3D Object Metamorphosis Through Energy Minimisation.

Ioannis Korfiatis and Yakup Paker

Department of Computer Science
Queen Mary and Westfield College
E1 4NS
London
UK

{korfiati, paker}@dcs.qmw.ac.uk

Abstract

This paper presents a technique for smoothly blending some special categories of three dimensional polygonal objects. Polygon blending is usually considered a two-part process: generating vertex correspondences and interpolating between corresponding vertices to create the intermediate polygons. This paper considers the problem of automatic vertex correspondence determination. The proposed algorithm is based on the work of Sederberg and Greenwood [SED92]. The resulting solution tends to associate regions of the two objects which look alike.

keywords: computer animation, object metamorphosis, physically based algorithms, shortest path algorithm, surface reconstruction from planar contours.

1. Introduction

A common approach in shape metamorphosis involves a pair of objects represented as a collection of polygons. The vertices of the first polygonal object are then displaced over time to coincide in position with the corresponding vertices of the second object. There are two steps in this approach: first establishing a desirable vertex correspondence and then interpolating the coordinates of the corresponding vertices to get the intermediate objects.

Correspondence determination is a somewhat ambiguous process, related more to human judgments rather than well-understood, universaly applicable definitions. We can distinguish two levels in solving this problem, the syntactic level where shapes are treated as geometric entities and the metamorphosis is concerned only with geometric properties of elements such as sizes or distances and the semantic level where correspondence is established on the basis of similarity in parts of the two objects.

The methods for correspondence determination in the semantic level that have been proposed so far rely on user interaction or on domain specific knowledge base. The former approach can be extremely tedious for complex objects while the later limits the metamorphosis to objects of a certain category.

Our approach is based on the fact that features of the two objects are assumed similar if small changes are needed to transform one to the other. The measure to quantify these 'small' or 'big' changes in our case is the energy needed to perform them assuming that the objects are made of a piece of wire. This is the basis of Sederberg's algorithm [SED92]. We extend this approach to three dimensional polygonal objects. To reduce the complexity of the proposed solution we consider a simplified representation of the original models.

2. Related work

Several approaches to 3D shape transformation exist in the literature. Brute force approaches for polyhedral models, as described by Terzides [TER89] essentially require the user to specify for each vertex a corresponding vertex from the other model. Hong et.al [HON88] propose a solution for polyhedra based on matching the faces of the objects whose centroids are closest. Bethel & Uselton [BET89] describe an algorithm that adds degenerate vertices and faces to two polyhedra until a common topology is achieved. Chen & Parent [CHE89] present a transformation algorithm for piecewise linear 2D contours, then briefly address an extension for 3D lofted objects. Parent [PAR91] describes a solution for polyhedra that establishes correspondence by splitting the surface of the models into pairs of sheets of faces then recursively subdividing them until the topology of each pair is identical. Kaul & Rosignac [KAU91] transform pairs of polyhedra by computing the Minkowski sum of scaled versions of the models. By gradually scaling one model from 100% to 0% while simultaneously scaling the other form 0% to 100% a transformation is obtained. Payne & Toga [PAY92] first convert each polyhedra into a distance field volumetric representation, interpolate the values at each point of the 3D volume then find a new isosurface that represents some combination of the original models. Kent [KENT92] projects the topology of both models onto the unit sphere, then the two topologies are merged by clipping the projected faces of one model to the projected faces of the other. The merged topology is then mapped onto the surface of both original models. This generates two new models that have the same shape as the original models but that share a common topology. This allows a transformation between the two shapes to be easily computed by interpolating the coordinates of each pair of corresponding vertices. Hughes [HUG92] interpolates between the fourier transform of the original volumetric models.

Some of these approaches give very interesting results, but none determines correspondences in the semantic level. The method we are proposing tries to recognize similar features on the two terminal objects and maintain those features throughout the blend.

3. Sederberg's method overview

Sederberg's method addresses the problem of finding correspondences between the vertices of two closed polygonal contours. The algorithm is based on a physical model. Imagining that each shape is made of a piece of wire the blend is determined by computing the minimum work required to bend and stretch one wire shape into another.

Suppose that we wish to find the correspondences between two polygonal contours $P^0=[P^0_0, P^0_1,...,P^0_m]$ and $P^1=[P^1_0,P^1_1, ...,P^1_n]$ where P^i_k denote the vertices.

