Volume Distortion and Morphing Using Disk Fields

MIN CHEN, MARK W. JONES AND PETER TOWNSEND

Department of Computer Science, University of Wales Swansea,
Singleton Park, Swansea SA2 8PP, United Kingdom.
E-mail: {m.chen, m.w.jones, p.townsend}@swansea.ac.uk

Abstract — *The rapid development of volume graphics offers ample scope to develop new methods and tools for computer graphics and visualisation. This paper is concerned with methods for specifying and controlling complex transformations of volumetric data. A volume distortion algorithm is presented, which allows one volume to be distorted into another under the control of two sets of disk fields. The algorithm also forms a primary module in a volume morphing process that is an extension of the feature-based image morphing technique developed by Beier and Neely [1]. For each inbetween volume, the algorithm is of time complexity O(**n·m**) depending on the size of the volume **n** and the number of disk fields **m**. Tests have shown that the algorithm is capable of performing complex volume transformations within an acceptable time using a small set of disk fields. The merits of the method in comparison with existing approaches for volume distortion and morphing are also discussed.*

Keywords — *volume visualisation, morphing, distortion, animation.*

1. INTRODUCTION

Image distortion is concerned with the application of geometric transformations to digital images. Early interest in this technique occurred in the 1960's when it was primarily used for geometric correction in remote sensing [2]. During the past few years, the technique was developed into a class of methods for computer animation, commonly called image morphing, which played an important role in creating stunning visual effects in the entertainment industry. By extrapolating the concepts of distortion and morphing to a three dimensional analogue, we can derive methods for transforming volumetric data, for which many visualisation tools are nowadays widely available. With the rapid development of volume graphics hardware and software [3], it is believed that graphics tools for volume distortion and morphing will have many applications in areas such as computer animation, scientific visualisation, medical imaging and aspects of natural science.

Volume distortion defines a geometric transformation from one volume to another under the influence of two control datasets, each of which usually consists of a set of geometric objects representing the main features of a volume. Given two volumes, a morphing process generates a sequence of inbetween volumes that represent a smooth transformation from one to another. The morphing process often consists of a distortion module. According to the use of control datasets, approaches to volume morphing can be classified into three categories [4], namely:

- **Cross Dissolving** — which requires no control datasets. The simplest cross dissolving method is a linear interpolation between the two volumes in the spatial domain [5]. The method normally yields unsatisfactory results, especially with binary volumetric data. To enhance the smoothness of the inbetween volumes, the Fourier transform has been used to schedule the interpolation in the frequency domain by favouring high-frequency components that are defined by the threshold of an interested iso-surface [6]. More recently, wavelet transform, which encapsulates both frequency and spatial information, has been employed in volume morphing in a multi-resolution manner [7].

  The methods in this category are generally easy to use and require little human interference. The smoothness of the inbetween volumes is well achieved by the wavelet-based method, and to some extent, by the Fourier-based method. However, both have difficulties in specifying slightly complex geometric transformations such as object rotation. By relying on the frequency information, the methods would also have difficulties in being applied to more general forms of volumes where voxels are associated with colours, opacities and texture information. Moreover, the wavelet-based method introduces a correspondence problem. The solution to the problem proposed in [7] is dependent on the order of the three axes, and constrained by the segment consecution in each dimension, therefore leaving much to be desired.

- **Mesh Warping** — where control datasets define volume subdivisions as well as coordinate mappings. In 2D mesh warping [2], the control dataset associated with each image specifies a planar subdivision over the image, typically a parametric grid or a triangular mesh. Extrapolating from the method to a three dimensional analogue, two volume warping methods have been derived [4]. Given a set of points, a volume subdivision can be automatically generated, for instance, by 3D triangulation. In mesh warping, the distortion is constrained by individual elements, and it is therefore relatively easier to achieve a desired transformation without causing "ghost shadows" [1] provided there is no "fold-over" structure. However, in most cases, both methods require a very large number of control elements, without which some undesirable visual artifacts would be quite obvious [4]. The manipulation of 3D subdivisions through a user interface also seem to be somewhat problematic.

