FACE IMAGE METAMORPHOSIS OR MORPHING

Rohan Nasina

Sindhu Kommareddy

MORPHING:

Morphing can be defined as an animated transformation of one image into another image. Morphing involves image processing techniques like warping and cross dissolving. Morphing is a special effect in motion pictures and animations that changes (or morphs) one image or shape into another through a seamless transition. Most often it is used to depict one person turning into another through technological means or as part of a fantasy or surreal sequence. Traditionally such a depiction would be achieved through cross-fading techniques on film.

BREIF HISTORY:

Computer-animated morphing was used in the 1974 Canadian animation Hunger. Though the 1986 movie The Golden Child implemented very crude morphing effects from animal to human and back, the first movie to employ detailed morphing was Willow, in 1988. A similar process was used a year later in Indiana Jones and the Last Crusade to create Walter Donovan's gruesome demise. Both effects were created by Industrial Light & Magic using grid warping techniques developed by Tom Brigham and Doug Smythe (AMPAS).

In 1985, Godley & Creme created a primitive "morph" effect using analogue cross-fades in the video for "Cry". The cover for Queen's 1989 album The Miracle featured the technique to morph the four band members' faces into one gestalt image. In 1991, morphing appeared notably in the Michael Jackson music video "Black or White" and in the movies Terminator 2: Judgment Day and Star Trek VI: The Undiscovered Country. The first application for personal computers to offer morphing was Gryphon Software Morph on the Macintosh. Other early morphing systems included ImageMaster, MorphPlus and CineMorph, all of which premiered for the Commodore Amiga in 1992. Other programs became widely available within a year, and for a time the effect became common to the point of cliché. For high-end use, Elastic Reality (based on MorphPlus) saw its first feature film use in The Line of Fire (1993) and was used in Quantum Leap (work performed by the Post Group). At VisionArt Ted Fay used Elastic Reality to morph Odo for Star Trek: Deep Space Nine. Elastic Reality was later purchased by Avid, having already become the de facto system of choice, used in many hundreds of films. The technology behind Elastic Reality earned two Academy Awards in 1996 for Scientific and

Technical Achievement going to Garth Dickie and Perry Kivolowitz. The effect is technically called a "spatially warped cross-dissolve". The first social network designed for user-generated morph examples to be posted online was Galleries by Morpheus (morphing software).

In Taiwan, Aderans, a hair loss solutions provider, did a TV commercial featuring a morphing sequence in which people with lush, thick hair morph into one another, reminiscent of the end sequence of the "Black or White" video.

MORPHING PROCESS:

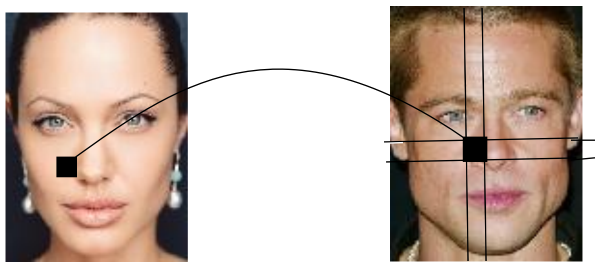
Morphing involves image processing techniques like warping and cross dissolving. Cross dissolving means that one image fades to another image using linear interpolation. This technique is visually poor because the features of both images are not aligned, and that will result in double exposure in misaligned regions.

In order to overcome this problem, warping is used to align the two images before cross dissolving. Warping determines the way pixels from one image are correlated with corresponding pixels from the other image. It is needed to map the important pixels, else warping doesn't work.

Warping is basically of two types

- **Forward mapping**

Consider pixel at (u,v) as a unit square in source image. Map square to a quadrilateral in destination image. Assign (u,v)'s gray level to pixels that the quadrilateral overlaps



Integrate source pixels' contributions to each output pixel. Destination pixel's gray level is weighted sum of intersecting source pixels' gray levels, where weight proportional to coverage of destination pixels, Avoids holes, but not folds, and requires intersection test.

- **Reverse mapping**

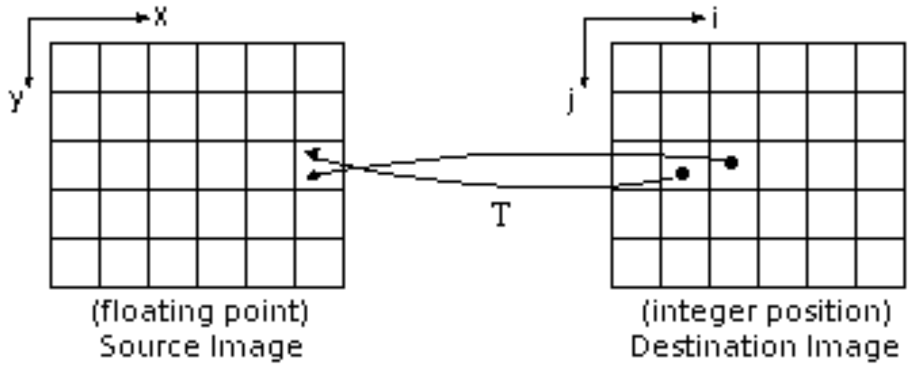
```
For x = xmin to xmax
for y = ymin to ymax
u = U(x, y)
v = V(x, y)
B[x, y] = A[u, v]
```

But (u, v) may not be at a pixel in A

(u, v) may be out of A's domain. If U and/or V are discontinuous, A may not be connected!

Digital transformations in general don't commute

Moving other pixels is obtained by extrapolating the information specified for the control pixels. Knowing cross dissolving is very simple, the real problem of morphing becomes the warping technique. Morphing is actually a cross dissolving applied to warped images.



Warping techniques vary in the way the mapping of control pixels is specified and the interpolating technique that is used for other pixels. Morphing applications are very easy to find. Film makers from Hollywood use advanced morphing techniques to generate special effects. Even Disney animations are made using morphing, for speeding production. Because there are a small number of applications to generate face morphing, there is an increased interest in this domain.

Cross – Dissolving:

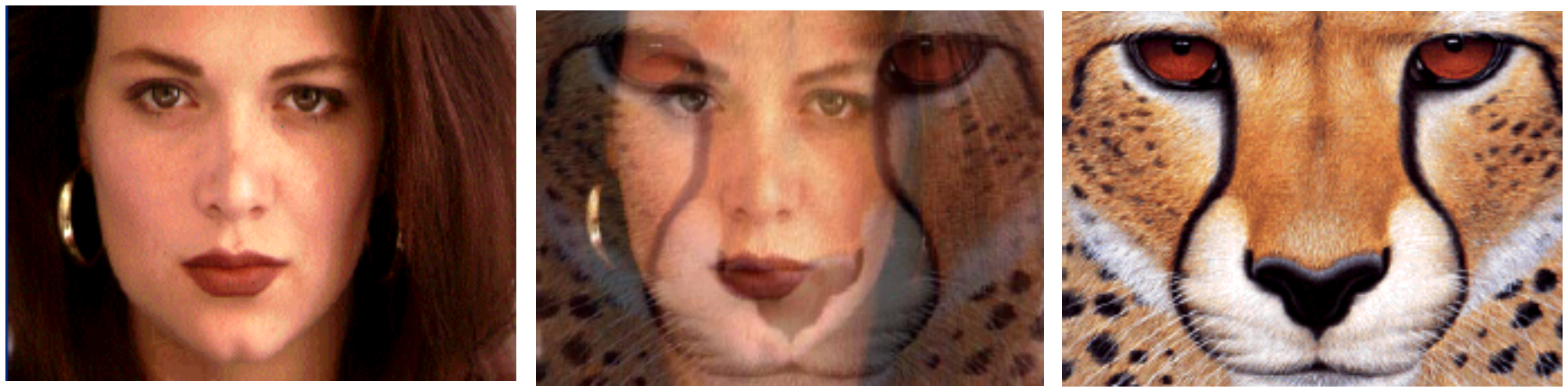
After coordinate transformations for each of the two facial images are performed, the feature points of these images are matched. i.e., the left eye in one image will be at the same position as the left eye in the other image. To complete face morphing, we need to do cross-dissolving as the coordinate transforms are taking place. Cross-dissolving is described by the following equation,

