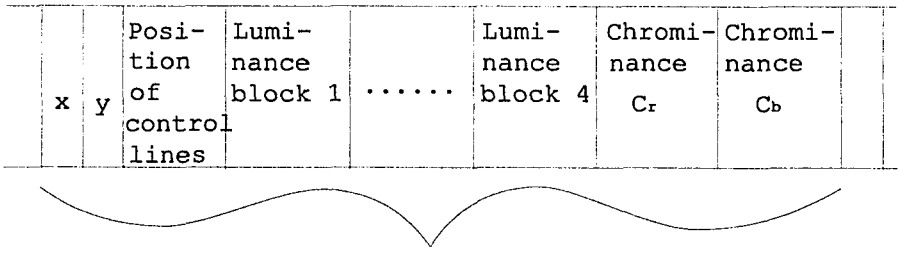


Figure 6.5 Coding structure of MB.

6.2 Encoder and Decoder

MMI method requires automatic control line placement and therefore knowledge of objects within the image and is not covered in this study. Control line positions for each encoded frame can be determined by using fuzzy logic.

There are two versions of MMI encoder and decoder. Version one determines the MB status at encode side; version two determines the MB status at decoder side. Figure 6.6 and 6.7 show the version 1 of MMI encoder and decoder.

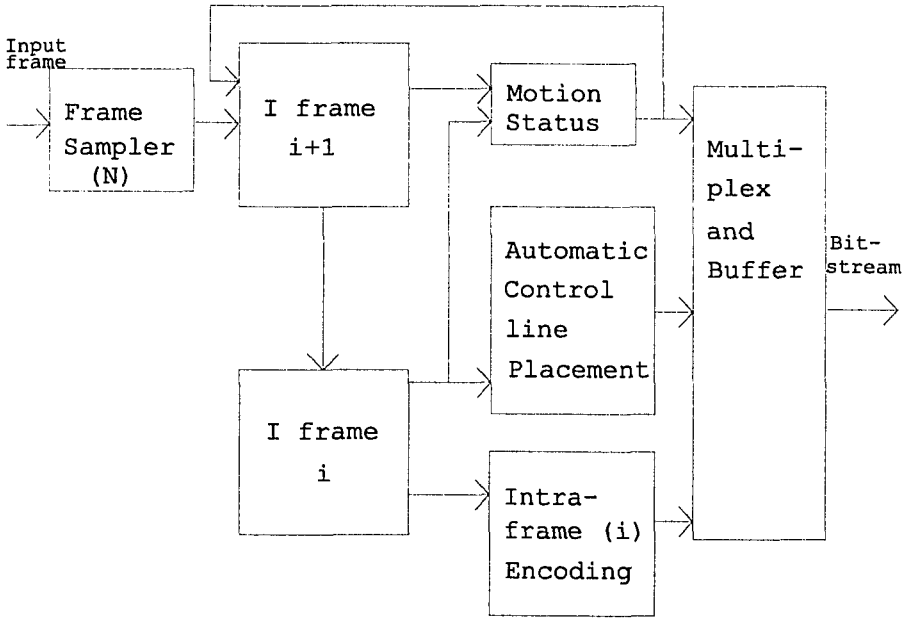


Figure 6.6 MMI encoder version 1

Frame sampler (N) takes one frame every N frames, drops the other N-1 frames. Two image frames i, i+1 are stored in the memory at the same time, the i+1th frame compares with its previous the i-th frame to get the motion status, save it at the x position of its own frame - the i+1th frame, see Figure 6.5, also save it at x position of the previous frame - the i-th frame. Automatic control line placement unit places the control lines or modifies the control line positions of i-th frame. Intra-frame encoding block encodes the MBs of I frame, including DCT transform, quantization, and entropy encoding. Then combination of the motion status information for each MB, control lines information for i-th frame, and Intra-frame encoded i-th frame is perform in multiplex and Buffer block to form the bitstream of the MMI encoded i-th frame.

The MMI decoder (version 1) is shown in Figure 6.7. Intra-frame decoder decodes the bitstream into an I frame, it decodes each MB the same way as MPEG decode the MB of an I frame, including IDCT transform,

dequantization, and entropy decoding. Motion status x in later frame and motion status y in previous frame are compared to ensure the correct decoding. Then MMI block uses control line information, motion status information

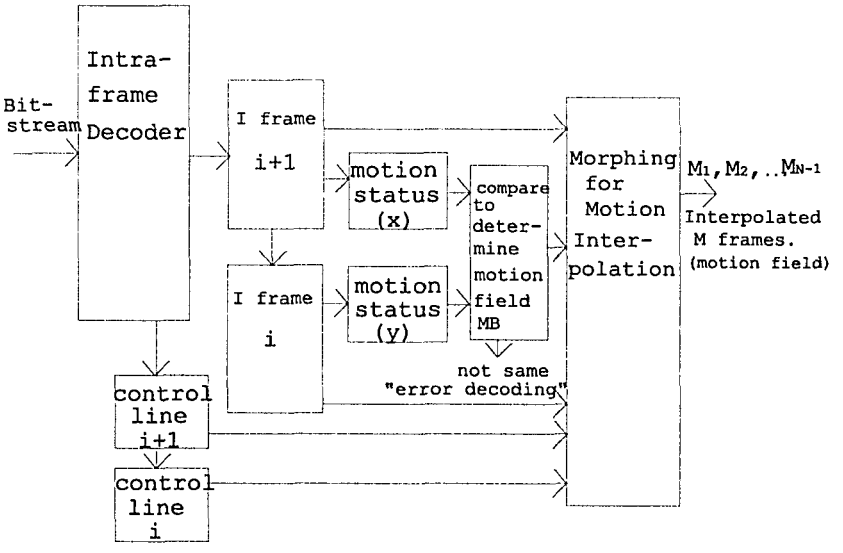


Figure 6.7 MMI decoder version 1.

and two I frame, frame i and i+1 to generate the motion field MBs for N-1 M frames. Combine the non-motion field MBs from either i frame, or i+1 frame, or average of these two frames to generate the complete M frames.

MMI encoder(Version 2) and decoder perform the similar operation as version one. The only difference is

the motion status, which are derived on the decoder side. This design decreases encoding time, and increases decoding time. It also increases compression ratio a little, because the motion status is not encoded. (Version 2) MMI encoder and decoder are shown in Figure 6.8 and Figure 6.9.