- **Field Morphing** — where control datasets are used to specify features of volumetric data and coordinate mappings. Although it requires user input of the control datasets, feature-based morphing has demonstrated its flexibility and controllability in image metamorphoses [1, 8]. The extensions of existing 2D methods have been proposed, including the use of point fields [9] and line fields [4]. Like in the image morphing, volumes are first distorted with an inverse mapping [1] determined by pairs of fields, and then linearly interpolated. A method using a combination of different fields, which include points, lines and boxes, has also been reported recently [10]. The naive extensions of point and line fields suffer from the inability to specify an arbitrary 3D coordinate mapping. The problem can be solved by introducing a supplementary vector in each line field (or two vectors in each point field) [4], or by solving a set of error functions [9]. However, unlike their 2D progenitors, such fields are generally difficult to define and manipulate.

In the remainder of this paper, a method for volume distortion and morphing using control fields of a disk shape is presented, and its merits over other distortion and morphing methods are discussed. A general feature-based volume morphing process is given in Section 2, and the concepts of disk fields, together with a volume distortion algorithm, are described in Section 3. The results of a number of tests are reported in Section 4, accompanied by concluding remarks.

2. FEATURE-BASED VOLUME MORPHING

Given a starting volume $\mathbf{V}_a$ and a finishing volume $\mathbf{V}_b$, morphing is a process that generates a sequence of inbetween volumes $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_l$ which represent a smooth transformation from $\mathbf{V}_a$ to $\mathbf{V}_b$. In the following discussion, we assume that each of these volumes is a collection of voxels organised in the form of a three dimensional grid, and all volumes are of the same size in each dimension. Similar to many methods used for image morphing [1, 2], $\mathbf{V}_a$ and $\mathbf{V}_b$ are usually associated with a pair of control datasets, $\mathbf{F}_a$ and $\mathbf{F}_b$, which specify the correspondence of key features between the two volumes. Figure 1 shows a general volume morphing process that employs two primary modules, namely *interpolation* and *distortion*.

For the i^{th} inbetween volume $\mathbf{V}_i$, an intermediate control dataset $\mathbf{F}_i$ is obtained by linear interpolation with a parameter $t = \frac{i}{l+1}$. Taking $\mathbf{V}_a$ and $\mathbf{V}_b$ as *source* volumes, the distortion module computes two *destination* volumes, $\mathbf{V}_{ia}$ and $\mathbf{V}_{ib}$, under the influence of corresponding control datasets. The target volume $\mathbf{V}_i$ is then simply an interpolation of the two destination volumes. When the distortion module degenerates into a simple copying operation, the morphing process results in a sequence of volumes cross-dissolving from $\mathbf{V}_a$ to $\mathbf{V}_b$.

Under the control of a set of fields, the distortion module maps a voxel in the destination volume onto multiple voxels in the source volume, from which a single voxel (or voxel value) is then determined. Disk fields, which are to be detailed in Section 3, provide an effective way of controlling such transformations.

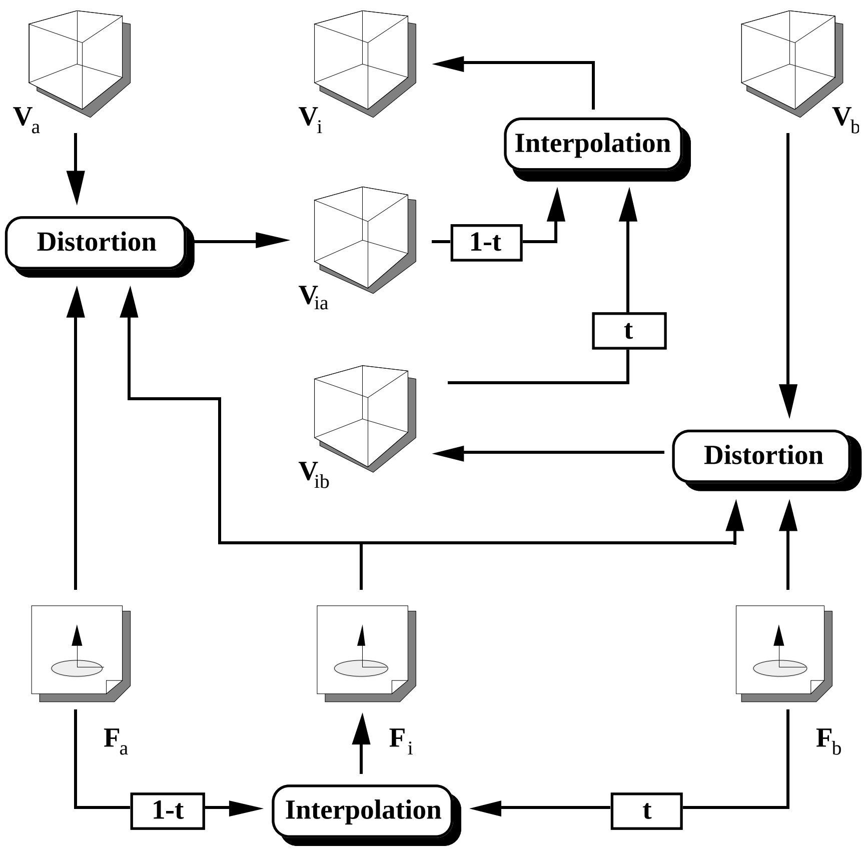

Figure 1: Feature-based volume morphing.