A correspondence between the vertices P^0_i, P^1_j is denoted as [i,j].

To keep the connectivity of the intermediate polygons during the shape transformation it is allowed [i,j] to be a correspondence if [i-1,j], [i, j-1], or [i-1, j-1] is also a correspondence - else intermediate polygons in the shape transformation would split apart. Stretching work is computed for each pair of neighbouring vertices on P^0 transformed to two vertices of P^1. Bending work is computed when three consecutive vertices of P^0 move to three consecutive vertices on P^1.

Suppose that [0,0] is a known correspondence. The basic strategy proceeds by considering polygon fragments consisting of vertices 0,..,i of P^0 and vertices 0,...,j of P^1. Let W(i,j) be the the minimum work required to transform the first fragment to the second and suppose that we know the minimum work values W(i-1,j), W(i,j-1), and W(i-1,j-1), then W(i,j) must equal one of those predecessors plus the incremental work involved in connecting that predecessor with [i,j]. A pointer is kept pointing to the correspondence [i-1,j], [i, j-1], or [i-1, j-1] which results to this minimum value.

The algorithm for computing the least work solution amounts to setting W(0,0)=0 and computing W(i,j); i=0,...,m; j=0,...,n. W(m,n) is then the optimal total least work and by backtracking the pointers it is easy to find the correspondences which led to this least work solution.

4. Approximation of the original objects with a set of planar contours.

Sederberg's method addresses the problem of morphing two dimensional polygonal contours. In this paper we extend this method for the metamorphosis of three dimensional polygonal objects.

Suppose that we have a pair of three dimensional objects, made up by a set of polygons, that we wish to morph (figure 1). In most practical cases the complexities of these objects make difficult to find an exact solution in respect of energy minimisation. We therefore need to simplify the original objects.

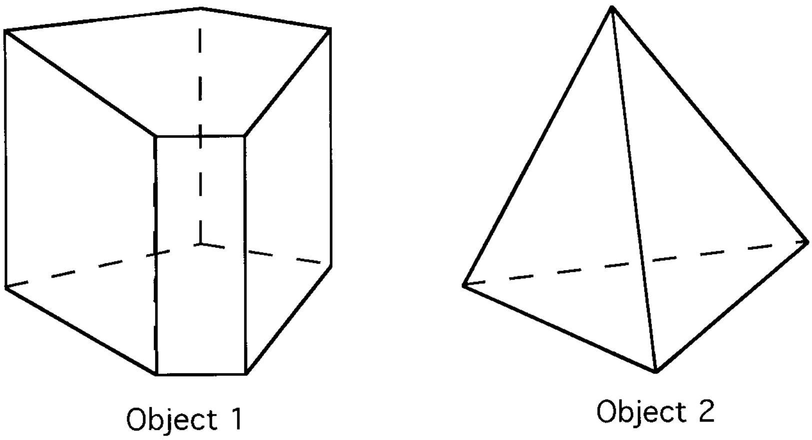


Figure 1

Consider the intersection of an object with offsets of a given plane. This results in a set of parallel planar contours. Depending on the number of contours, we can reconstruct the original model to a reasonable precision using one of the existing algorithms for surface reconstruction from planar contours. We used the method descibed in [FUC77] for the examples we present at the end of this document.

For the time being we will consider situations where the intersection of the object with each of the different planes is a single contour.

We will consider the problem of morphing these two sets of planar contours with least work. Since the contours do not intersect we can order them according to their geometric position in space. We will use the position of a planar contour in this ordering as an index for referencing it.

Suppose that m contours are enough to describe the first object reasonably well, while for the second object n contours are needed. We can order these contours of the first object from 1 to m and for the second object from 1 to n (figure 2).