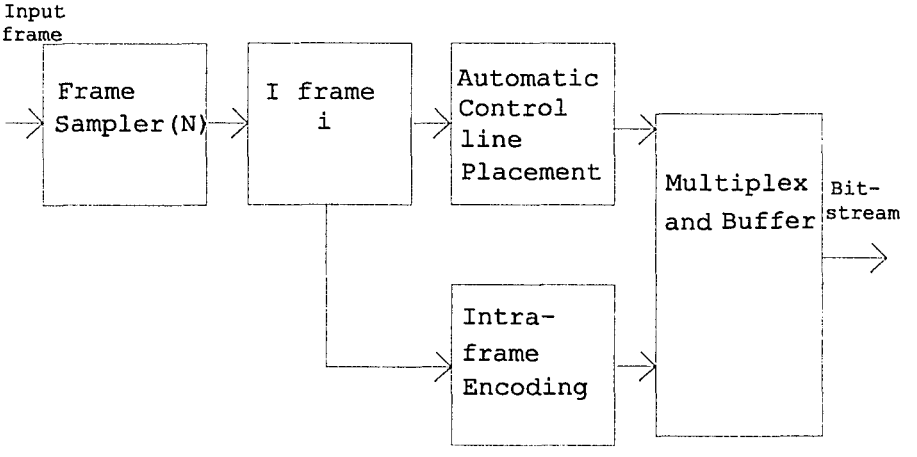


Figure 6.8 MMI encoder version 2

6.3 Discussion of the performance

This discussion is only on MMI encoder and decoder version one. The performance of version two is similar

to version one and straightforward.

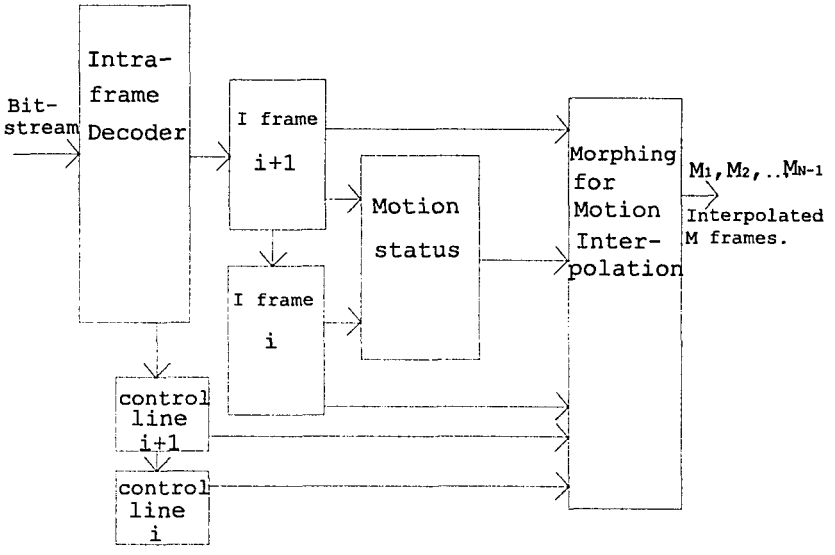


Figure 6.9 MMI decoder version 2

6.3.1. Forward play

The video sequence is reconstructed in units of the N-frame GOF. First i-th and i+1-th I frame are decoded and stored in the frame-memory. The i-th frame is played, then the motion fields of first M frame is generated by MMI, and it replaces the motion fields of i-th I frame. Next, motion field of the second M frame is generated by MMI and it replaces the motion fields of the first M frame. And the motion fields of third M

frame is generated by MMI... and so on, until the motion fields of the N-1th M frame is generated by MMI, and it replace the motion fields of the N-2th M frame. Then i+2th I frame is decoded and stored in the frame memory to replace i-th I frame. After i+1th I frame is played and the same procedure is repeated.

6.3.2. Reverse play

The reverse play is the same to forward play, when I frames are played in reversed order and x field and y field of each MB are exchanged. Because from i-th I frame i+1 I frame, and from i+1th I frame to i-th I frame, morphing generate the same sequence of M frames with different.

6.3.3. Random access

To access a desired frame, first the I frame is determined. Then this I frame and next I frame are decoded and store in the frame memory, take this frames as the first frame and the next frame as the second

frame, start forward play.

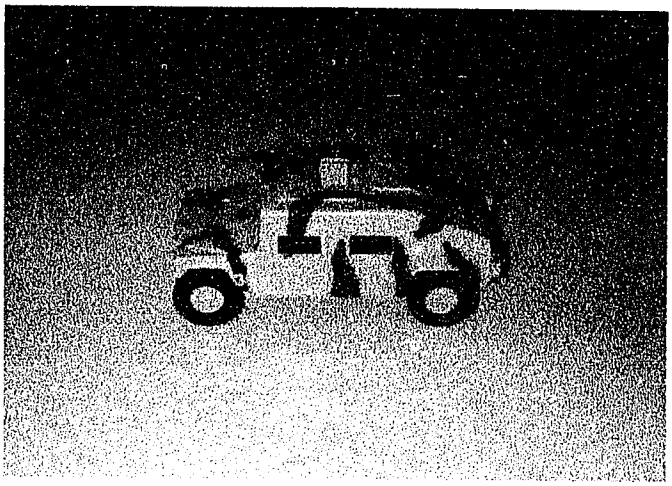
4. Fast forward/reverse

Fast forward/reverse depends on the number N, the frame number of GOF. If N=9, only I frames are played, and each I frame is played nine times. By using motion status, only motion fields of I frame are played. This is nine times faster forward/reverse. If N=19, only the motion fields of I frames and one M frame of each GOF are played. This is also nine times faster forward/reverse.

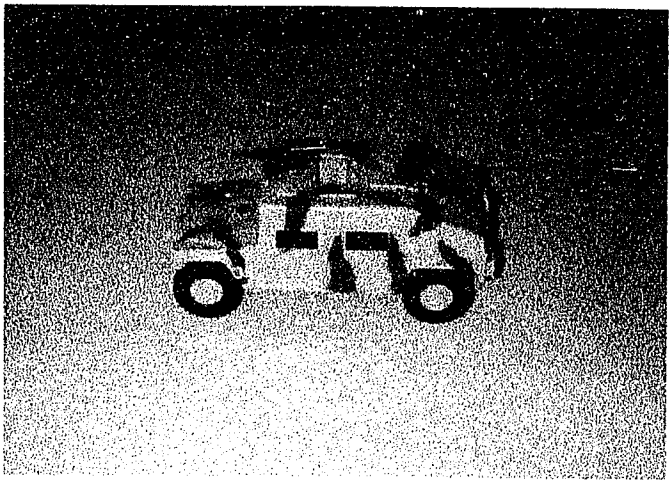
Appendix

Some morphing examples are given here by using the software WinImage: Morph. There are three sets of pictures. Each set contains fourteen images. Only the first and the fourteenth exist. The first picture is used as source image, and the fourteenth picture is used as target image. The other twelve interpolated images are generated by morphing.