3. DISK FIELDS

3.1. Coordinate Mapping with a Single Pair of Disk Fields

A disk field is specified by its centre point **c**, normal vector **N** and radial vector **R** in a Euclidean space $\mathcal{E}^3$, where **R** is perpendicular to **N**. Any arbitrary point **v** defined in $\mathcal{E}^3$ may therefore be represented by cylindrical coordinates (ζ, ρ, φ) relative to the disk field (**c**, **N**, **R**), as illustrated in Figure 2. Point **u** is obtained by projecting **v** onto the disk plane in a direction parallel to **N**.

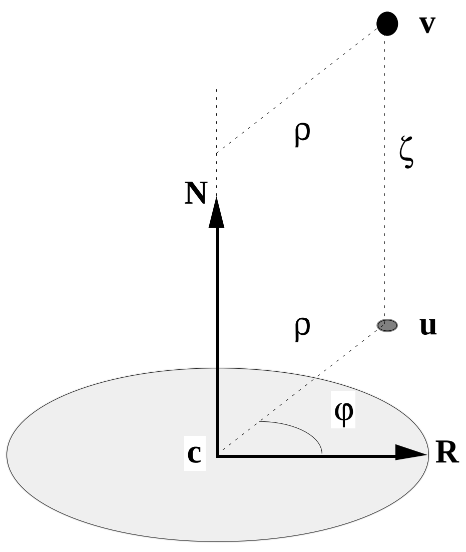

Figure 2: A cylindrical coordinate system defined by a disk.

A pair of corresponding disk fields ($\mathbf{c}_s$, $\mathbf{N}_s$, $\mathbf{R}_s$) and ($\mathbf{c}_d$, $\mathbf{N}_d$, $\mathbf{R}_d$) defines a coordinate mapping in $\mathcal{E}^3$ and can be used to control the distortion from a source volume $\mathbf{V}_s$ to a destination volume $\mathbf{V}_d$. To determine the value of a voxel $\mathbf{v}_d \in \mathbf{V}_d$, we first transform $\mathbf{v}_d$ to its local cylindrical coordinates (ζ, ρ, φ) which are defined as

$$\zeta = \frac{\|\mathbf{v}_d - \mathbf{u}_d\|}{\|\mathbf{N}_d\|} = \frac{(\mathbf{v}_d - \mathbf{c}_d) \bullet \mathbf{N}_d}{\|\mathbf{N}_d\|^2}, \tag{1}$$

$$\rho = \frac{\|\mathbf{u}_d - \mathbf{c}_d\|}{\|\mathbf{R}_d\|} = \frac{\|(\mathbf{v}_d - \zeta\,\mathbf{N}_d) - \mathbf{c}_d\|}{\|\mathbf{R}_d\|}, \tag{2}$$

and

$$\begin{aligned} \cos\varphi &= \frac{(\mathbf{u}_d - \mathbf{c}_d) \bullet \mathbf{R}_d}{\|\mathbf{u}_d - \mathbf{c}_d\|\,\|\mathbf{R}_d\|} = \frac{(\mathbf{u}_d - \mathbf{c}_d) \bullet \mathbf{R}_d}{\rho\,\|\mathbf{R}_d\|^2}, \\ \sin\varphi &= \operatorname{sign}\left[(\mathbf{N}_d \times \mathbf{R}_d) \bullet (\mathbf{u}_d - \mathbf{c}_d)\right] \sqrt{1 - \cos^2\varphi}. \end{aligned} \tag{3}$$

A corresponding point $\mathbf{v}_s$ is then obtained by transforming (ζ, ρ, φ) back to Euclidean coordinates as

$$\mathbf{v}_s = \mathbf{u}_s + \zeta\,\mathbf{N}_s = \left(\mathbf{c}_s + \rho\cos\varphi\,\mathbf{R}_s + \rho\sin\varphi \frac{\|\mathbf{R}_s\|}{\|\mathbf{N}_s \times \mathbf{R}_s\|} (\mathbf{N}_s \times \mathbf{R}_s) \right) + \zeta\,\mathbf{N}_s. \tag{4}$$