$C(x,y) = \alpha A(x,y) + (1-\alpha) B(x,y)$

$0 \leq \alpha \leq 1$

where A, B are the pair of images, and C is the morphing result. This operation is performed pixel by pixel, and each of the color components RGB is dealt with individually.

CROSS – FADING/ CROSS-DISSOLVING



FACE METAMORPHOSIS:

Image metamorphosis, or image morphing, denotes interpolation between images of different objects from (user-defined) correspondences alone, i.e., without any additional information such as geometry or camera calibration. Well-known is the line-based morphing method proposed by Beier and Neely [1992] from its use in Michael Jackson's music video "Black & White". Lerios et al. [1995] extended the approach to 3D voxels and addressed ghosting artifacts by correcting the warp field. Other warping techniques have been discussed by Wolberg, including the popular thin-plate spline interpolation which is based on point correspondences. A computationally more complex method based on line features was recently proposed by Schaefer et al.

Image morphing algorithms are based on a dense 2D vector field of correspondences along which both images are warped and linearly blended to obtain in-between images. This simplistic motion model, however, does not allow one to properly handle, e.g., openings and closings. Recent advances in perception research give clues on how non-linear blending can be employed to conceal otherwise annoyingly visible inconsistencies of in between images. [Giese and Poggio 2000; Giese and Poggio 2003].

Fig: The process of morphing a man into a woman

The optical flow plays a major role in perceptional motion analysis. Since the pioneering work on local and global optical flow reconstruction by Lucas and Kanade [1981] and Horn and Schunck [1981]. Image blending as a processing step in image compositing is traditionally realized as the linear combination of. Only recently, Grundland et al. [2006] proposed different non-linear blending functions to preserve image contrast, color, or salient regions. When applied to image morphing, however, preserving any of these characteristics can prove detrimental since occlusion artifacts could actually become amplified if, e.g., a more salient foreground vanishes into a less salient background. Nevertheless, by adapting the blending function, non-linear blending can also conceal (dis)occlusions during image morphing.

DIFFERENT TYPES OF TECHNIQUES IN IMAGE METAMORPHOSIS:

Mesh warping

Feature based warping

Thin spline based image morphing

Multilevel B-spline morphing

Improved multilevel B-spline morphing

MESH WARPING:

This is an algorithm formed in two steps that accepts a source image and two 2D arrays of coordinates S and D. The S coordinates represent the control pixels in the source image, and the D coordinates are the locations where the S coordinates will match. The final image is the initial image warped by means of mesh S and mesh D. The 2D arrays in which the control points are stored, impose a rectangular topology to the mesh. The only constraint is that the meshes defined by both arrays be topologically. Therefore the D coordinates are coordinates that may move as far from S as necessary, as long as they do not intersect with themselves.

The first step means re-sampling each row independently. An intermediate array of points I, whose x coordinates are same as those in D and whose y coordinates are the same as those in S, is created.

Vertical splines are generated to fit each column of data in S and in I. The data for each region in a row is interpolated to create intermediate image I. The second step consists in re-sampling each column independently. Horizontal splines are then generated to fit each row of data in arrays I and D. The data for each region in a column is interpolated from intermediate image I to create destination image D. The collection of vertical splines fitted through S and I in the first step and with the horizontal splines fitted through I and D in the second step,

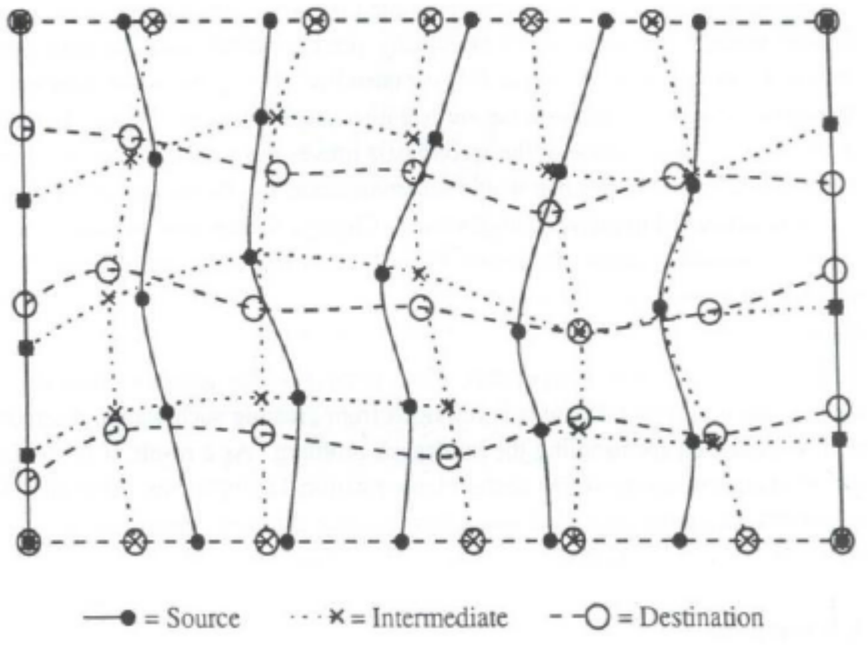


Fig. 1 Vertical splines.

FEATURE BASED IMAGE WARPING:

This is a method that offers a high level of control over the process. The corresponding feature lines in the two images that are being morphed, are interactively selected. The algorithm uses lines to relate features in the source image to features in the final image. This algorithm is based upon fields of influence surrounding the feature lines selected. It uses reverse mapping for warping the image.

A pair of lines (one defined relative to the source image, the other defined relative to the destination image) defines a mapping from one image to the other as in figure 2

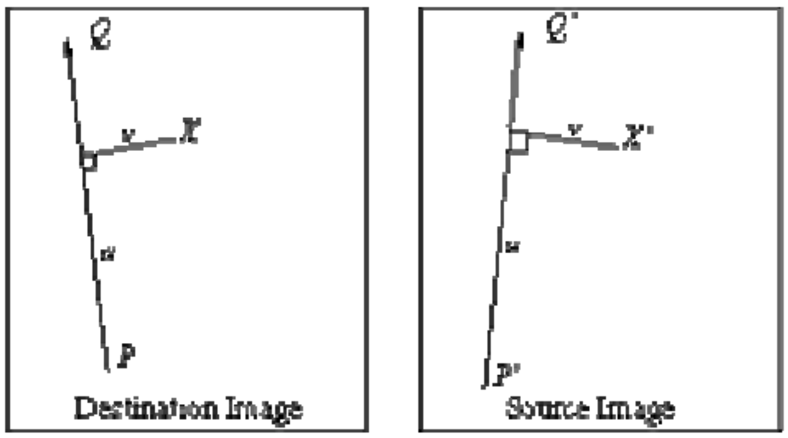


Fig. 2 Mapping from one image to the other.