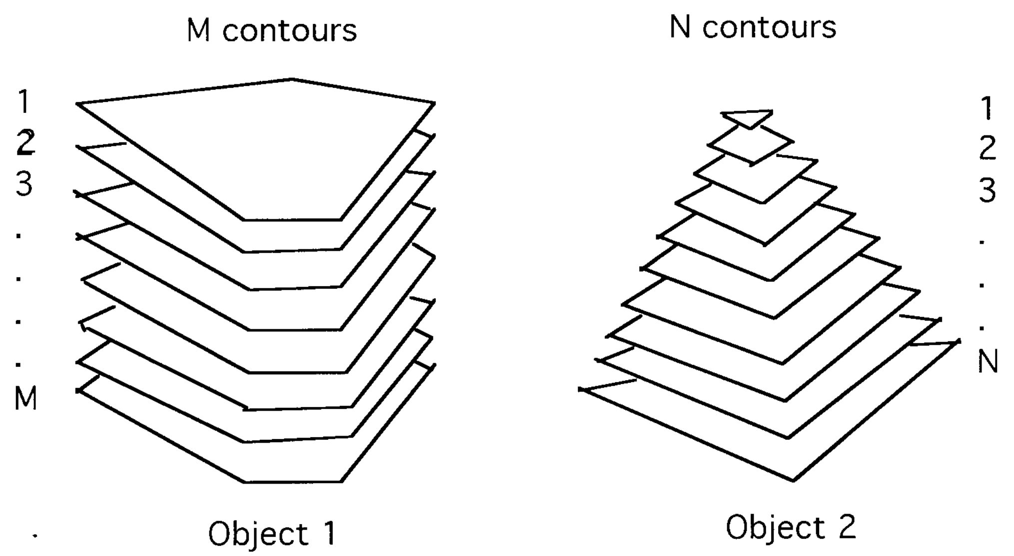


Figure 2

The next section presents a method for determining how the contours should correspond so that the object transformation is accomplished with least work. We will use Sederberg's algorithm to determine the minimum energy needed to transform one contour to another. This algorithm will also determine the correspondence of the contour vertices that results in this minimum energy transformation.

A *valid solution* to the correspondence problem of the contours is one that for every contour of object1 there is one or more corresponding contours of object2, and this leads to an animation sequence without crossovers. Crossovers occur when one contour crosses another one, or two contours are interchanged. To avoid crossovers we expect that if contour i of object1 corresponds to contour j of object2 then there is no contour of object1 with index greater than i corresponding to a contour of object2 with index smaller than j.

We will assume that contours 1 and m of object1 correspond to contours 1 and n of object2 respectively.

5. Expressing the contour correspondence problem as the shortest path problem in a directed graph.

We first construct a directed graph G containing mn vertices denoted [i,j] i=1,...,n and j=1,...,m. The vertex [i,j] represents a correspondence of the i'th contour of object1 with the j'th contour of object2. Furthermore each vertex [i,j] (with i>1, j>1) has three edges incident to it namely ([i-1, j-1], [i,j]), ([i,j-1],[i,j]) and ([i-1,j],[i,j]). This leads to a directed graph having vertex [1,1] as the source and vertex [m,n] as the sink, we will denote these vertices s and t respectively. The vertices s, t represent the known correspondences for the extreme contours of the two objects.

In figure 3 we see the graph constructed for the case of object1 approximated by four contours transformed to object2 approximated by three contours.

The directed graph for the metamorphosis of object1 approximated by four contours to object2 approximated by three contours. Every path from vertex [1,1] to vertex [4,3] represents a valid correspondence for the contours.