In equations (1), (2), (3) and (4), $\mathbf{u}_s$ and $\mathbf{u}_d$ denote, respectively, the projection of $\mathbf{v}_s$ and $\mathbf{v}_d$ in the related disk planes. Note that $\mathbf{v}_s$ is given in real coordinates and can possibly be located outside of $\mathbf{V}_s$. The value of $\mathbf{v}_d$ is normally determined by identifying a voxel $\mathbf{v}'_s \in \mathbf{V}_s$ that is the nearest to $\mathbf{v}_s$, though a tri-linear interpolation of the values of the eight neighbouring voxels is sometimes more appropriate in volume morphing. Figure 3 shows the rotation of a volume under the control of a single pair of disk fields. However, multiple pairs of disk fields are necessary for specifying a more complex transformation.

3.2. Volume Distortion with Multiple Pairs of Disk Fields

Given two sets of control fields $\mathbf{F}_s=\{\mathbf{f}_{s,1}, \mathbf{f}_{s,2}, \ldots, \mathbf{f}_{s,m}\}$ and $\mathbf{F}_d=\{\mathbf{f}_{d,1}, \mathbf{f}_{d,2}, \ldots, \mathbf{f}_{d,m}\}$ and a source volume $\mathbf{V}_s$, each voxel $\mathbf{v}_d$ in a destination volume $\mathbf{V}_d$ is mapped onto a set of points $\{\mathbf{v}_{s,1}, \mathbf{v}_{s,2}, \ldots, \mathbf{v}_{s,m}\}$. A point $\mathbf{v}_s$ is then obtained as a weighted average of $\mathbf{v}_{s,1}, \mathbf{v}_{s,2}, \ldots, \mathbf{v}_{s,m}$. We use the following equation, which is a modification of the one proposed in [1], for calculating the weight of a disk field ($\mathbf{c}$, $\mathbf{N}$, $\mathbf{R}$) at a voxel given in its local cylindrical coordinates (ζ, ρ, φ):

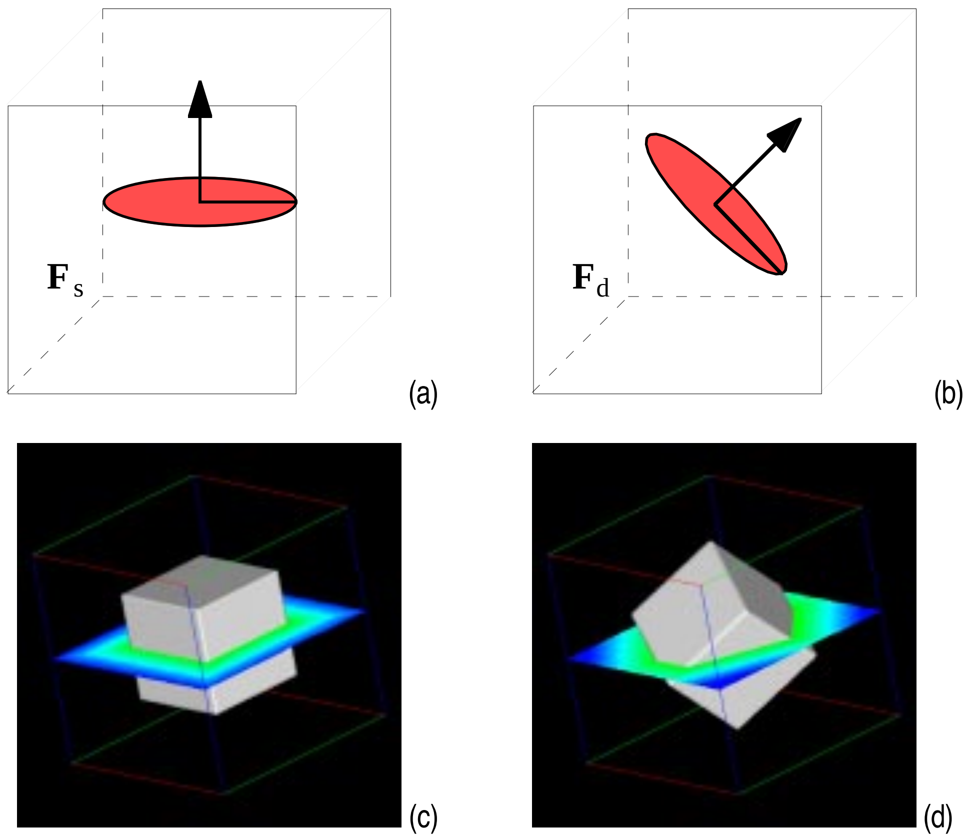

Figure 3: Volume distortion using a single pair of disk fields: (a) the source disk field; (b) the destination disk field; (c) the source volume; (d) the constructed destination volume.