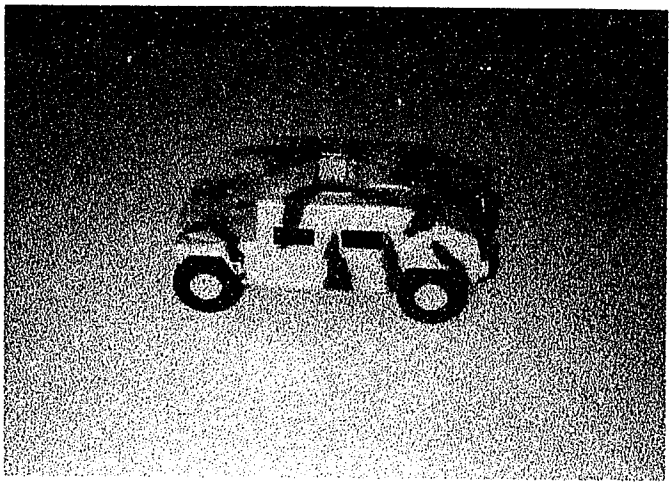
Pic 1



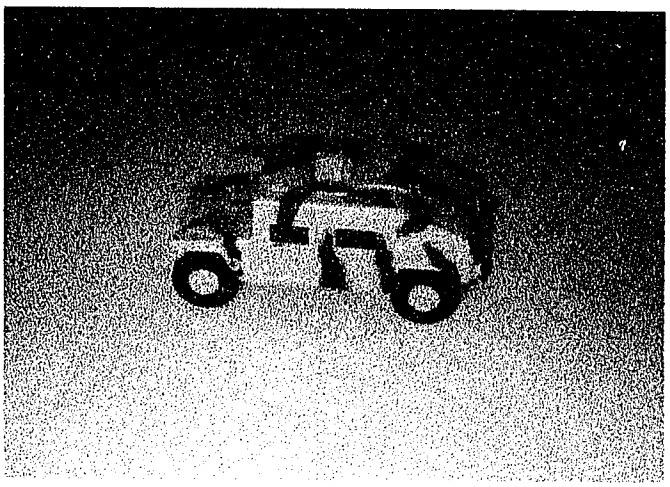
Pic 2



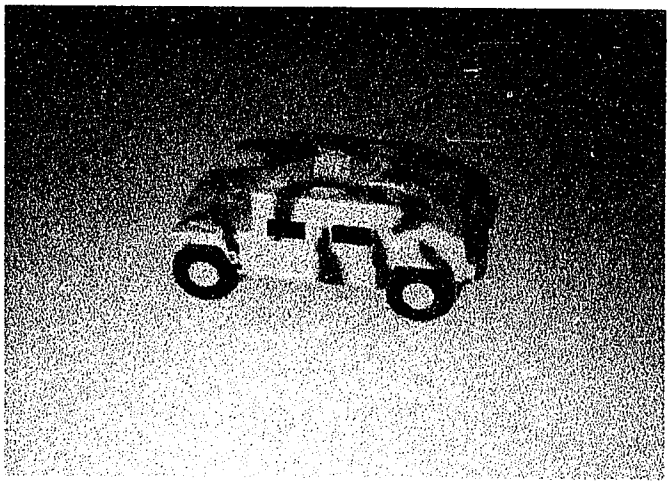
Pic 3



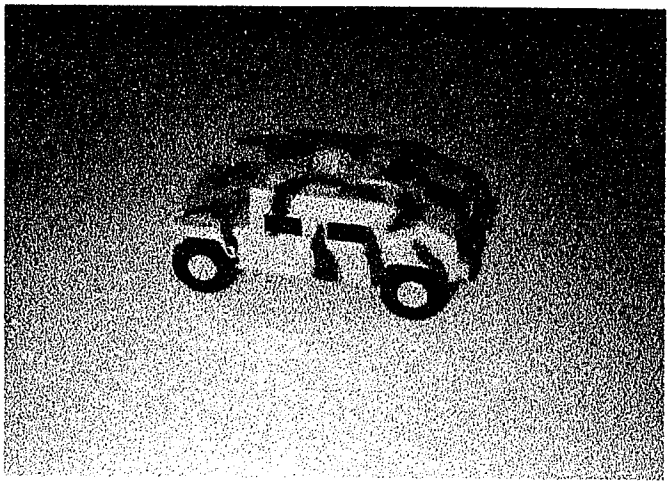
Pic 4



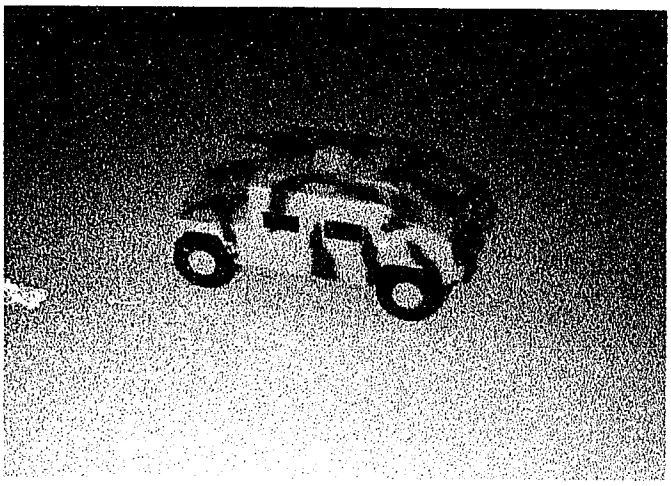
Pic. 5



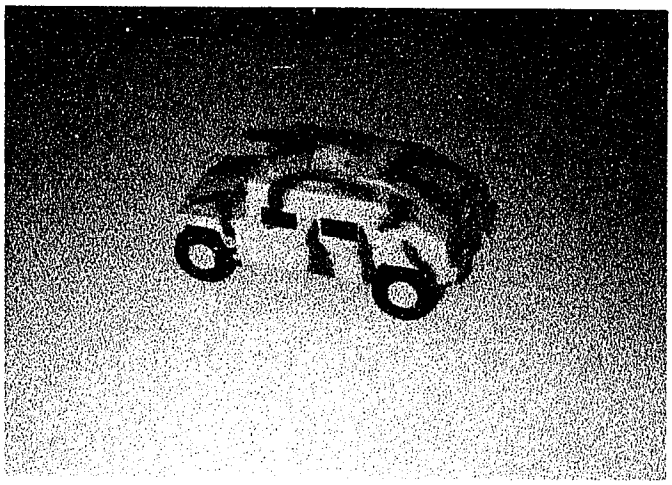
Pic 6



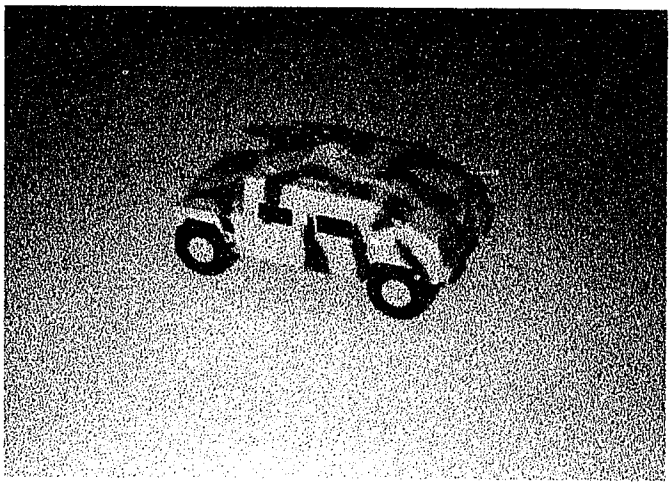
Pic 7