The algorithm transforms each pixel coordinate by a rotation, translation, and/or a scale, in this way transforming the whole image. In a normal morphing scenario, however there are multiple features in the images to be morphed and consequently multiple feature line pairs are specified. The displacement of a point in the source image is then, actually a weighted sum of the mappings due to each line pair, with the weights attributed to distance and line length. The weight assigned to each line should be strongest when the pixel is exactly on the line, and weaker the further the pixel is from it.

Thin Plate Spline Based Image Warping:

Thin-plate Spline is a conventional tool for surface interpolation over scattered data. It is an interpolation method that finds a "minimally bended" smooth surface that passes through all given points. The name "Thin Plate" comes from the fact that a TPS more or less simulates how a thin metal plate would behave if it was forced through the same control points.

Let us denote the target function values vi at locations (xi, yi) in the plane, with I = 1,2,…….p , where p is the number of feature points. In particular, we will set vi equal to the coordinates (xi', yi') in turn to obtain one continuous transformation for each coordinate. An assumption is made that the locations (xi, yi) are all different and are not collinear.

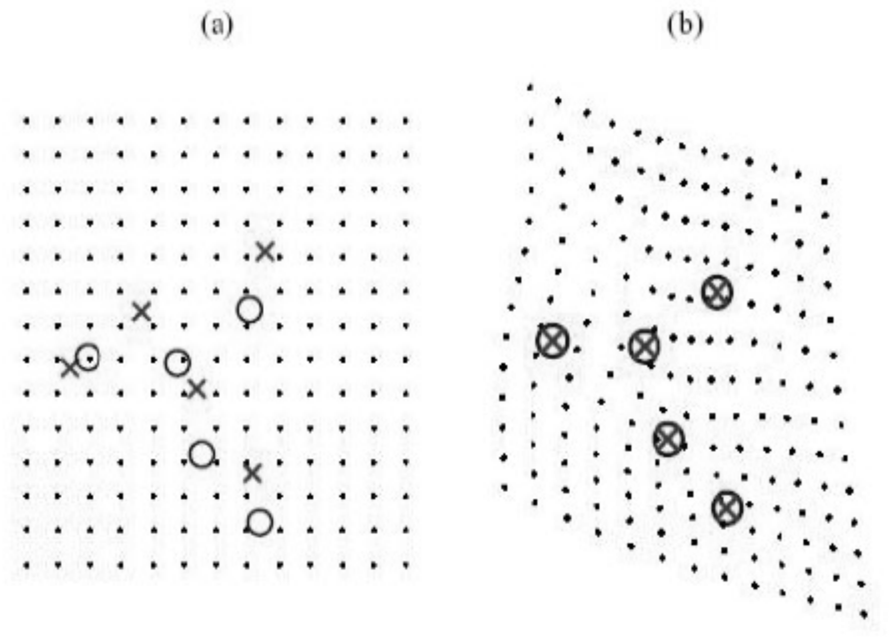


Fig. 3 Example of coordinate transformation using TPS.

The figure 3 is a simple example of coordinate transformation using TPS. It starts form two sets of points for which it is assumed that the correspondences are known (a). The TPS warping allows an alignment of the points and the bending of the grid shows the deformation needed to bring the two sets on top of each other (b). In the case of TPS applied to coordinate transformation we actually use two splines, one for the displacement in the x direction and one for the displacement in the y direction. The two resulting transformations are combined into a single mapping.

B-SPLINE APPROXIMATION:

The free-form deformation based on B-Spline approximation is a powerful and useful morphing algorithm and is proven to have the one-to-one property which can prevent the warped image from folding back upon itself. The control lattice of basic B-spline approximation is shown in Figure. We perform transformation for a local domain which contains 9 blocks and 16 control points and then we shift the local domain by one block and repeat the transformation.

Let Ω = {(x, y) | -1≤ x ≤m +1,-1≤ y ≤ n + 1} be a rectangular domain in xy-plane. Use a set of scattered points P = {(x, y, z)} in 3D space. To approximate scattered data P, we formulate approximation function f as a uniform bicubic B-spline function, which is defined by a control lattice ϕ overlaid on domain Ω. Let ϕij be the value of the ij-th control point on ϕ. The approximation function f is defined in terms of these control points by

$$Z= f(x,y)=\sum_{k=0}^{3} \mathrm{Bk}(s)\mathrm{Bl}(t)\square(i+k)(j+l)$$

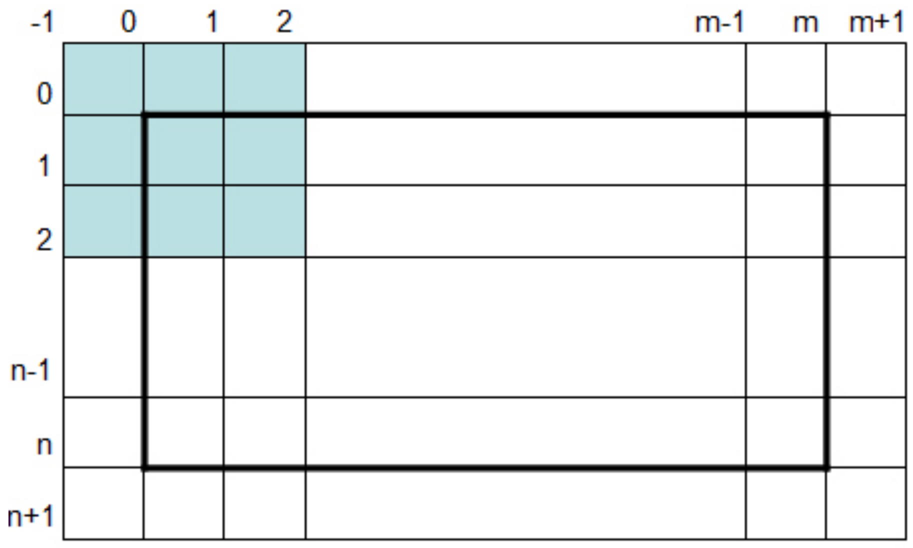
where I =|x|- 1, j = |y|-1 , s = |x|- x, t = |y| - y. Bk and Bl are uniform cubic B-spline basis functions defined as

B0(s) = $(1-s)^3/6$

B1(s) = $(3s^3 - 6s^2 + 4) / 6$.

B2(s) = $(-3s^3+3s^2+3s+1)/ 6$

where 0 ≤s ≤1.