$$\text{distance} = \begin{cases} |\zeta|\ \|\mathbf{N}\| & \rho \le 1, \\ \sqrt{\zeta^2 \|\mathbf{N}\|^2 + (\rho - 1)^2 \|\mathbf{R}\|^2} & \rho > 1; \end{cases} \qquad (5)$$

$$\text{weight} = \left[\frac{\|\mathbf{N}\|^{\boldsymbol{c}}}{\boldsymbol{a} + \boldsymbol{b}\ \text{distance}} \right]^{\boldsymbol{d}}.$$

where ***a***, ***b***, ***c*** and ***d*** are three constants whose values are assigned by users for controlling the relative effect of disk fields and are in the ranges of (0,), [0,), [0, 1] and [0,) respectively. The main use of the constant ***a*** is to keep the denominator from becoming zero. When ***b*** is equal to or barely greater than zero, the weight of a disk field at different voxels is hardly differentiated by the distances from the disk to these voxels. The greater the value of ***b***, the more differentiation between different distances. The constant ***c*** moderates the strength of disk fields which is proportional to the length of their normals. When ***c*** is zero, all disk fields have the same strength regardless of the length of their normals. The constant ***d*** is used to control the relative influence which each disk field has upon a voxel. When ***d*** is of a large value, the voxel is likely to be affected only by the strong and nearby disk fields. With the same source volume as in Figure 3, Figure 4 shows the distortion of a volume under the control of two pairs of disk fields and demonstrates effects of varying constants ***b***, ***c*** and ***d***.