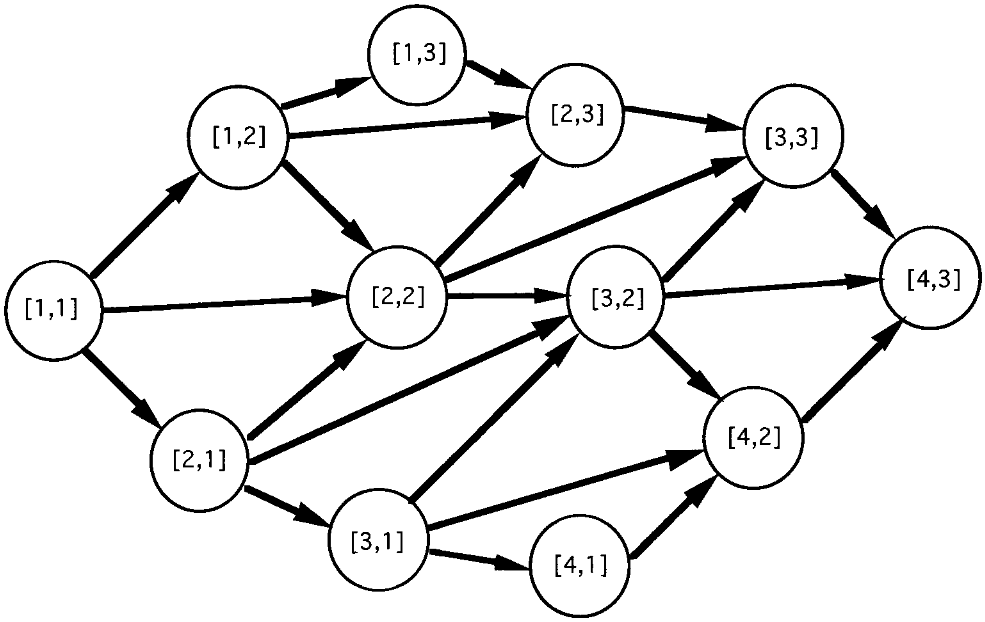


Figure 3

Now let us assign lengths to the edges of graph G, such as edge ([i,j],[k,l]) has a length equal to the energy needed to transform contour k of object1 to contour l of object2 (as given by Sederberg's algorithm).

We will prove that there is one-to-one mapping between the set of the paths in G from vertex s to vertex t and the set of all valid solutions to the correspondence problem of the contours. Furthermore we will prove that the shortest of those paths represents the solution of the correspondence problem of the contours with least energy.

To prove this we must demonstrate that

A) Every path of G from vertex s to vertex t represents a valid solution to the correspondence problem, and

B)For every valid correspondence of the contours, there is a unique path from vertex s to vertex t in G representing it.

We first prove the following:
Lemma 1: First we will prove that if vertices $v=[i_v,j_v]$ and $p=[i_p,j_p]$ of G are in the same path from vertex s to vertex t and vertex v precedes vertex p in the path then $i_v<=i_p$ and $j_v<=j_p$.

We will use induction.

This is true if the distance of the two vertices on the path is 1. Since vertex p is adjacent from vertex v and the only vertices adjacent to p are $[i_p-1,j_p-1]$, $[i_p,j_p-1]$ and $[i_p-1,j_p]$, vertex v must be one of these vertices, which means that i_v is either i_p or i_p-1 and j_v is either j_p or j_p-1. Therefore $i_v \leq i_p$ and $j_v \leq j_p$ holds.

Suppose that the above is true if the distance of these vertices in the path is d.

We will prove below that the above lemma is true if the distance of these vertices in the path is d+1.

Let $g=[i_g,j_g]$ be the vertex immediately preceding p in the path so $i_g \leq i_p$ and $j_g \leq j_p$ (1). The distance of vertices v,g in the path is d, therefore $i_v \leq i_g$ and $j_v \leq j_g$ (2).

from (1) and (2) it follows that $i_v \leq i_p$ and $j_v \leq j_p$.

In order **A** to hold we must prove that
i) *Each path from vertex [1,1] to vertex [m,n] in G contains vertices representing a correspondence for every contour of the two objects.*
ii) *This set of correspondences for the contours will not lead to an overlapping during the animation sequence.*

First we prove **i)**

We will assume that there is a path P from vertex s to vertex t in G that does not include any corresponding contour for a number of contours of one of the objects. Let i be one of these contours with the smallest index (without loss of generality we will assume that this contour belongs to object1). Since i is the contour with no established correspondence with the smallest index there is at least one corresponding contour for contour i-1. Let v=[i-1, l] be the last vertex in the path, representing a correspondence of contour i-1, this vertex v cannot be the last vertex of the path since $i-1<i\leq m$ and the last vertex of the path is vertex t=[m,n]. So there must be an edge leaving v incident to another vertex q of the path. This vertex q cannot have the first element of its ordered pair equal to i (since we assumed there is no implied correspondence for contour i of any vertex of P) or i-1 (since v is the the last vertex of the path, representing a correspondence of contour i-1). This is impossible since the only vertices that are adjacent from a vertex with the first element of the ordered pair equal i-1 are vertices having the first element of the ordered pair either i or i-1.

we now prove **ii)**

Overlapping occurs when the order of contours in object1 and object2 is not maintained when correspondences are established. More precisely for $i<j$ and $k>l$ if the i'th contour of object1 corresponds with the k'th contour of object2 and the j'th contour of object1 corresponds with the l'th contour of object2.