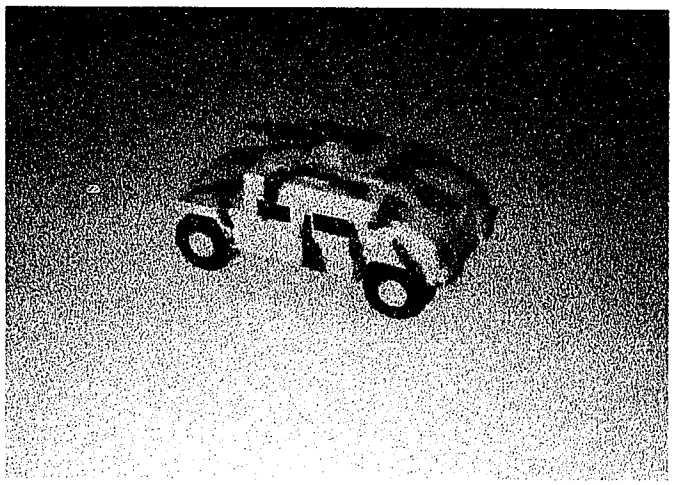
Pic 8



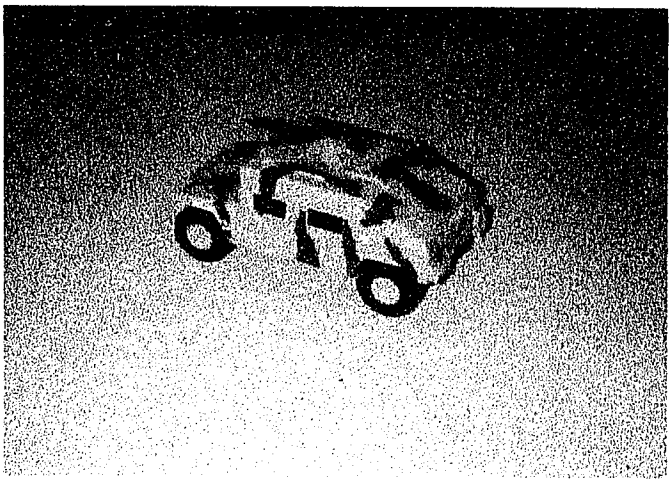
Pic 9



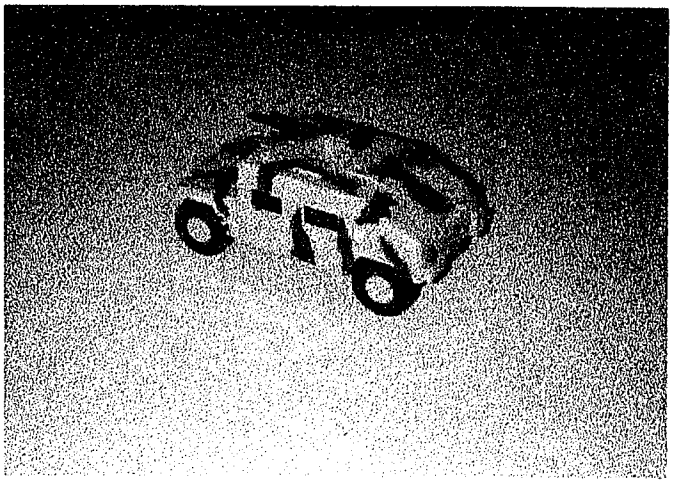
Pic 10



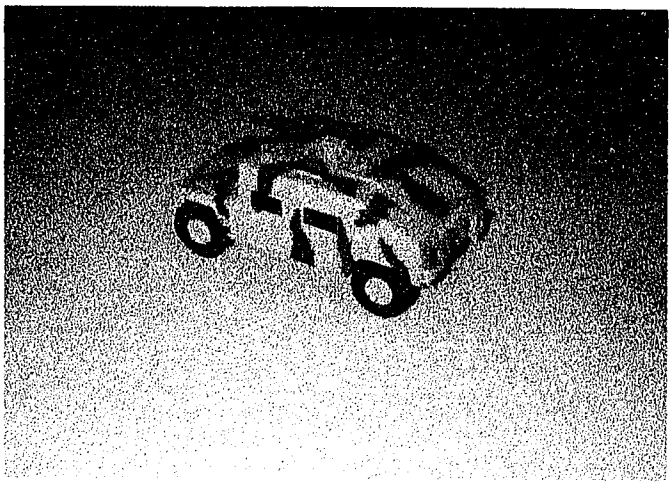
Pic 11



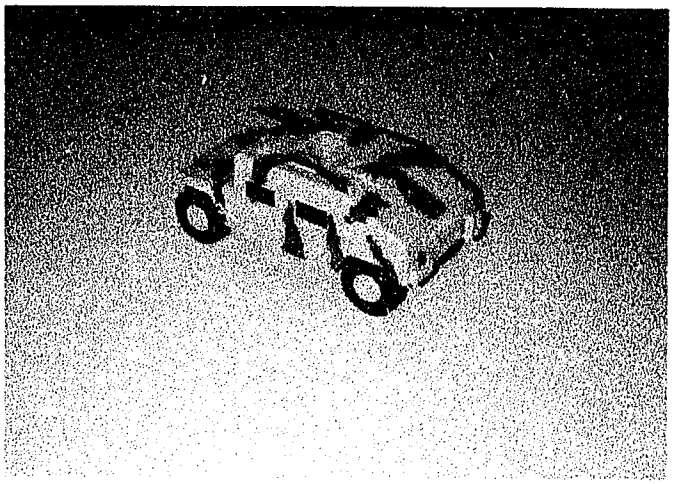
Pic 12



Pic 13



Pic 14



Moving Tank Pic. 1 - 14

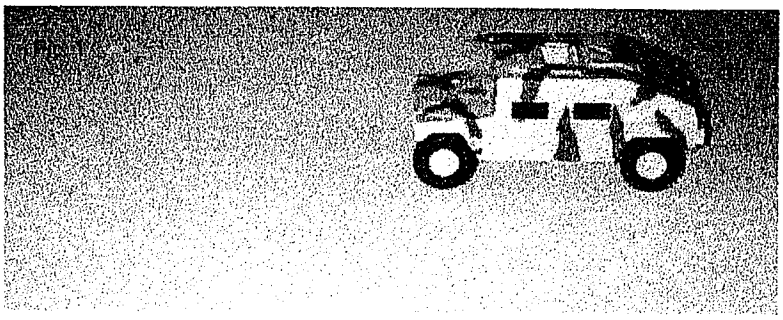


Fig. 1

