Multidimensional free-form deformation tools

Dominique Bechmann

LSIIT ULP-CNRS UPRES-A 7005, Department of Computer Science, Louis Pasteur University, Strasbourg, France

Abstract

A survey of free-form deformation tools where the deformation is controlled by manipulating a 0-D to 3-D tool. Characteristics of a model that includes all these and generalises the concept will be presented.

1. Introduction

In this state of the art, we propose a survey of free-form deformation techniques where the deformation is controlled by manipulating a user-defined deformation tool. Recent researches have provided deformation tools of any topological dimensions: 0-D (points), 1-D (lines), 2-D (surfaces) or 3-D (volumes).

The most famous of these models (see section 2) is certainly the one using 3-D parallelepipedical lattices and all its extensions. The latest invented is using 2-D rectangular surfaces as deformation tool (see section 3). Axial deformation are specifying the deformation through the modification of a 1-D line (see section 3). Finally, with models such as the deformation is simply defined by several user-defined point displacements i.e. 0-D constraints (see section 4).

These deformation techniques are all independent of the underlying object representation. It can be proved that the underlying mathematical formalism for several of these models is just the same. The fact that they fall into a unique mathematical formalism involving a mapping establishes the links between these models. Thus, we will study the differences and the links from one formulation to the other.

However for the interactive modeling, the important questions are more : what kind of deformation can be easily achieved with these techniques? Are they highly interactive and intuitive? Thus, to answer these questions, we will explain, for each class of models, how to control the resulting deformation by manipulating the deformation tool. In particular, we will precise the position, the size and the boundary of the deformed area as well as the shape of the deformation. So a formal comparison as well as a practical one will be done (see section 5).

Finally, we will introduce the characteristics of a free-form deformation model that would generalize and include all these deformation models using a multidimensional deformation tool (see section 6).

2. 3-D deformation tools

Deformation models requiring a 3-dimensional deformation tool such as a parallelepipedical volume called lattice are presented in this section. Using these techniques the deformation of an object is computed from the deformation applied to the 3-dimensional deformation tool.

2.1. FFD

The deformation tool used for the free-form deformation technique called FFD is a trivariate volume defined by an array of control points. To deform an object the user deforms the lattice by moving its control points. Any point lying inside the lattice is deformed accordingly to the lattice deformation. In particular, the deformation of an object inside the lattice follows the displacement of the lattice control points.

Before deforming the lattice, each point of the object should be associated to the lattice. Let be the cartesian co-ordinates of a point in the global co-ordinate system and be its co-ordinates in the lattice co-ordinate system. The transformation of from its global co-ordinates to its lattice co-ordinates is obtained from:

\[
\begin{align*}
\text{ud}_1 &= \frac{S \cdot T \cdot (U - U_0)}{S \cdot T \cdot R} \\
\text{ud}_2 &= \frac{S \cdot T \cdot (U - U_0)}{S \cdot T \cdot R} \\
\text{ud}_3 &= \frac{S \cdot T \cdot (U - U_0)}{S \cdot T \cdot R}
\end{align*}
\]
In particular, elementary or composite prismatic lattices are the FFD technique to allow non-parallelepipedical lattices. Extended free-form deformation or shorter EFFD 12, extends the FFD technique to allow non-parallelepipedical lattices. Extended free-form deformation or shorter EFFD 12, extends

2.2. Extended FFD

Extended free-form deformation or shorter EFFD 12, extends the FFD technique to allow non-parallelepipedical lattices. In particular, elementary or composite prismatic lattices are defined. Elementary prismatic lattices are obtained by moving or merging control points of a parallelepipedical lattice. For example, the cylindrical lattice is obtained by welding two opposite faces of a parallelepipedical lattice and by merging all control points of the cylinder axis. Composite prismatic lattices are defined as several elementary lattices welded together. The welding operation is realised by merging the control points of each lattice. Some continuity problems may occur specially when several control points are merged together. In 12 one can see on the figure 10b page 196 that the continuity of the surface in the center of the lattice can not be checked. However in this example the fold effect was expected, and gives a visually satisfying result. As continuity constraints would be penalizing for this technique, the authors insure lattice continuity only for the simplest cases. Non prismatic lattices can also be used. However the use of complex lattices can lead to unpredictable results.