We will prove that vertices [i,k], [j,l] with $i<j$ and $k>l$ cannot belong to the same path from vertex s to vertex t.

If vertices [i,k], [j,l] belong to the same path either vertex [i,k] precedes [j,l] in the path or vice versa. Let us assume that vertex [i,k] precedes vertex [j,l]. From lemma 1 since vertex [i,k] precedes vertex [j,l], $i\leq j$ and $k\leq l$, but that contradicts our initial statement that $k>l$. Similarly it can be proven that it is impossible for the vertex [j,l] to precede the vertex [i,k].

To prove **B** we must show that for every valid solution to the correspondence problem of the contours
i) there is a path representing it.
ii) this path is unique.

i). First we will show that if contour i corresponds to a number of k_i contours of the other object these contours will be adjacent. Let p_{min} be the one of these k_i contours with the smallest index and let p_{max} the one with the largest index. Suppose that the corresponding contours of contour i are not adjacent, then there must be at least one contour with index p $p_{min}<p<p_{max}$ that does not correspond to contour i but to contour k, $k\neq i$. If $k<i$ then we have the situation $k<i$ and $p>p_{min}$ and if $k>i$ we have the situation $k>i$ and $p<p_{max}$ both of these which lead to crossovers (not a valid solution). Therefore in every valid solution a contour i will correspond to a number of k_i adjacent contours of the other object (with $k_i\geq 1$).

Now we will show that in a valid solution if contour i corresponds to contours $k_1,...,k_{1+b}$ and contour i+1 to contours $k_2,...,k_{2+d}$ then $k_2=k_{1+b}$ or $k_2=(k_{1+b})+1$. It is obvious that $k_2>=k_{1+b}$. Suppose that $k_2-k_{1+b}>1$ then there must be a contour k_t with $k_{1+b}<k_t<k_2$ neither corresponding to contour i nor to contour i+1 but to another contour j. It must be either j<i or j>i+1. Suppose j<i since $k_t>k_{1+b}$ our solution is not valid, if j>i+1 since $k_t<k_2$ again the solution is not valid. This contradicts our initial statement that we are considering a valid solution to the correspondence problem of the contours. The contradiction occured from our assumption that $k_2-k_{1+b}>1$, so it must be $k_2-k_{1+b}\leq 1$ therefore $k_2-k_{1+b}=0$ or $k_2-k_{1+b}=1$ (note that $k_2\geq k_{1+b}$).

Let S be a valid solution to the correspondence problem for the contours, this solution will contain a set of ordered pairs (i,j) representing a correspondence of the i'th contour of object1 to the j'th contour of object2. If we order these pairs using their first element as the primary index and the second element as a secondary index, S will be in the form: $(1,1)$, $(1,2)$,...,$(1,k_1)$, $(2, k_1+1)$, $(2,k_1+k_2)$,...$(m, n-1)$, (m,n). So for every two consecutive pairs of S their elements have a maximum difference of one. Since there is an edge in G from every vertex $v=[i_v,j_v]$ to all the vertices $p=[i_p,j_p]$ such that $0\leq i_p-j_v\leq 1$ and $0\leq j_p-j_v\leq 1$ it is obvious that there is a path in G from vertex s=[1,1] to vertex t=[m,n] traversing all the vertices representing the pairs of every valid solution S.

ii) To show that this path is unique, suppose that there are two different paths that represent the same solution to the correspondence problem. These paths must pass from the same set of vertices but traverse this set of vertices in a different order.

Let us find the first place that the two paths differ. Suppose that in the i_{th} position path P_1 has vertex $v=[i_v,j_v]$,and path P_2 has vertex $p=[i_p,j_p]$. Since path P_1 must also contain vertex p, p must follow v in P_1 so $i_v\leq i_p$ and $j_v\leq j_p$ (3). In the same manner path P_2 must contain vertex v so v must follow p in P_2 so $i_p\leq i_v$ and $j_p\leq j_v$ (4). From (3) and (4) it follows that $i_p=i_v$ and $j_p=j_v$ and vertex p is the same with vertex v which contradicts our initial statement that vertices p and v are different.