In image morphing, suppose a set of corresponding landmarks {(x1, y1),(x2, y2), …., (xn, yn) in image 1 and {(x1 ', y1 '),(x2', y2'), ….. , (xn', yn')} in image 2 are given. The values of control points ϕklx and ϕkly of two spatial transformations (B-spline functions) can be estimated or assigned by minimizing the following cost functions.

$$E(\phi_{kl}^{x}) = \sum_{i=1}^{n} \left(\sum_{k=0}^{3} \sum_{l=0}^{3} B_k(s_i) B_l(t_i) \phi^{x}_{(i+k)(i+j)} - x_i' \right)^2$$

$$E(\phi_{kl}^{y}) = \sum^{n} \left(\sum^{3} \sum^{3} B_k(s_i) B_l(t_i) \phi^{y}_{(i+k)(i+j)} - y_i' \right)^2$$

MULTILEVEL B-SPLINE MORPHING:

For image morphing, we want to establish a spatial transformation between the points (x, y) in image 1 (input image) and their corresponding points (x’, y’) in image 2 (output image). The spatial transformation can be defined in terms of two mapping functions. The mapping functions fx and fy can be approximated by two B-spline functions. That is, image warping is to estimate the coefficients of fx and fy that best approximate the surfaces pass through points (x, y, x’) and (x, y, y’), respectively.

The image metamorphosis method with B-spline approximation usually generates a transformation error. We can reduce this error by lowering the size of the control lattices. However, as a result, the image of the transformation result was not smooth. A tradeoff exists between the shape smoothness and accuracy of the approximation function generated by B-spline approximation algorithm. Multilevel B-spline approximation is proposed to circumvent this tradeoff.

In this method, we use multiple control lattice, ϕ1, ϕ2….ϕh . We assume that the spacing between control points for ϕ1 is given and that the spacing is halved from one lattice to next. Therefore, if ϕk is an (m + 3)*(n + 3) lattice, the next finer lattice ϕ k+1 will have (2m + 3) * (2n + 3) control points. The position of the ij-th control point in ϕk coincides with that of the (2i, 2j)-th control point in ϕk+1

At first we perform the basic B-spline approximation with a coarse lattice ϕ1. The resulting function f1 serves as a smooth initial approximation that possibly leaves large discrepancies at the data points in P. The next finer control lattice ϕ2 is then used to obtain function 2 f that approximates the difference P1 = ({x, y, Δ1z)} . Then, the sum f1 + f2 yields a smaller deviation for each point (x, y, z) in P. The transformation results of multilevel B-spline approximation are shown in figure landmarks are extracted manually as shown with red points. The blue points show the target points. The 1st level B-spline is the basic B-spline. We denote B-spline approximation as BA, and multilevel B-spline approximation as MBA.

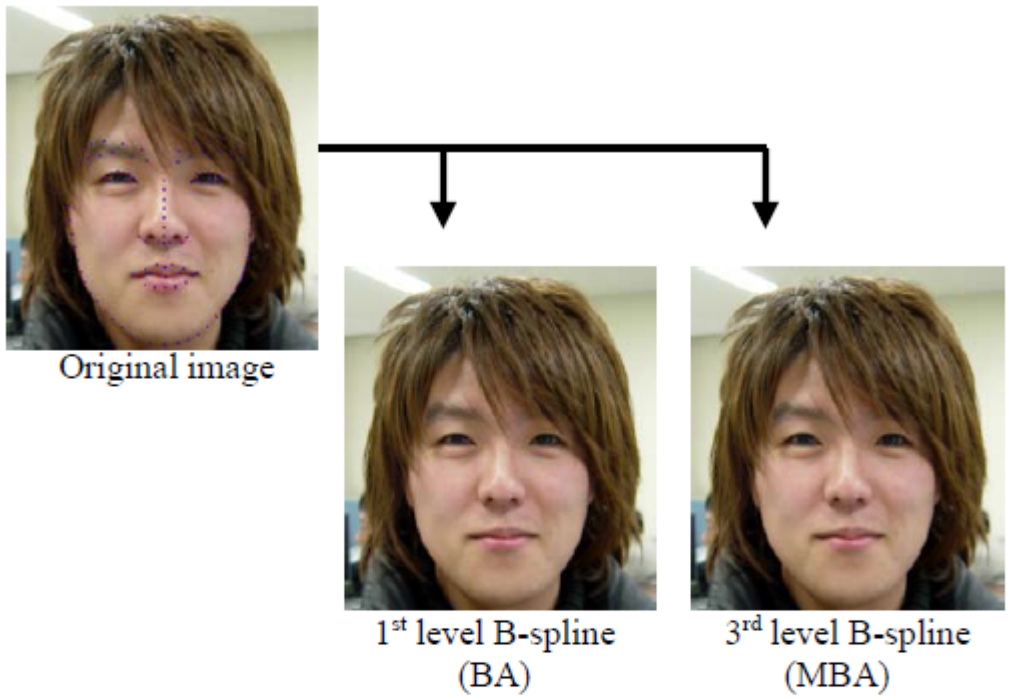


Fig: image morphing using multilevel B-spline

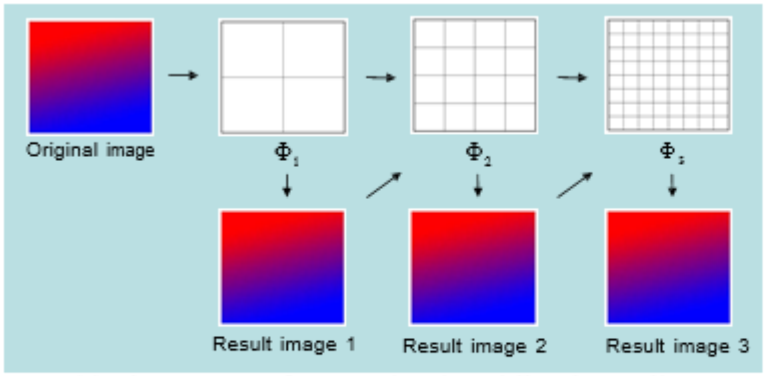
It can be seen in Fig.2 that BA has rough transformation and there are some errors in BA based morphed image, especially in the area where there are a lot of landmarks. And MBA can reduce these errors to 0. Though multilevel B-spline is very useful and powerful method, the implementation of multilevel B-spline approximation is time-consuming. So we proposed an improved multilevel B-spline approximation method in order to reduce the large computation cost.

IMPROVED MULTILEVEL B-SPLINE APPROXIMATION:

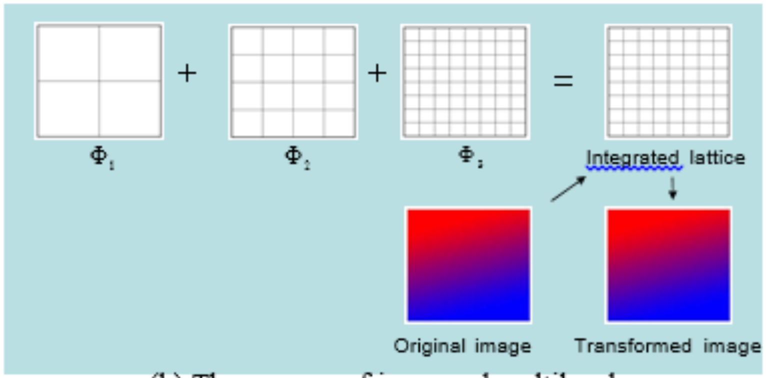
There are two types of fast implementation methods.