In addition, elementary lattices are composed by several "chucks" where a chunk is a trivariate volume represented by a tensor product of Bernstein polynomials of degree three. Such a chunk is defined by 4 x 4 x 4 control points.

To compute the deformation due to an elementary lattice, the co-ordinates of the object points in the lattice parameter space are computed. First, the chunk where the point is supposed to lie is determined. Then, the co-ordinates inside the chunk are computed using Newton approximation. Finally, the deformation of a point inside a given chunk is defined by

$$D(U) = \sum_{i=0}^{3} \sum_{j=0}^{3} \sum_{k=0}^{3} P_{i,j,k} B_i^3(u_1) B_j^3(u_2) B_k^3(u_3)$$

where the $B_i^3$, $B_j^3$, $B_k^3$ are Bernstein polynomials of degree three and the $P_{i,j,k}$ are the 4 x 4 x 4 control points with $0 \leq i + j + k \leq 3$. When a composite lattice is used, the deformations due to each elementary lattice are applied successively.

The definition of elementary lattices in chunks provides a local control of the deformation. Indeed, the displacement of a given control point modifies only the points lying inside the corresponding chunk. It is possible to localize a deformation by modifying the number of subdivisions of the lattice. However, the subdivision process is not easy to understand for a user not familiar with the Bernstein polynomials. Unfortunately, no intuitive tools are provided to control the deformed area: its size, its position and its boundary.

The deformed area corresponds exactly to the volume of the lattice only when every chunk is modified. Then, the user controls the deformed area by editing the position, the size and the boundary of the lattice. As with parallelepipedical lattices, the deformation could also be local when only a limited part of the object lies inside the lattice.

To conclude, the deformed area corresponds to the volume embedding the interior of the modified chunks but no intuitive tools are provided to control it. So, this technique is recommended for global deformation obtained when every chunk is modified through the deformation. Only in that
2.3. Rational FFD

Rational free-form deformation \(^{18}\) is another extension of FFD. It allows incorporation of weights defined at each control point of the parallelepipedal lattice. However, when the weights at each control point are unity, the deformations are equivalent to the FFD. To control the deformation, the user either moves the lattice control points or modifies their associated weights. The co-ordinates of a point are computed in the lattice parameter space before editing the lattice of control points. Then, the deformed point is computed from:

\[
D(U) = \frac{\sum_{i,j,k} W_{i,j,k} P_{i,j,k} B_{p_1}^i(u_i) B_{p_2}^j(u_j) B_{p_3}^k(u_k)}{\sum_{i,j,k} W_{i,j,k} B_{p_1}^i(u_i) B_{p_2}^j(u_j) B_{p_3}^k(u_k)}
\]

where \(W_{i,j,k}\) represents the weight associated to the control point \(P_{i,j,k}\) with \(0 \leq i \leq p_1, 0 \leq j \leq p_2, 0 \leq k \leq p_3\) and where the \(B_{p_1}^i, B_{p_2}^j, B_{p_3}^k\) are the polynomials of degree \(p_1, p_2, p_3\).

The deformed area and the shape of the deformation both depend on the polynomial basis. The implementation of Rational FFD is presented with Bernstein polynomials of degree \(p_1, p_2, p_3\) respectively. Thus, the deformed area corresponds to the lattice. The difference with FFD lies in the fact that a weight associated at each control point provides one more degree of freedom to define the deformation. However the unpredictability of the deformation obtained by changing the weight at a control point could be a limitation of this technique for the uninstantiated user.

2.4. Others FFD techniques

Actually, other extensions of the classical FFD model called NURBS-based FFD \(^{16,19}\) adopted a trivariate B-Spline volume as deformation tool. The principle is the same but, due to the B-Spline properties, local deformations can be easily defined. Indeed, by moving one control point of the lattice, only a limited area of space defined by some chunks of the lattice is deformed.

In order to obtain a deformation tool which can describe any 3D space subdivision, and to permit an easy control of the continuity of an object embedded in these lattices, deformation models \(^5,22\) based on the same principles as the ones of FFD are defined.

With the first one called Continuous FFD \(^5\) by the authors, the deformation tool is a set of tetrahedral Bézier volumes. Initial lattices are made of regular Bézier tetrahedrons, i.e., tetrahedrons which control points are regularly distributed with respect to the four vertices. This permits to directly compute the local coordinates of the object points in the lattices, without using a Newton algorithm. This model which can describe any subdivision of the tridimensional space also permits to control the \(C^1\) continuity of the deformation of an object embedded in these lattices.