As we mentioned at the beginning of this section we assign lengths to the edges of graph G, such as edge ([i,j],[k,l]) having a length equal to the energy needed to transform contour k to contour l (as given by Sederberg's algorithm). Every edge of G has a length equal to the energy needed to transform a part of object1 to a part of object2 and the length of a path from vertex [1,1] to vertex [m,n] equals to the energy needed to perform the complete transformation of object1 to object2 according to its equivalent correspondence solution for the contours. So *the minimum energy transformation is accomplished by finding the solution that is represented by the shortest path from vertex [1,1] to vertex [m,n] in G.*

6. Implementation details

First using Sederberg's method we calculate the length of the edges of graph G. When we have the lengths of all the edges of G we compute the shortest path from vertex [1,1] to vertex [m,n]. The method we used for finding this path is a simplified version of Dijkstra's algorithm. Suppose that we want to find the shortest path from vertex [1,1] to vertex [i,j], denoting L(i,j) the length of this path. This is easily accomplished using the simple observation that this shortest path must pass through one of the predecessor vertices of vertex [i,j] namely [i-1,j], [i, j-1] or [i-1,j-1] and the length of it will be equal to the length of the shortest path to one of these predecessors (L(i-1,j), L(i-1,j-1) or L(i,j-1)) plus the length of the edge connecting this predecessor vertex to the vertex [i,j]. So

$$L(i,j)=\min\{(L(i-1,j)+length([i-1,j], [i,j]), L(i,j-1)+length([i,j-1], [i,j]), L(i-1,j-1)+length([i-1,j-1], [i,j])\}$$

When we find the value L(i,j) we store it in a mn matrix and keep a pointer to the predecessor vertex of the path that resulted in this value.

The algorithm for computing the least work solution for the contours amounts to setting L(1,1)=0 and computing L(i,j) i=1,...,m; j=1,...,n. L(m,n) is then the length of the shortest path from vertex [1,1] to vertex [m,n] in G which is equal to the minimum energy for the transformation of object1 to object2. By backtracking the stored pointers we find the correspondence of the contours that results in this minimum energy transformation. Knowing the correspondence of the contours Sederberg's algorithm gives us the correspondence of the vertices. Having the correspondence of vertices we can interpolate their positions and compute the intermediate contour set for the desired number of frames. Then a surface reconstruction method from these sets of planar contours gives us the intermediate object for each frame of the animation sequence.

7. Examples

In figures 4 and 5 we present two animation sequences obtained by the application of this method. In the first one we transform a pot to an airplane while in the second we transform a banana to a glove. We show the original models of the shapes we morph in figure 6. The objects that we use in these examples have very dissimilar shapes, none of them is convex and the objects 'banana', 'glove' are not star-shaped. These examples demonstrate clearly that the method we propose can be applied to generic polygonal objects. A limitation of the proposed method is that it can not yet handle situations where the intersection of the object and a plane results into multiple contours during the slicing step.

In both figures 4,5 the first column contains sets of contours that approximate the object during the metamorphosis. In the first and last line of this column we have a set of contours that approximate the initial and the final object respectively. These contours are obtained by the intersection of the original object (figure 6) with a set of parallel planar planes. The orientation and the number of planes are user defined and this is a way that the user can influence the resulting animation sequence. In the first example (figure 4) we can see that the two terminal objects are approximated with a different number of contours (seven for the first object and ten for the second) while in the second example (figure 5) both terminal objects are approximated by ten contours. By applying our method to these two sets of contours we can find a correspondence for the contours and their vertices. Once the correspondence of the contours and their vertices has been established, the intermediate contour sets are computed by interpolating between each pair of corresponding vertex locations. The preservation of the contour ordering and the usage of Serdeberg's method for determining the vertex correspondence, minimises the situations where any kind of self intersection occurs during the animation sequence. The remaining pictures in column one represent the intermediate contour sets for three still frames of the metamorphosis. Once we have found the correspondence of the contours and have calculated the intermediate contour sets we apply an algorithm for surface reconstruction from planar contours [FUC77] to get the intermediate objects. This algorithm calculates a triangulated surface between every two successive contours. Since all faces in the resulting intermediate objects are triangles they remain planar during the animation. In the second column of figures 4, 5 there are the reconstructed wireframe objects from the sets of parallel planar contours of column 1. The third column of figures 4, 5 contains the same objects of the second column but rendered. These examples were rendered using faceted shading and neutral colours to better illustrate the topological structure of the intermediate models. Since our method uses both topological and geometric information of the original models we avoid undesirable effects such as faces flying apart during the metamorphosis or uneven, distorted transformations (i.e. large parts of one object changing to small parts of another). In each example pay particular attention to the lack of distortion in the intermediate objects.