Lattice Integration Method:

The first proposed method is a lattice integration method. In conventional multilevel B-spline, we first halve the spacing between the lattice points and then transform the image and calculate its error. The process is repeated until an error becomes small enough. The process of conventional multilevel B-spline is shown in Figure. In our proposed lattice integration method, we integrate multiple control lattices and transform it by one calculation. The process of our proposed method is shown in the following figure. An (m + 3)*(n + 3) control lattice ϕ is refined to a (2m + 3) * (2n + 3) lattice $\phi 2$ whose control point spacing is half as large as that of ϕ. Let ϕij and $\phi ij'$ be the ij-th control points in ϕ and ϕ', respectively.



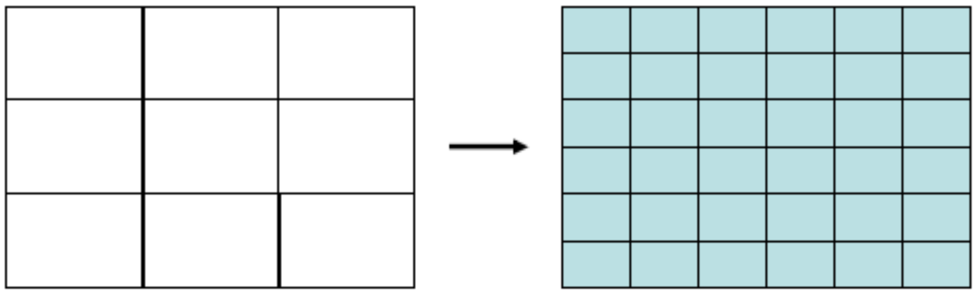
(a) The process of conventional multilevel B-spline approximation



(b) The process of improved multilevel B-spline method

Adaptative Lattice Method:

The second proposal method is an adaptive lattice method. In conventional multilevel B-spline, if there is an error, the lattice control points of the whole image will be increased to 4 times by halving the spacing between the lattices and transform the whole image again as shown in Fig.(a). In our proposed adaptive lattice method, the increase of lattice points is performed only in the domain where an error exists, which is shown in Fig (b). The transformation is also performed in the local domain.


(a) Conventional multilevel B-spline

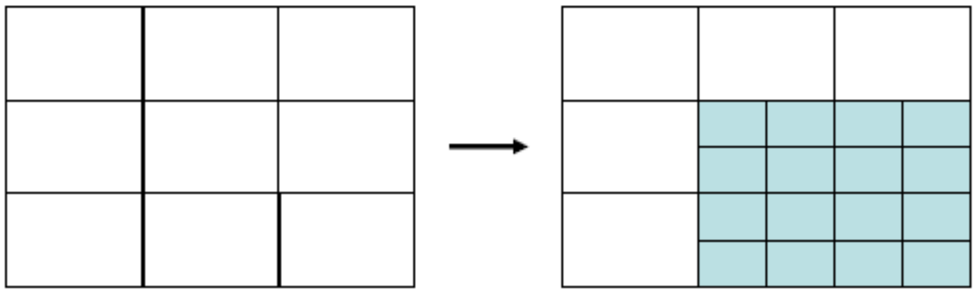

(b) Our proposed adaptive lattice method

Fig: Increase of lattice points in multilevel B-spline

One example of local domain for transformation is shown in the following figure. As increasing the level, the local domain needed to be transform is shrank.

Original image

Transform domain
(level-1)

Transform domain
(level-2)

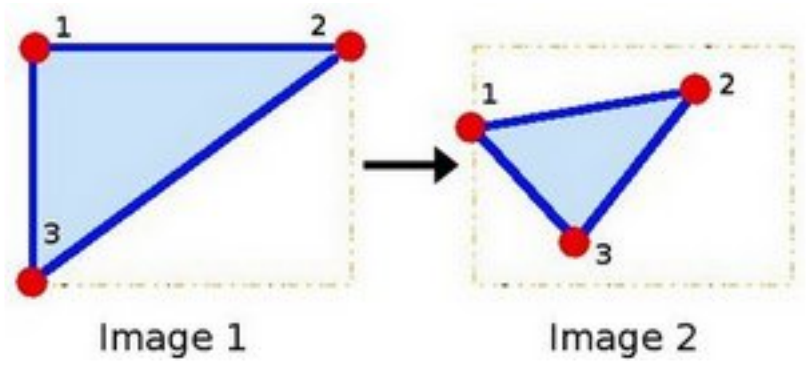
Transform domain
(level-3)

COORDINATE TRANSFORMATIONS IN FACE MORPHING:

There are many coordinate transformations for the mapping between two triangles or between two quadrangles. It is usually used affine and bilinear transformations for the triangles and quadrangles, respectively. Besides, bilinear interpolation is performed in pixel sense.

Affine Transformation:

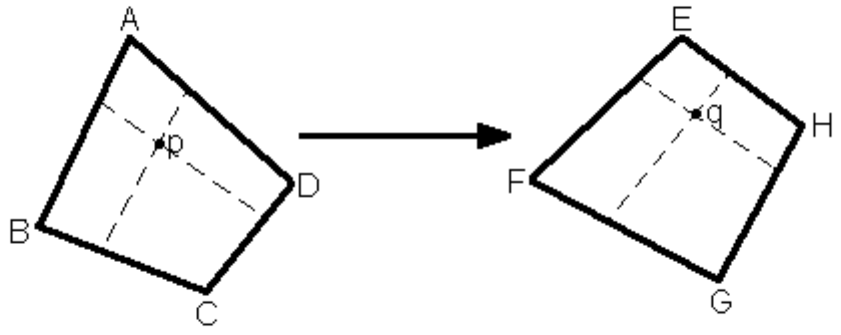
Suppose there are two triangles ABC and DEF. An affine transformation is a linear mapping from one triangle to another. For every pixel p within triangle ABC, assume the position of p is a linear combination of A, B, and C vectors.



Here λ1 and λ2, are unknown and there are two equations for each of the two dimensions. The affine transformation is a one-to-one mapping between two triangles.

Bilinear Transformation:

Suppose there are two quadrangles ABCD and EFGH.



The bilinear transformation is a mapping from one quadrangle to another. For every pixel p within quadrangle ABCD, it is assumed that the position of p is a linear combination of vectors A, B, C, and D. Bilinear transformation is given by the following equations:

$p = (1-u)(1-v)A + u(1-v)B + uvC + (1-u)vD$

$0 \le u , v \le 1$

$Q = (1-u)(1-v)E + u(1-v)F + uvG + (1-u)vH$

There are two unknown components: u and v. Because this is a 2D problem, we have 2 equations. So, u and v can be solved, and they are used to obtain q. Again, the bilinear transformation is a one-to-one mapping for two quadrangles.

MORPHING IN PARTICLE SYSTEMS:

PARTICLE SYSTEMS:

A particle system is a technique in game physics and computer graphics that uses a large number of very small sprites or other graphic objects to simulate certain kinds of "fuzzy" phenomena, which are otherwise very hard to reproduce with conventional rendering techniques – usually highly chaotic systems, natural phenomena, or processes caused by chemical reactions. Examples of such phenomena which are commonly replicated using particle systems include fire, explosions, smoke, moving water (such as a waterfall), sparks, falling leaves, clouds, fog, snow, dust, meteor tails, stars and galaxies, or abstract visual effects like glowing trails, magic spells, etc. - these use particles that fade out quickly and are then re-emitted from the effect's source. Another technique can be used for things that contain many strands - such as fur, hair, and grass - involving rendering an entire particle's lifetime at once, which can then be drawn and manipulated as a single strand of the material in question. Particle systems may be two-dimensional or three-dimensional.