The other one \(^{22}\) uses a lattice of arbitrary topology that is subdivided with the Catmull-Clark subdivision method. After the first subdivision all cells of the resulting lattice have a hexahedral structure. After a few subdivisions, all cells are arranged in a regular pattern except at a finite number of points. Finally, the initial lattice of arbitrary topology is refined \(n\) times until the largest cell of the lattice has a volume less than a specified size. The local co-ordinates of each point of the object in the subdivided lattice is computed. To deform the object the control points of the initial lattice are moved. Then the deformed lattice is once again subdivided \(n\) times. The new position of each point is easily obtained since its local co-ordinates according to a cell are invariant through the subdivision process.

Since the user only manipulates the initial lattice of arbitrary topology and not the refined one, the link between the displacement of control points and the deformation of the object might be quite hard to predict. Some problems also come from the subdivision process itself: very small cells together with very large ones can be obtained in the resulting lattice. The resulting deformation may suffer from such configuration. In addition, the deformed area is arbitrarily constrained by the lattice topology.

3. 2-D deformation tools

In this section, the object is controlled through the manipulation of a two-dimensional deformation tool. A model \(^{15}\) in which a so-called shape surface is used to control the deformation of an object is introduced.

The shape surface is defined by a B-Spline tensor product surface \(S(u,v)\), initially forming a rectangular planar grid on the \(X0Y\)-plane. To deform an object, the user deforms the associated shape surface by moving its control points \(P_{i,j}\).

First, for each point \(U = (u_1, u_2, u_3)\) of the object, its projection \(U_P\) on the shape surface and along its normal is computed. Let \((u_1', u_2')\) be the parametric co-ordinate of \(U_P\) such that \(U_P = S(u_1', u_2')\) and \(u_3\) be the distance between \(U\) and \(U_P\). So we have:

\[
U = S(u_1', u_2') + u_3 N(u_1', u_2')
\]

where \(N(u, v)\) is the vector normal to the shape surface.

The deformation model is defined such that the coordinate \((u_1', u_2', u_3')\) of any point \(U\) relatively to the shape surface are invariant. This property allows to compute the new position \(D(U)\) of \(U\):

\[
D(U) = S(u_1', u_2') + u_3' N(u_1', u_2')
\]

where \(S(u, v)\) is the modified shape surface obtained after moving its control points and \(N(u, v)\) is the vector normal of the modified shape surface.
Through the control of a shape surface, it is quite easy to bent or to twist any object. However to obtain, for example, a tapered object, the user should be able to modify through the deformation, the distance between a point and its projection on the shape surface.

The authors propose to use the z component of a so-called height surface \( H(u, v) \) as an additional parameter such that:

\[
D(U) = S(t(u_1, u_2)) + H_z(u_1, u_2)\cdot t_3 N(u_1, u_2)
\]

Initially, the height surface is a B-Spline tensor product surface forming the same rectangular planar grid than the shape surface, but on the plane \( z = 1 \).

The user controls the deformation by moving the control points of the shape surface and of the height surface. The shape of the deformation is strongly linked with the modified shape surface. Precise displacements of the object are hardly obtained.

The deformed area is not explicitly defined. Indeed, the deformation is global if the projection of the object lies totally on the shape surface. Then, global deformations such as bending, tapering and twisting can be accomplished.

Otherwise, the deformation of the object is local and, to insure the continuity of the surface of the deformed object, the user should not move some control points of the shape surface.

4. **1-D deformation tools**

Deformation models where the object deformation is controlled by a one-dimensional deformation tool i.e. an axis are presented in this section. The object deformation is linked to the axis deformation.