Even though this method produces even and not distorted transformations it allows parts of an object to extend or compress so that they can be mapped to similar parts of the other object. In the first example (figure 4) we can see that the two terminal objects are approximated with different number of contours (seven for the first object and ten for the second). The lid of the pot which is quite similar with the rear body of the plane is extended through the generation of three more contours.

8. Conclusions and future research

The examples that have been used so far show that the algorithm tends to associate regions of the two objects which look alike. The preservation of the contour ordering and the usage of Serdeberg's method for determining the vertex correspondence minimises the situations where any kind of self intersection occurs. The described method allows some user control over the transformation through mechanisms such as selecting the axis for the slicing of the objects and defining the number of slices for each object. There is an ongoing research for the application of a similar methodology to more general polygonal objects (situations where the intersection of the object and a plane results into multiple contours). A quite obvious approach is to allow the user to break the objects into parts that can be treated by the described method and define correspondences for these parts.

References

[BET89]] Bethel, E. and Uselton, S. "*Shape Distortion in Computer - Assisted keyframe Animation. In State of the Art in Computer Animation*". Magnenat Thalmann,N. and Thalmann, D ., eds., Springer Verlag, New York, 1989, pp. 215-224.

[BUR76] Burtnyk N. and M. Wein , "*Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation*". Communications of the ACM, Vol 19, No. 10, October 1976, pp. 564-569.

[CHE89] Chen,E., and Parent, R. "*Shape Averaging and Its Applications to Industrial Design*". IEEE Computer Graphics and Applications 9,1(Jan. 1989) 47-54.

[CHR78] Christiansen Henry N. and Sederberg Thomas W. "*Conversion of Complex Contour line definitions into Polygonal element mosaics*". Computer Graphics (Proc.SIGGRAPH), 12(3):187-192 1978.

[FUC77] Henry Fuchs, Z. M. Kedem , and S. P. Uselton. "*Optimal surface reconstruction from planar contours*". Comm. ACM, 20(10):693-702, 1977.

[HON88] Hong,T., Magnemat-Thalmann, N. and Thalmann, D. "*A General Algorithm for 3-D Shape Interpolation in a Facet-Based Representation*". Proceedings of Graphics Interface '88 (June 1988) 229-235 .

[[HUG92] Hughes John F. "*Scheduled Fourier Volume Morphing*". Computer Graphics (proc. SIGGRAPH '92) 26(2): 43-46 1992.

[KAU91] Kaul, A and Rossignac,J. "*Solid-Interpolating Deformations: Construction and Animation of PIPs*". Proceedings of Eurographics '91.

[KEN92] Kent, J, Parent, R. and Carlson, W. "*Shape Transformation for Polyhedral objects*" Proceedings of SIGGRAPH '92 (Chicago, July 26-31, 1992) 47-54.

[PAR91] Parent, R. "*Shape transformation by boundary representation interpolation: a recursive approach to establishing face correspondences*", Journal of Visualisation and Computer Animation. Vol 3 Iss 4, pp 219-239 1992.

PAY92] Payne, B. and Toga, "*A distance Field Manipulation of Surface* Models". IEEE Computer Graphics and Applications 12, 1 (Jan. 1992) 65-71.

[SED92] Sederberg Thomas W. and Greenwood Eugene. "*A Physically Based Approach for 2-d Shape Blending*". Computer Graphics (Proc. SIGGRAPH) 26(2):25-34, 1992.

[TER89]] Terzides, C. "*Transformational Design. Knowledge Aided Architectural Problem Solving and Design*", NFS Project#DMC-8609893, Final Report, (June 1989).