Particle systems have been used to model natural phenomena such as waterfalls and fire. In these early particle systems the particles are acted on by force fields and constraints; however there is no interaction between the particles, so particles are allowed to intersect with each other without experiencing any forces from other particles in the local area.\

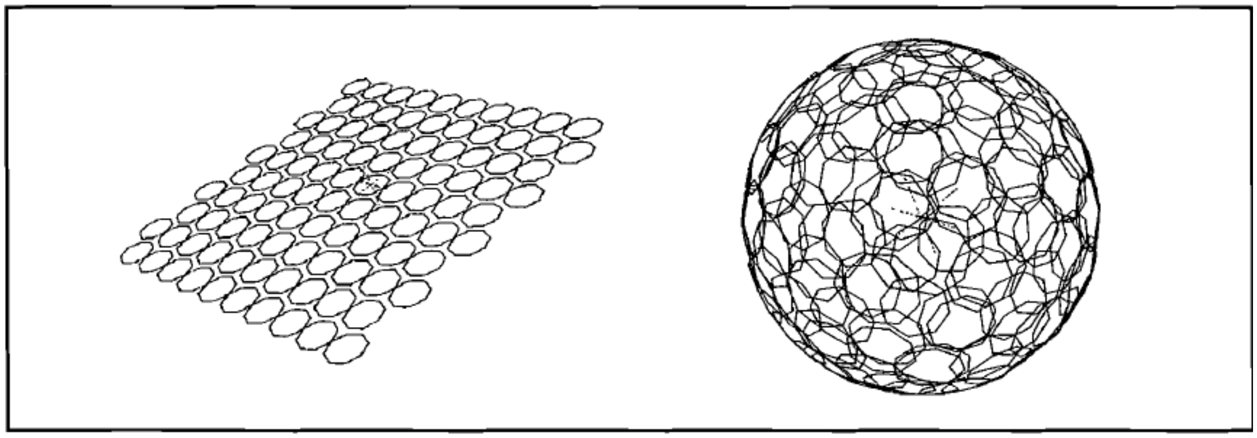
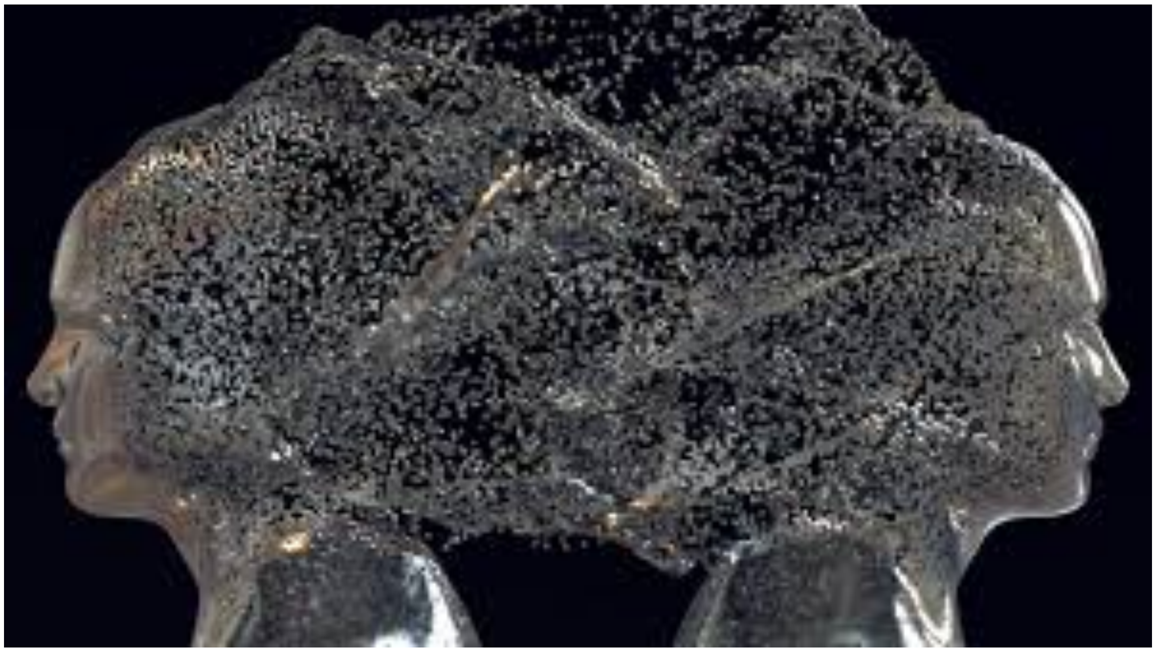
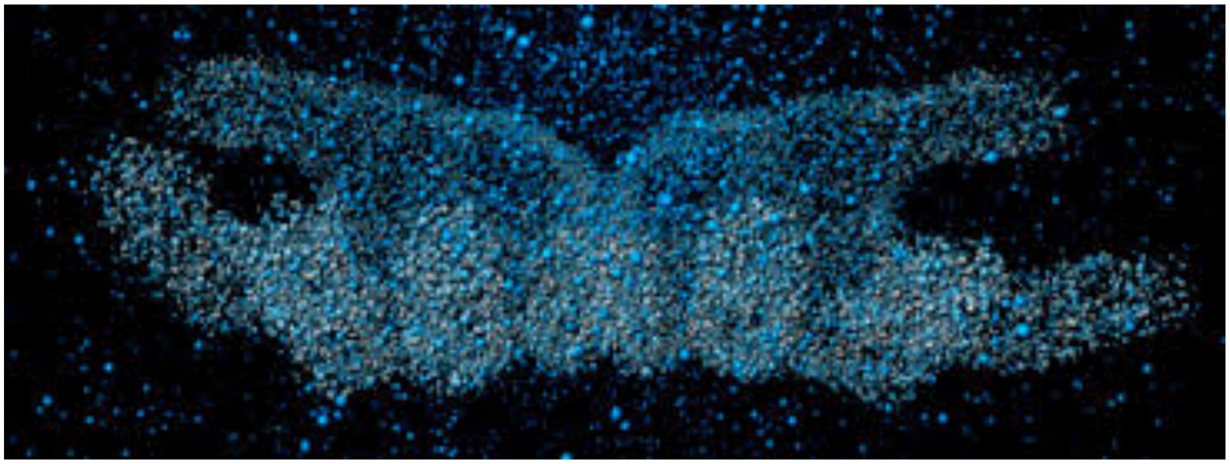


FIG: AN EXAMPLE REPRESENTED BY ORIENTATION OF PARTICLE SYSTEMS

MORPHING:

We divide the morphing process into two separate steps which we call the macro and the micro morphs. The macro morph consists of either a standard or generalized rigid body transformation involving eigenvectors which translates and rotates the initial model in such a way so that it achieves a best fit with the destination model. Once this best fit has been achieved, we perform a micro morph which actually transforms the initial model into an exact copy of the destination one. The micro morph assumes the two models are to be super-imposed in the best fit way, as indicated by the macro morph. The final visual transformation is obtained by combining the macro morph with the micro morph. That is, in the i^{th} frame of the visual morph, the user sees the i^{th} step model in the micro morph in i^{th} the position of the macro transformation. We believe this split up to be the most intuitive approach to a more general 3D morphing problem; furthermore, it affords the user, with only minimal input, a great deal of influence over the appearance of the final morph.