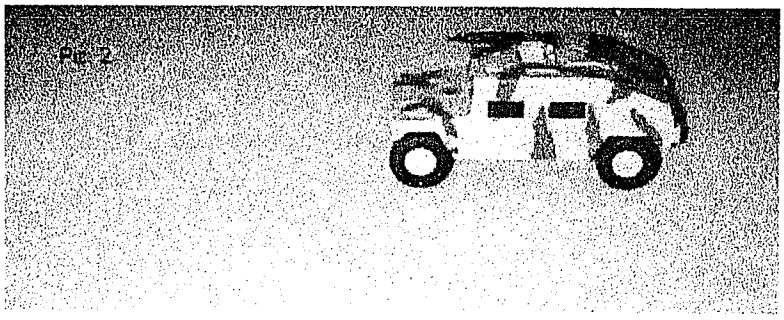


Fig. 2

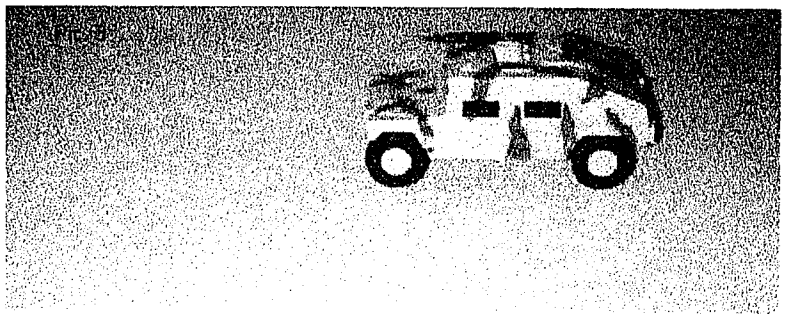


Fig. 3

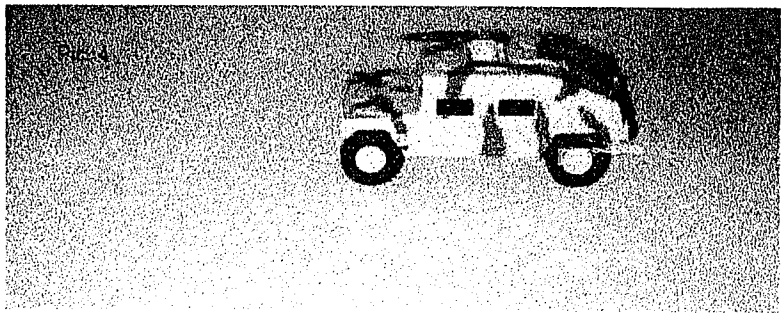


Fig. 4

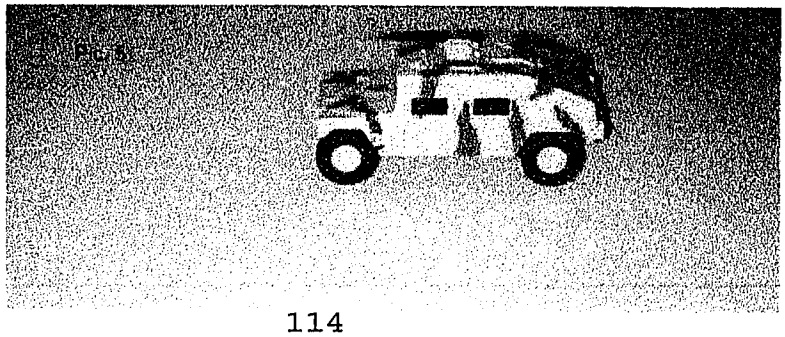
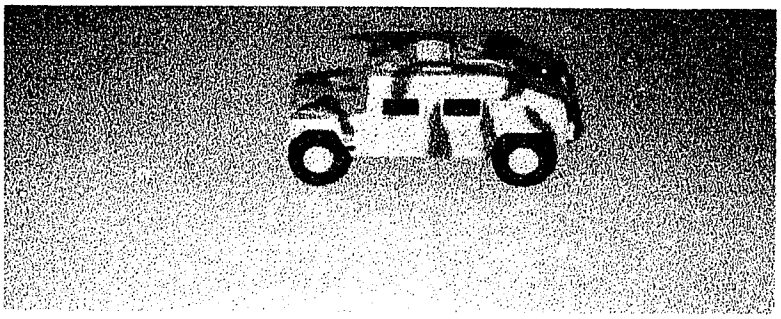
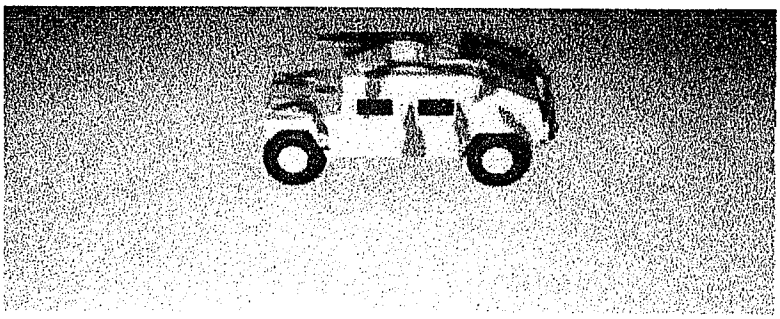
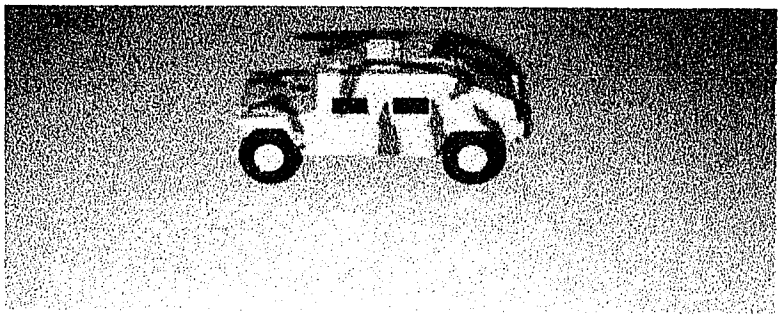
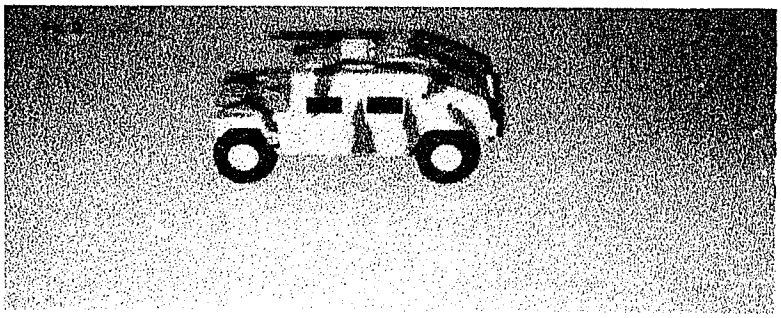