4.1. **AxDf**

Axial deformation called AxDf \(^{20}\) is explained here. The first step is to attach the axis to the object. To each point \( U = (u_1, u_2, u_3) \) of the object a point \( A_U \) on the axis is associated. A local co-ordinate system is defined around each point \( A_U \) of the axis. Then, the co-ordinates \( U = (u_1, u_2, u_3) \) of the object point in the local co-ordinate system associated to \( A_U \) are calculated using the matrix \( R_1 \). The axis is deformed and to a point \( A_U \) on the initial axis corresponds a point \( A_{\bar{U}} \) on the deformed axis. The co-ordinates \( D(U) \) of the deformed point in the local co-ordinate system associated to \( A_{\bar{U}} \) are equal to the co-ordinate of \( U = (u_1, u_2, u_3) \) in the local co-ordinate system associated to \( A_{\bar{U}} \). The co-ordinate of \( D(U) \) in the global co-ordinate system are calculated using matrix \( R_2 \) as follows:

\[
D(U) = T_2 R_2 (T_1 R_1)^{-1} U
\]

where \( T_1 \) is the translation matrix of vector \( OA_U \) and \( T_2 \) is the translation matrix of vector \( OA_{\bar{U}} \).

The shape of the deformation is linked to the axis deformation. The only deformations that can be obtained are specified by bending or stretching the axis. In addition, scaling and twisting operations are possible by associating to each point of the axis a scale and a twist factors before and after the deformation. Finally, the axial deformation of a point is defined by:

\[
D(U) = T_2 S_2 W_2 R_2 (T_1 S_1 W_1 R_1)^{-1} U
\]

where the matrix \( W_1 \) of twisting angle \( q_1 \) and the matrix \( S_1 \) of scaling factor \( \theta_1 \) are specified at each point \( A_U \) of the axis, and the matrix \( W_2 \) of angle \( q_2 \) and the matrix \( S_2 \) of factor \( \theta_2 \) are applied to the deformed object points associated to \( A_{\bar{U}} \).

Axial deformation influences the whole three-dimensional space although it is applied only to the point representing an object. In order to localize the deformed area, a zone of influence is introduced to defined the portion of the three-dimensional space to be deformed. A general cylinder around the axis is proposed to define the deformed area. It defines the size, the position and the boundary of the deformed area.

4.2. **Deformation using De Casteljau algorithm**

An other deformation model \(^{14}\) using 1-dimensional deformation tool is introduced here. The 1-dimensional deformation tool is a Bézier curve.

A Bézier curve is defined by a set of control points forming the control polygon. The De Casteljau algorithm transforms a segment of the polygon of control into a segment of the curve by an iterative process.

Initially, the object to deform is mapped by an affine transformation on each segment of the control polygon. Then by applying the De Casteljau algorithm, the resulting object takes the shape of the Bézier curve.

Only global deformation such as stretching, bending, twisting, tapering can be obtained with this model. To obtain local deformation the authors suggest to use De Boor algorithm \(^{21}\) instead of De Casteljau.

5. **0-D deformation tools**

For the models presented in this section the deformation tool is reduced to points such that the deformation of the object follows the displacement of these points.

5.1. **DOGME**

The deformation model called DOGME \(^{3,4}\) is summarized in this section. The deformation of an object is defined by the displacement of points called constraint points. In particular, a constraint point can be an object point. Then, it is trivial to achieve exact placement of object points. The model can
satisfy as many constraints as the user enters unless two opposite constraints are applied to a unique point in space. In that case, the system computes the best approximation to the solution.

The deformation is expressed as the composition of an extrusion function \( f \) and a matrix \( M \) computed so as to achieve the constraints:

\[
D(U) = U + \sum_{i=0}^{n-1} \frac{M_i f(U)}{(n-1)!}
\]

The resolution method is based on pseudo-inverses. The dimensions of the matrix \( M \) and of the function \( f \) are written underneath the system.

The shape of the deformation around each constraint point depends on the so-called extrusion function \( f \).

Local deformation around the constraint point is obtained using B-Splines polynomials for example. Due to the local support of the B-Splines polynomials only a limited area of space is deformed. Furthermore, the shape of the extrusion function is imprinted on the deformed area. Using B-Splines of degree greater or equal to two, the shape of the resulting deformation is very intuitive and smooth. Arbitrary shaped bumps can easily be designed using this technique.

A bounding box centred at each constraint point visualizes the extent of the deformation. The user interactively manipulates it to localize the deformation or to modify its shape.

The size of the bounding box is linked to the support of the B-Splines polynomials. Due to their local properties, the displacement of one control point influences a limited space area. The shape of the B-Splines polynomials is imprinted in the area surrounding a given displaced point. The size, the position and the boundary of the deformed area are strongly linked to the distribution of the control points of the lattice. Thus, to control the locality of the deformation, the user has to modify the initial lattice. No tools are offered to the user to control the deformed area. Due to that, part of the advantages of the direct manipulation are lost.