Macro:

The user can decide whether it is best to specify the macro level morph by providing some user input selection of particles or by having the system automatically align two models using a technique employing a special symmetric matrix and the resulting eigenvectors.

User Input and Rigid Body Alignment:

We implemented UI functionality which allows the user to pick one particle from each particle system which should correspond when the macro morph is complete. We first compute the centroids of the objects and then we compute a macro alignment vector for each system which is basically a normalized vector which emanates from the computed centroid towards the chosen particle in that system. We then align these two centroids via incremental translation and incremental rotation to align these two macro alignment vectors over the course of the user specified number of frames. For each frame we apply a rigid body transform to translate by the total translation vector divided by the number of frames. We then update the centroid to this new location, and then translate to the origin and rotate by the total rotation divided by the total number of frames and then translate back to the new centroid position. There is one major drawback to this approach: the user does not have full control over the alignment, because she is only aligning one axis of each object.

Automatic Macro Alignment using Eigenvectors:

The second approach does not require user interaction. Instead, this more general solution involves computing the eigenvectors of a symmetric matrix which has the following entries.

$$\begin{bmatrix} \sum_p x\cdot x & \sum_p x\cdot y & \sum_p x\cdot z \\ \sum_p x\cdot y & \sum_p y\cdot y & \sum_p y\cdot z \\ \sum_p x\cdot z & \sum_p x\cdot z & \sum_p z\cdot z \end{bmatrix} \text{ where } \sum_p \mathit{component}\cdot\mathit{component} \text{ is a summing over all particles.}$$

The computed eigenvectors of any symmetric matrix will be distinct and are orthogonal. We can utilize these 3 distinct eigenvectors to align two oriented particle systems. This is accomplished by aligning the largest eigenvector in each system, and then the next largest eigenvector. After we auto align these two largest eigenvectors, we can then align the smallest eigenvectors that are currently not aligned. After making eigenvectors correspond, we need to normalize them and to compute the transformation to align the eigenvectors. We do this by computing a dot product to compute the angle to rotate the largest two eigenvectors. We then compute the same ordered cross product in each system to compute the third axis for both systems. As the initial handedness of the two systems may be different we may need to perform an inversion of a direction of an eigenvector and possibly rotate 180 degrees to ensure that the two systems will align properly. One use of this more general approach is noticed when we have two surfaces which are initially identical. If we first arbitrarily translate and rotate one surface about some axis while leaving the other surface alone, then the computed eigen based solution that aligns the two systems in an affine manner is exactly, within the constraints of numerical accuracy, the same rotation matrix that initially rotated the surface in space.

Micro:

The micro morph itself consists of a two-step process. The first step is to establish a correspondence between the initial and destination particle systems. The second step involves interpolating between the two models, or intelligently moving the particles from the source system to the location of their corresponding particles in the destination system.

The User-Specified Initial Correspondence:

The user specifies the initial particle correspondence between the initial and destination particle systems. The initial correspondence consists of an arbitrary number of correspondence pairs, as selected by the user. Clearly by selecting more initial correspondence pairs, the user will have more control over the appearance of the micro morph. While the upper limit on the number of correspondence pairs is the number of particles in the smaller of the two systems, we need at least one user specified correspondence pair to automatically compute a correspondence.

In order to select a single correspondence pair, the user first selects a particle in the-initial model. Then the view of the model is rotated in such a way so that the camera faces the selected particle along that particle's normal in order to allow the selection of another point in that particle's plane. This point serves to define a sweep line, which is simply any vector emanating from the particle center and passing through the user selected point which lies in the plane of that particle. The correspondence algorithm uses this sweep line as an indication of the order in which the neighbors of the selected particle will be entered into the correspondence. Once this particle and sweep line have been selected in the initial system, the user repeats the process for the destination system. This pair of choices of particle and sweep line constitutes a single correspondence pair. Initial correspondence pairs are generated until the user is satisfied that the correspondence algorithm have been given enough clues what it should do.

The function of the sweep line is to indicate the order in which the neighbors of the first particle in a correspondence pair will be matched up with the neighbors of the second particle in the correspondence pair. Keep in mind that the first particle in each correspondence pair comes from the initial system, and the second particle comes from the destination system. On a high level, the correspondence algorithm partitions the neighbors of each particle in a correspondence pair into some arbitrary number of wedges, with the first wedge lying directly to the right of the sweep line, the second lying directly to the right of the first, and so on (i.e., the wedges are swept out clock-wise initial at the user specified sweep line). The particles which lie in the first wedge of the first particle (i.e., those from the initial model) are matched up to the particles which are in the first wedge of the second particle (i.e., those from the destination model). These two sets of particles serve to create a new correspondence pair.

We should note at this point that our algorithm computes a correspondence between two particle systems based on user input. In fact, we believe it erroneous to assume that a single correct correspondence between any two particle systems actually exists. Each correspondence leads.to a different morph which may please one user and displease another. Likewise, no single correct morph exists between any two systems, as the visual appeal of any given morph is a purely subjective judgment.

Particle Generation:

Often the source and the destination particle systems which are to be morphed do not contain the same, or even a similar, number of particles. Despite this, we would still like to be able to morph between the two systems. On a high level, as the morph proceeds, we would like to create new particles in the source system as soon as the space to contain them becomes available. Furthermore, we would like to make sure that these particles become a part of a real correspondence with particles from the destination system.

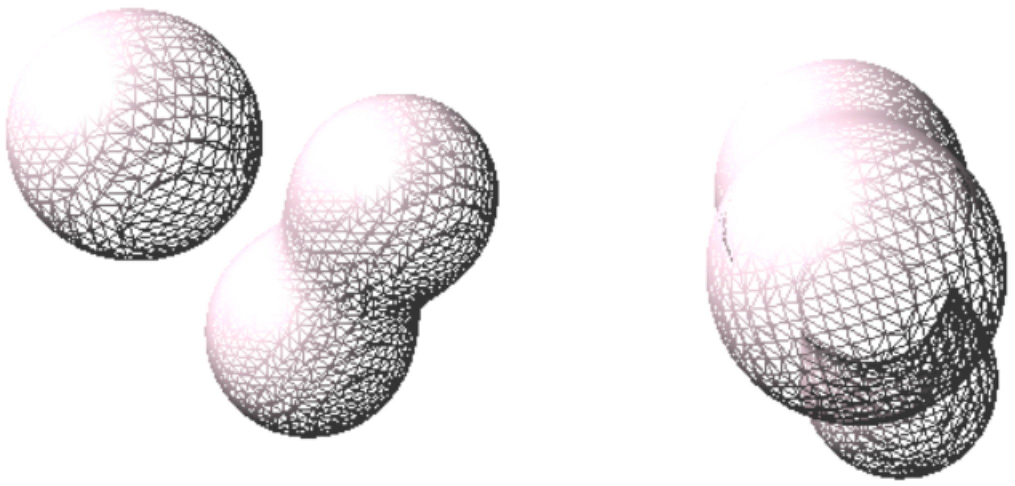
First we establish a correspondence between the initial and the destination morph systems. Since the initial system has fewer particles than the destination one, this will essentially map the former system onto parts of the latter one. We begin the interpolation part of the morph sequence, and move the initial system particles towards their corresponding particles in the destination system. For every correspondence pair in the correspondence pair list, we establish a partition of the source and destination particle sets. If it is the case that some wedge ‘I’ in the source partition is empty and the wedge ‘I’ in the destination partition is not, and the destination wedge ‘I’ contains at least one particle which is not already a member of a correspondence pair, we attempt to grow a particle in the space represented by wedge ‘I’ in the source system. If space exists for this potential new particle in the source system, we create a new correspondence pair which contains in its source particle set the newly created particle, and contains in its destination particle set the particle in the destination partition wedge ‘I’ which had not been a member of a correspondence pair.