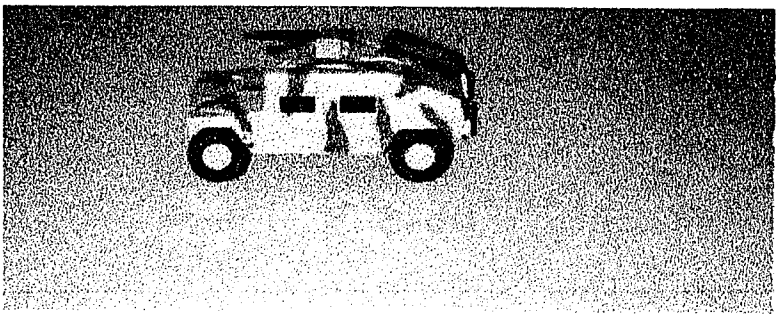
Fig. 5

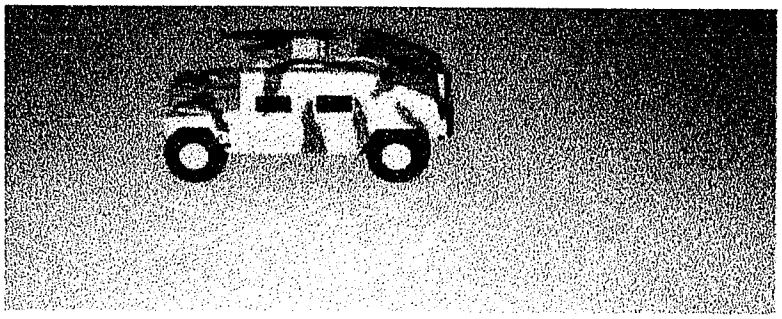


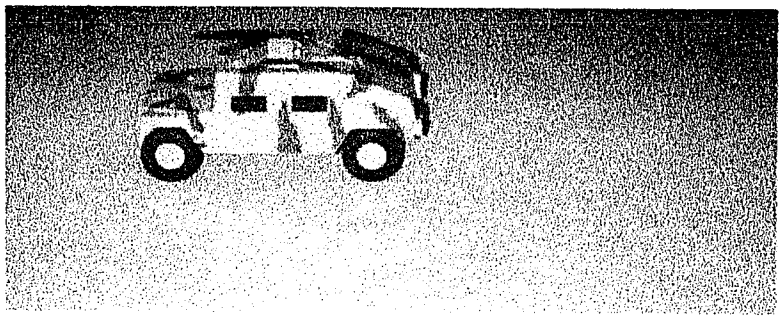


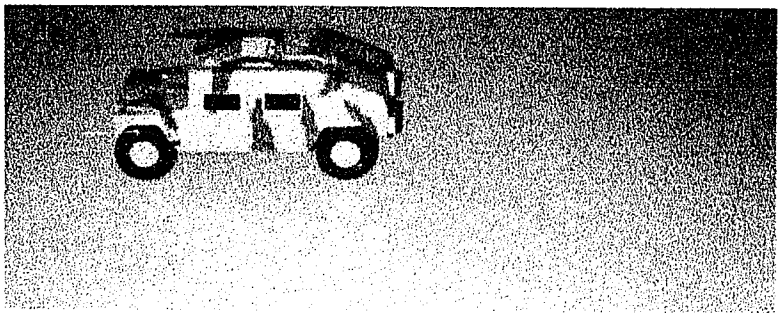


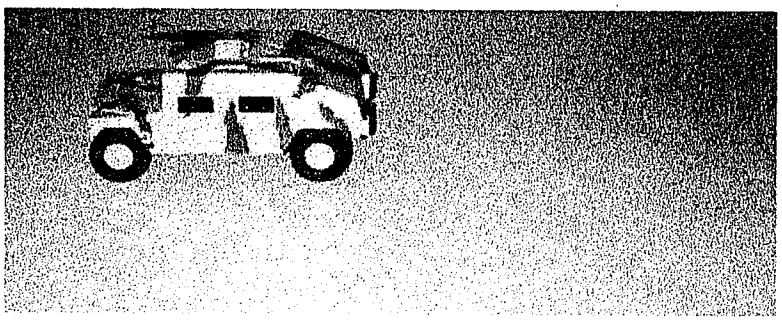




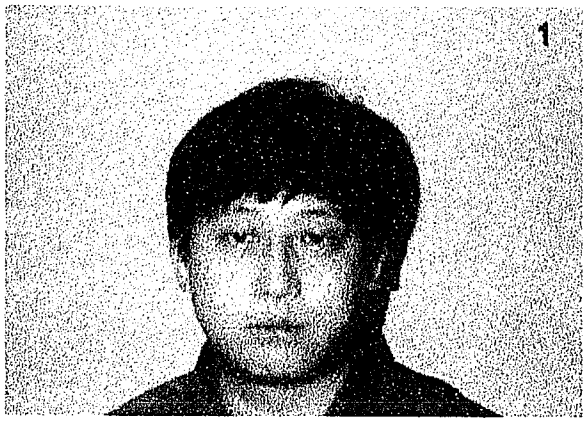




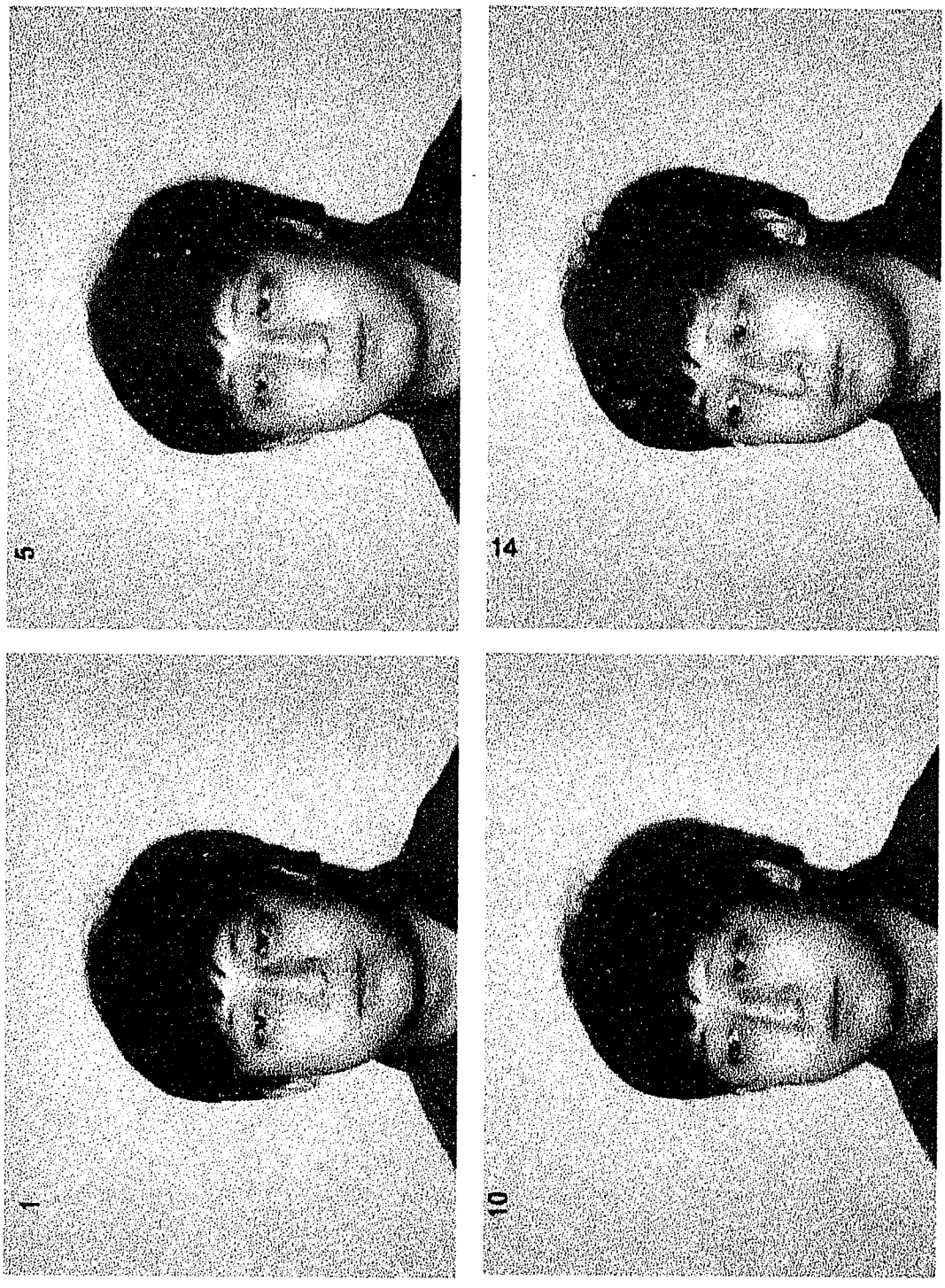




Pic. 1 - 7



Pic. 8 - 14


5
14
1
10

REFERENCES

1. Scott Anderson, *Morphing Magic*. Sams Publishing, Carmel, Indiana, 1993. ISBN 0-672-30320-5.

2. C. Watkins, A. Sadun and S. Marenka, *Modern Image Processing: Warping, Morphing, and Classical Techniques*. Academic Press Professional, Cambridge MA, 1993. ISBN 0-12-737860-X.

3. George Wolberg, *Digital Image Warping*. IEEE Computer Society Press, Los Alamitos CA, 1990. ISBN 0-8186-8944-7.

4. David F. Rogers and J. Alan Adams, *Mathematical Elements for Computer Graphics*, 2nd edition. McGraw-Hill Publishing Company, New York, 1990. ISBN 0-07-053529-9.

5. Thaddeus Beier and Shawn Neely, *Feature-Based Image Metamorphosis*. Computer Graphics, Vol.26, No.2, July 1992. pp. 35-42.

6. John F. Hughes, *Scheduled Fourier Volume Morphing*, Computer Graphics, Vol.26, No.2, July 1992. pp. 43-46.

7. James R. Kent, Wayne E. Carlson and Richard E. Parent, *Shape Transformation for Polyhedral Objects*, Computer Graphics, Vol.26, No.2, July 1992. pp. 47-54.

8. Leonidas Guibas and John Hershberger, *Morphing Simple Polygons*, Proceedings of the Annual Symposium on Computational Geometry, 1994. pp. 267-276.

9. Shenchang Eric Chen and Lance Williams, *View Interpolation for Image Synthesis*, Proceeding ACM SIGGRAPH 93 Conference on Computer Graphics, 1993. pp. 279-285.

10. Cliff Gromer, *Morphing It*, Poplar Mechanics, October 1992. pp. 54-55.

11. Mike Heck, *ElasticReality Transforms Images with Intuitive Drawing Process*, InfoWorld, Jan. 17, 1994. Vol.16, No.3. p. 95.

12. Glenn M. Lewis, *Morphing 3-D Objects in C++*, Dr. Dobb's Journal, July 1994. pp. 18-22.

13. Philip J. Benson, *Morph Transformation of the Facial Image*, Image and Vision Computing, December 1994. Vol.12, No.10. pp. 691-696.