Multiple levels FFD is introduced to solve problems that may occur when several constraints influence the same control point of the lattice. The lattice is recursively refined by inserting control points. The resulting deformation tool is a multi-level lattice where each lattice is contributing to the displacement of only one constraint point.

### 5.3. Dirichlet FFD

A very interesting deformation model called Dirichlet free-form deformation is presented in this section. This deformation model uses a 0-dimensional deformation tool since the deformation of the object follows the displacement of a set of points.

A set of points called here control points is initially positioned in space. The convex hull of this set of control points is considered. The Delaunay triangulation and the associated Voronoi diagram of this set of control points are computed. For a given point \( P \), its natural neighbors can be defined as the control points with which \( P \) would share an edge in the Delaunay triangulation. Then \( P \) is linearly defined by its Sibson or natural co-ordinates defined over the set of natural neighbors in terms of areas in 2 dimensions or volumes in 3 dimensions. The region defined by \( P \) and its natural neighbors is the region of influence of \( P \). Then, a multivariate Bézier simplex is built over the set of its Sibson neighbors such that the Bernstein polynomials value is tending to 1 when approaching \( P \) and tending to 0 when approaching the border of the region of influence. Such a surface is called a Dirichlet surface.

Any point \( U \) in space so in particular points of an object can be expressed in term of the Dirichlet surface associated to its region of influence:

\[
U = \sum_{l=0}^{n} P_l B_l^n(U)
\]
where \( I = (i_1, i_2, \ldots, i_n) \) represents a multi-index, \( P_I \) are the control points, \( U = (u_1', u_2', \ldots, u_n') \) are the Sibson coordinates of \( U \) and \( B_{m, I}^n \) the \( n \)-variate Bernstein polynomials of degree \( m \).

This expression is invariant through the displacement of the constraint points, so the deformation \( D(U) \) of a point \( U \) is defined as follows:

\[
D(U) = \sum_{I=0}^{m} P_I B_{m, I}^n(U)
\]

The author also explains how to integrate various extensions to this deformation model. A weight can be associated to each control point in the same way as in Rational FFD. Exact displacement of an object point is obtained by identifying it with a control point.

Finally, Dirichlet FFD seems very convenient for local deformations. Indeed its application to hand modeling and animation needs a precise control of the deformed area. For each control point the deformed area is the region of influence defined by its natural neighbors. So the deformed area can be of any shape. The shape of the deformation in the deformed area is given by the Bernstein polynomials. Global deformation are also possible by moving control points in a group.

6. Comparison between multidimensional deformation techniques

To compare these deformation models it is important to establish the links between their mathematical formalism. But it is also crucial to compare them in terms of capabilities to deform objects.

All these models involve a mapping represented by the deformation function \( D : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \) (except \( \mathbb{R}^2 \) in \( \mathbb{R}^3 \)) that associates with each point \( U \) its new position \( D(U) \). In addition, it can be proved that the mathematical model defining techniques using 3-D deformation tools such as FFD and using 0-D deformation tools such as DOGME is the same. Unfortunately, models using 1-dimensional or 2-dimensional tools are hardly included in the same formalism.

Eventhough, all these techniques involve a mapping represented by the deformation function, the interactive techniques they involve to define the deformation using the deformation tool are quite different.

Table 1 shows for which kind of deformations the different models are appropriate. For example, it is much easier to obtain exact displacement of object points using 0-dimensional deformation tool than by manipulating a 3-dimensional lattice. Deformation techniques using 0-dimensional tool are particularly well suited to imprint arbitrary shaped bumps. Axial deformation is well adapted to bending or stretching. Deformations with a 3-D lattice seems to be appropriate if it involves a regular set of control points.

The shape of the deformation is the same when using 3-dimensional or 0-dimensional deformation tools since it is simply linked to the polynomials (Bernstein, B-Splines, etc). However, the extrusion function of DOGME that defines the shape of the deformation is not restricted to polynomials. Thus, although these models are meant to provide free-form deformation, physical properties on the deformed object can be obtained with an appropriate choice for the extrusion function.