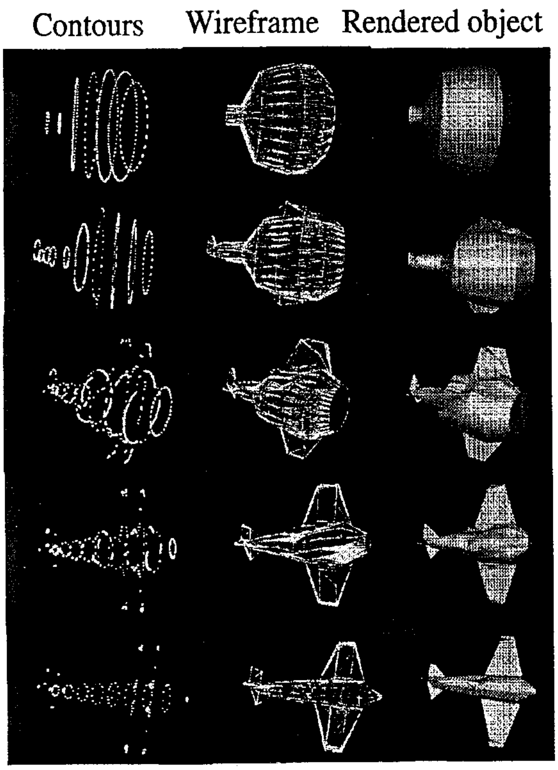


Figure 4

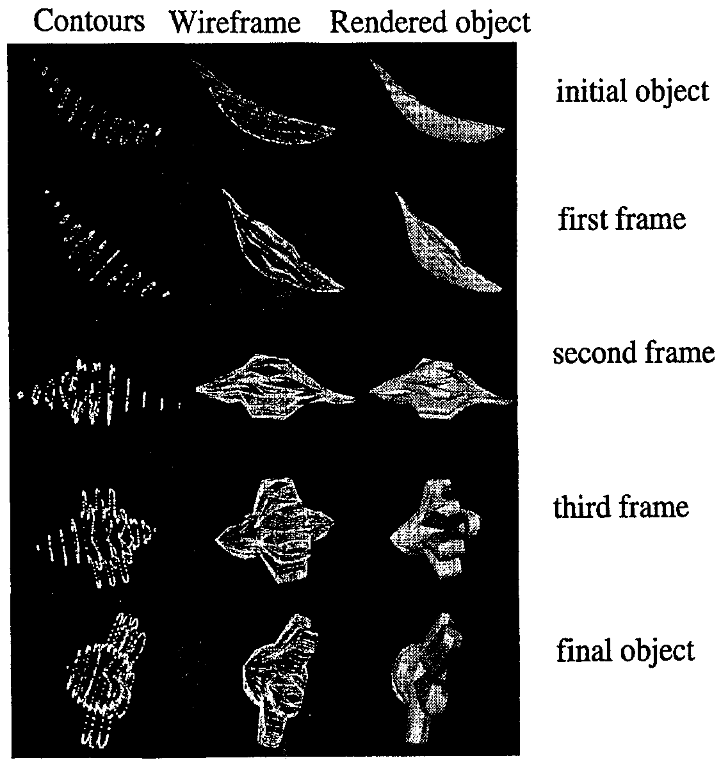
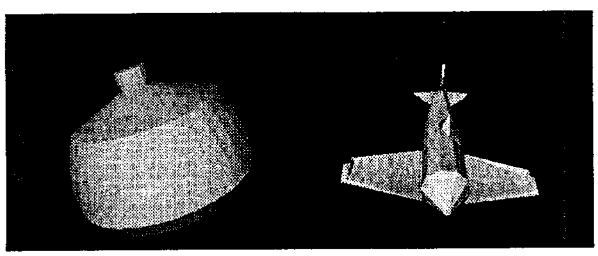
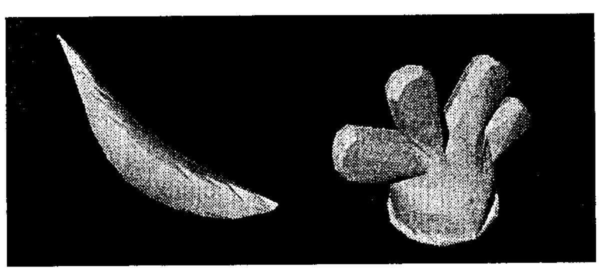


Figure 5



The original models 'pot' and 'plane' for the first metamorphosis



The original models 'banana' and 'glove' for the second metamorphosis

Figure 6