14. David Pogue, *Elastic Reality 1.0*, MacWorld, Feb. 1994. Vol.11, No.2. p. 64.

15. Rick Broida, *MetaMorf*, Compute, July 1994. Vol.16, No.7. pp. 101-102.

16. David Pogue, *Morph 2.0*, MacWorld, April 1994. Vol.11, No. 4. p. 79.

17. Thomas Marshall, *Morphing Transforms Ordinary Images*, InfoWorld, Nov. 1993. Vol.15, No.45. p. 121.

18. Richard Corliss, *They Put The ILM In Film*, Time, April 13, 1992. pp. 68-69.

19. Joshua Quittner, *A Mighty Morphing*, Time, January 16, 1995. p.56.

20. Deborah Erickson, *Machine Vision*, Scientific American, July 1992. p. 112.

21. Marcelle M. Soviero, *Changing Faces: Morphing Software Comes To PCs*, Popular Science, September 1993. p. 43.

22. Guy Gracia, *Make Sticky, Morph!* Time, July 8, 1991. p. 56.

23. David A. Kaplan, *Lights! Action! Disk Drives!* NewsWeek, July 22, 1991. p. 54.

24. A.R. Smith, *Planar 2-Pass Texture Mapping and Warping*, Computer Graphics, July 1987. Vol.21, No.4. pp. 263-272.

25. G. Wolberg, *Skeleton Based Image Warping*, Visual Computer, March 1989. Vol.5, No.1/2. pp. 95-108.

26. Steven Anzovin, *WinImages: Morph.* Compute, Dec. 1993. Vol.15. p. 122.

27. Tom Soter, *The Stricks of 'Terminator 2'.* Video, Jan. 1992. Vol.15. p.12.

28. Stewart Weiner, *Morphing: How T2's Hit Man Shifted Shape.* TV Guide, Aug. 1-7 1992. Vol. 40. pp. 19-20.

29. Peter H. Lewis, *Now You See It, Now You Don't,* New York Times, April 4, 1993. Vol.40, No.3.

30. Ed. Will, *Morphing into Super Mario Land,* Denver Post, April 16, 1993. Vol.20, No.1.

31. John Stanley, *The Artist Who Make 'Morphing' Real.* San Francisco Chronicle, March 25, 1993. Vol.1, No.2.

32. Mark Potts, *Jurassic Desktop: Special effects by PC,* Washington Post, Aug. 29, 1994. Vol.15, No.1.

33. Woody Hochswender, *When the Camera Lies.*, New York Times, June 21, 1992. Vol.8, No.3. p. 9.

34. Eileen O. Grady, *Latest Commercial for Blue Bell Features Singing Cow.*, Houston Post, March 16, 1992. Vol.11, No.1.