Intuitively, we treat each source particle in a correspondence as a marker into the destination system. Thus we attempt to make the neighbors of a given source particle as similar as possible to the neighbors of the corresponding destination particle. The process of adding new particles stops when every particle in the destination system has been entered into a correspondence pair. At this point, the source and destination systems have a very similar, if not equivalent, number of particles. In addition to this, the newly created source particles are all real members of correspondence pairs, and have neighbors which correspond in a properly oriented way to the neighbors of the destination particles in their correspondence pairs. This procedure leads to very intuitive looking morphs.

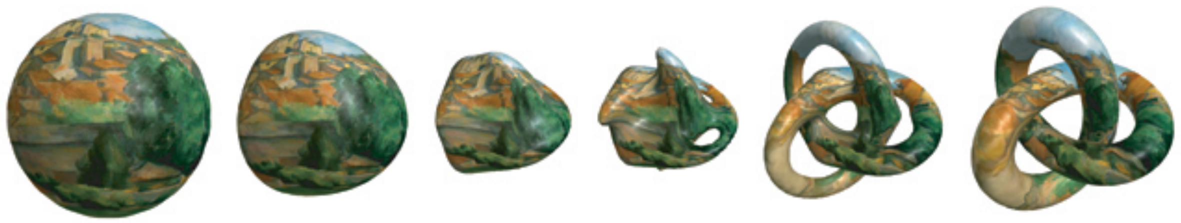
Particle Deletion:

When the source system has a greater number of particles than the destination system, we are confronted with the opposite problem of the one we addressed in the previous section. Specifically, we now need to delete particles, instead of adding new ones. In this case we expect that the destination system will have each of its particles in a correspondence pair with a particle set from the source system. Thus any particle in the source system which is a real correspondence pair member (as opposed to having been an orphan) needs to be left alone, as it will be morphed onto its corresponding particle set in the destination system. The orphaned particles, on the other hand, have no place to go in the destination system. We continuously check all orphaned and unmatched (or un-corresponded) particles to see whether they are tightly surrounded by other particles.

Some Morphing Results:



Metaballs



Morphing of sphere into a sphere-knot

Conclusion:

In this write–up we discussed about various morphing algorithms and provide the animator with sufficient information to make an informed choice suiting his particular needs. In doing so we have defined a few easily comparable attributes, such as visual quality of morph, the ease with which the animator can select control pixels and the computational complexity. We found that Mesh morphing gives the best result among the algorithms we implemented but it requires a significant amount of animator effort in selecting the control pixels. The Thin Plate Spline gives results, which are of comparable quality with very little effort required from the animator. The Feature based morphing algorithm requires the animator to select a significantly larger number of feature lines to give the same results and also a lattice integration method and an adaptive lattice method for multilevel B-spline approximation. The experimental results show that the improved multilevel B- spline method is more efficient than conventional multilevel B-spline method. We also discussed about morphing in particle systems where we proposed methods in morphing which give a pretty good result.

References:

[1] S. Lee, G. Wolberg, S.Y. Shin, “Scattered Data Interpolation with Multilevel B-spline”, IEEE Transaction on Visualization and Computer Graphics, Vol. 3, PP.228-244, 1997.

[2] G. Wolberg, “Recente Advances in Image Morphing”, Computer Graphics International’96, Korea, 1996.

[3] G. Wolberg, “Image Metamorphosis Using Snakes and Free-Form Deformation”, Department of Computer
Science, City College of New York / CUNY, 1995.

[4] G. Wolberg, “Digital Image Warping”, IEEE Computer Society Press, Los Alamitos, CA, 1990.

[5] N.Arad, N.Dyn, D.Reisfeld, Y.Yeshurun “Image warping by radial basis functions: Applications to facial
expressions”, CVGIP: Graphical Models and large Processing, 56(2): 161-172, March 1994.

[6] P.Litwinowicz, L.Williams “Animating images with drawings”, Computer Graphics (Proc. SIGGRAPH ‘94), pages 409-412, 1994.

[7] S.-Y.Lee, K.-Y.Chwa, J.Hahn, S.Y.Shin, “Image metamorphosis using deformable surfaces”, Proc. Computer Animation ’94, pages 31-39, 1994, IEEE

[8] Beier, T., and Neely, S. (1992). Feature-based image metamorphosis, Proc. SIGRAPH 92, in Computer
Graphics, pp. 35-42.

[9] Bookstein, F. L. (1989). Principal Warps: Thin-plates splines and decomposition of deformations, IEEE
Trans. Pattern Analysis and Machine Intelligence, vol 11(6), pp. 567-585.

[10] Seitz, S.M., and Dyer, C. R. (1996). View Morphing, Proc. SIGRAPH 96, in Computer Graphics, pp. 21-30.

[11] Wolberg, G. (1990). Digital Image Warping. IEEE Computer Society Press, Los Alamitos, CA.

[Beie92] Beier, Thaddeus and Shawn Neely, "Feature-Based Image Metamorphosis", Computer Graphics (SIGGRAPH 1992), Vol 26, No.2, July 1992, pp.35-43.

[Beth89] Bethel, E. Wesley and Samuel P. Uselton, "Shape Distortion in Computer-Assisted Keyframe Animation," State-of-the-art in Computer Animation: Proceedings of Computer Animation 1989, N. Magnenat-Thalmann and D. Thalmann, eds., pp. 215-224.

[LBois84] Boissonat, J.D., "Representing 2D and 3D shapes with Delaunay triangulation," in Seventh International Conference of Pattern Recognition ([CPR (84), pp. 745-748, Montreal, Canada, July 1984.

[Burd93] Burden, Richard and J. Douglas Faires, "Numerical Analysis," Plus-Kent Publishing Company, Boston, 1993.

[Chen85] Chen, Eric and Richard Parent, "Shape Averaging and Its Applications to Industrial Design", IEEE Computer Graphics and Applications, Vol. 9, No. 11, January 1989, pp. 187-196.

[Cocq90] Cocquillart, Sabine, "Extended Free-Form Deformations: A Sculpting Tool for 3D Geometric Modeling," Computer Graphics (SIGGRAPH (90), August 1990, pp.187-196.

[Corm90] Cormen, Thomas H. , Charles E. Leiserson, and Ronald L. Rivest, "Introduction to Algorithms," MIT Press, 1990.

[Ferg92] Ferguson, Helaman, Alyn Rockwood, and Jordan Cox, "Topological Design of Sculptured Surfaces, "SIGGRAPH 1992, pp. 149-156.