The extent of the deformation is predictable in all these methods. However, to control the position, the size and the boundary of the deformed area, DOGME provides a more flexible tool than others. For each implementation of the deformation models (first and second columns), Table 2 indicates which entity defines the deformed area (third column) and also indicates if the deformed area is visualized (fourth column).

Indeed, within DOGME only the points lying inside the bounding box move. The bounding box allows to control the extent of the deformation. It corresponds exactly to the deformed area.

Two techniques using a 1-dimensional deformation tool were presented. Deformation using De Casteljau algorithm is applied to the whole object so only global deformations are provided for now. The authors mention that local deformations can be achived using De Boor’s algorithm. AxDF also influences the whole space but the authors explain that it could be limited to a generalized cylinder.

When a shape surface controls the deformation, every point projecting on it is inside the deformed area. Initially the shape surface is planar, forming a regular grid, so the deformed area is simply a parallelepiped infinite in the direction perpendicular to the shape surface.

Using FFD techniques although only the portion of space inside the lattice is considered, no mention is made of the deformed area. Actually, when the lattice is defined by a trivariate Bernstein polynomial, its volume defines the deformed area. However with a different polynomial basis such as B-Splines, the size and the boundary of the deformed area when moving one control point depends on various parameters such as the number of control points of the lattice, their distribution through space and the degree of the polynomials. Thus, it is hardly predictable for a user not familiar with the effect of control point displacements. The Direct FFD model relying on FFD techniques has the same characteristics.

The function \( D \) defines the deformation of the whole space since it expresses the transformation of any point in \( \mathbb{R}^3 \). Although the whole space is deformed, the deformation can be applied to a selected set of points only. Consider an object represented by a set of points with topological relations. The deformation modifies the position of the points independently of their topology. This property of indepen-
Bechmann / Deformation

<table>
<thead>
<tr>
<th></th>
<th>Global deformation</th>
<th>Local deformation</th>
<th>Precise displacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-D FFD, EFFD, RFFD</td>
<td>yes</td>
<td>continuity problems</td>
<td>no</td>
</tr>
<tr>
<td>3-D NURBS-based FFD</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>3-D Continuous FFD</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>3-D Arbitrary topology FFD</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Deformed area</th>
<th>Visualized ?</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-D FFD, RFFD</td>
<td>parallelepipedical lattice</td>
<td>yes</td>
</tr>
<tr>
<td>3-D EFFD</td>
<td>some prismatic or any shape chunks of the lattice</td>
<td>no</td>
</tr>
<tr>
<td>3-D NURBS-based FFD</td>
<td>some parallelepipedical volumes</td>
<td>no</td>
</tr>
<tr>
<td>3-D Continuous FFD</td>
<td>some tetrahedral volumes</td>
<td>no</td>
</tr>
<tr>
<td>3-D Arbitrary topology FFD</td>
<td>some hexahedral volumes</td>
<td>no</td>
</tr>
</tbody>
</table>

Table 1: Which kind of deformations for which deformation model?

<table>
<thead>
<tr>
<th></th>
<th>Deformed area</th>
<th>Visualized ?</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-D Surface</td>
<td>the space projecting on the shape surface</td>
<td>no</td>
</tr>
</tbody>
</table>

Table 2: Possible shapes of the deformed area

Evidence from the underlying representation of the object is common to all deformation methods presented here. Due to this independence, these methods can easily be integrated into most existing modelers or animation systems.

7. Discussion

The characteristics of a deformation model that includes all these and generalises the concept are discussed in this section.

First of all, the model should allow a large variety of deformation: global deformation such as bending, tapering, twisting, etc. and local deformation with a precise control of the deformed area and the deformation itself.

In term of interactive tools used to deform objects, it should be of any dimension or even of various dimension. More precisely the deformation tool could be a skeleton of any shape and any dimension.

The ideal model would be a mathematical generalisation of the various existing formulations. In fact it could be a multi-model that would use one existing model or the other depending on the skeleton. A skeleton reduced to a 1-dimensional tool could simply use AxDf for example. While a multi-dimensional skeleton could use a different model for each part.

Finally, such a model would be helpful for animation since some animation models are already founded on these free-form deformation models.
References


2. F. Aubert, D. Bechmann, “Animation by Deformation of Space-Time Objects”, *EUROGRAPHICS*’97, 21(5), Budapest (Hongrie) (4-8 September 1997).