35. Mustafa T. Avcilar, *Scaled Differential Image Compression for Videoconferencing Applications*, Ph.D dissertation. City University of New York, 1994.

36. A. Puri, R. Aravind, B.G. Haskell and R. Leonardi, *Video Coding with Motion-Compensated Interpolation for CD-ROM Applications.*, Signal Processing: Image Communication, 1990. Vol.2, No.2. pp. 127-144.

37. *Draft Revision of Recommendation H.261: Video Codec for Audiovisual Services at px64 kbit/s*, Signal Processing: Image Communication, 1990. Vol.2, No.2. pp. 221-240.

38. C.A. Gonzales, L. Allman, T. McCarthy, T. Wendt and A.N. Akansu, *DCT Coding for Motion Video Storage Using Adaptive Arithmetic Coding.*, Signal Processing: Image Communication, 1990. Vol.2, No.2. pp. 145-154.

39. F. Pereira, L. Contin, M. Quaglia and P. Delicati, *A CCITT Compatible Coding Algorithm for Digital Recording of Moving Images.*, Signal Processing: Image Communication, 1990. Vol.2, No.2. pp. 155-170.

40. Carsten Herpel, Dietmar Hepper and Dietrich Westerkamp, *Adaptation and Improvement of CCITT Reference Model 8 Video Coding for Digital Storage Media Applications.*, Signal Processing: Image Communication, 1990. Vol.2, No.2. pp. 171-186.

41. M. Haghiri and P. Denoyelle, *A Low Bit Rate Coding Algorithm for Full Motion Video Signal.*, Signal Processing: Image Communication, 1990. Vol.2, No.2. pp. 187-200.

42. Kazuto Kamikura and Hiroshi Watanabe, *Video Coding for Digital Storage Media Using Hierarchical Intraframe Scheme.*,SPIE Vol.1360 Visual Communications and Image Processing, 1990. pp. 1540-1548.

43. A. Puri and R. Aravind, *On Comparing Motion-Interpolation Structures for Video Coding.*, SPIE Vol.1360 Visual Communications and Image Processing, 1990. pp. 1560-1567.

44. E. Viscito and C.A. Gonzales, *Encoding of Motion Video Sequences for the MPEG Environment Using Arithmetic Coding,* SPIE Vol.1360 Visual Communications and Image Processing, 1990. pp. 1572-1575.

45. Atul Puri, *Efficient Motion-Compensated Coding for Low Bit-Rate Video Applications.*, Ph.D. dissertation. The City University of New York, 1988.

46. Sakae Okubo, *Video Codec Standardization In CCITT Study Group XV.*, Signal Processing: Image Communication, June 1989. Vol.1, No.1. pp. 45-54.

47. S.O. Leung, K.L. Chan and P.W. Fung, *Compression Techniques for Still Image and Motion Video,* IEEE TENCON 1993 / Beijing. pp. 365-366.

48. Gregory K. Wallace, *The JPEG Sill Picture Compression Standard,* Communication of the ACM, 1991. Vol.34, No.4.

49. T. Koga, K. Iinuma, A. Hirano, Y. Iijima and T. Ishiguro, *Motion-Compensated Interframe Coding for Video Conferencing*, Proceeding of National Telecommunication Conference, Dec. 1981. pp. G5.3.1-5.3.5.

50. P. Ang, P. Ruetz and D. Auld, *Video Compression Makes Big Gains.*, IEEE Spectrum. October, 1991.

51. A. Puri, H. M. Hang, D. L. Schilling. *An efficient block-matching algorithm for motion compensated coding.* Proc. IEEE ICASSP. 1987. pp. 25.4.1-25.4.4.

52. CCITT SG XV, *Specialists Group on Coding for Visual Telephony.*, Description of Reference Model 8. Document 525, June 1989.

53. S. Kappagantula and K.R. Rao, *Motion Compensated Predictive Coding.*, Proc. Int. Tech. Symp. SPIE, Aug. 1983.

54. J.R. Jain and A.K. Jain, *Displacement Measurement* and Its Application in Interframe Image Coding. IEEE Transaction on Communications. COM-29. Dec. 1981. pp. 1799-1808.

55. Chuen-Cherng Cheng, Ta-Kang Ku, Jhing-Fa Wang and Jau-Yien Lee, *Adaptive Implementation of the MPEG Encoder*, Proc. 1993 IEEE Reg. 10 Conf. Comput. Commun. Control Power Eng. 1993.

56. Stan Baron and Robin W. Wilson, *MPEG Overview*, SMPTE Journal, June 1994. Vol.103, No.6. pp. 391-394.

57. Xiaobing Lee, Ya-Qin Zhang and A. Leon-Garcia, *Image and Video Reconstruction Using Fuzzy Logic.* IEEE Global Telecommunications Conference, Vol.2, 1993. pp. 975-979.

58. Frank Laczko, *Motion Pictures Experts Group Digital Compression Standard and Its Role in Satellite Systems.*, IEEE National Telesystems Conference, 1993. pp. 117-119.

59. Ketan Patel, Brian C. Smith and Lawrence A. Rowe, *Performance of a Software MPEG Video decoder.*, Proc. 1 ACM Int. Conf. Multimedia 1993. pp. 75-82.

60. Huifang Sun, *Hierarchical Decoder for MPEG Compressed Video Data*, IEEE Transactions on Consumer Electronics, Aug. 1993. Vol.39, No. 3. pp. 559-564.

61. Chi-Fa Chen and Khee K. Pang, *Optimal Transform of Motion-Compensated Frame Difference Images in a Hybrid Coder,* IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, June 1993. Vol.40, No.6. pp. 393-397.

62. J. Ziv and A. Lempel, *Compression of Individual Sequences Via Variable-Rate Coding.* IEEE Transaction on Information Theory. Sept. 1978. pp. 530-536.