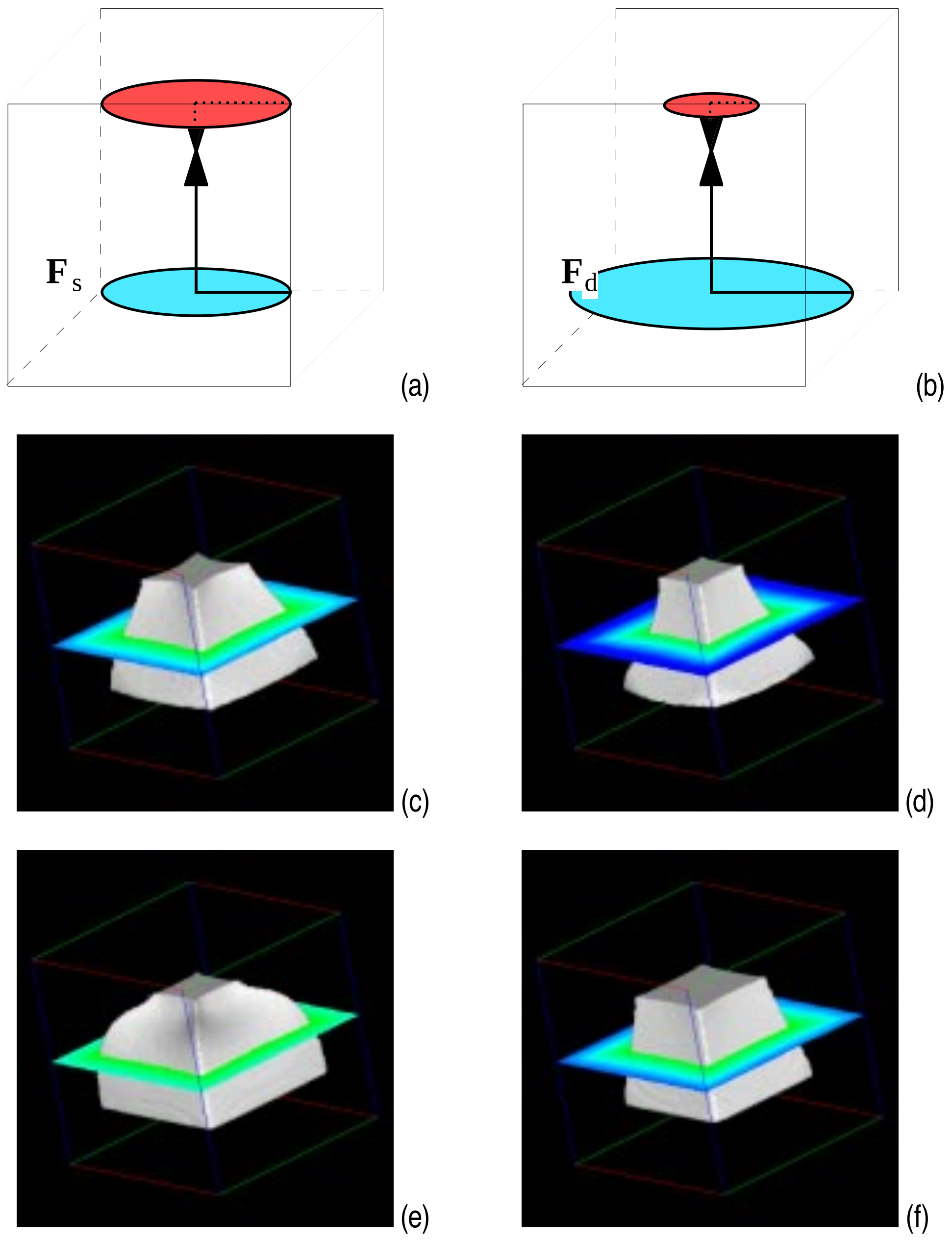

Figure 4: Volume distortion using two pairs of disk fields: (a) the source disk fields; (b) the destination disk fields; (c) the destination volume constructed with [***a***, ***b***, ***c***, ***d***]=[1, 1, 1, 1]; (d) [***a***, ***b***, ***c***, ***d***]=[1, 0, 1, 1]; (e) [***a***, ***b***, ***c***, ***d***]=[1, 1, 0, 1]; (f) [***a***, ***b***, ***c***, ***d***]=[1, 1, 1, 8].

3.3. Disk Specification and Interpolation

The design and development of a sophisticated user interface is a challenge facing almost every application that requires 3D input. It is unlikely that such a user interface is completely avoidable in volume distortion and morphing, especially when a volume contains many objects or complex transformation are involved. In general, we are not being entangled by a dilemma as to whether to have an intelligent and automated correspondence method or a sophisticated user interface, but rather facing the challenge to develop the best methods which are consistent, effective, easy to use and fast to compute; and upon which a user interface can be built.

Among all geometric objects that have been considered, such as point, line and box, disk fields are the most appropriate for a user interface. The visual representation of a disk field encapsulates more 3D information than other objects, and does not show any ambiguity. On the other hand, vectors in a point field or a line field would have to be labelled for identification, and their directions to be indicated by depth cueing or varying object sizes. 3D objects such as a box or hexahedron, which could be used to specify a local Cartesian coordinate system, would have much more faces and edges than a disk filed. With a few disk fields, the transformation of a volume can be effectively controlled. Table 1 lists a set of most commonly used transformations, which can be used as the constructs for defining complex transformations.

Table 1: Disk specifications for commonly used transformations.

Transformation	Disk Specification
translation:	move $\mathbf{c}$
1D scaling (in the direction of $\mathbf{N}$ and $-\mathbf{N}$):	resize $\lvert\mathbf{N}\rvert$
2D scaling (in all directions perpendicular to $\mathbf{N}$):	resize $\lvert\mathbf{R}\rvert$
3D scaling:	resize both $\lvert\mathbf{N}\rvert$ and $\lvert\mathbf{R}\rvert$
rotation about a line (defined by $\mathbf{N}$):	rotate $\mathbf{R}$
arbitrary rotation about a point (defined by $\mathbf{c}$):	rotate $\mathbf{N}$ and then recompute $\mathbf{R}$
flipping (in the direction of $\mathbf{N}$):	change the sign of $\mathbf{N}$
skewing (using two disks ($\mathbf{c}_1$, $\mathbf{N}_1$, $\mathbf{R}_1$) and ($\mathbf{c}_2$, $\mathbf{N}_2$, $\mathbf{R}_{12}$) in the direction of $\mathbf{R}_2$):	place $\mathbf{c}_1$ below the object and $\mathbf{c}_2$ above the object, and then move $\mathbf{c}_2$ in the direction of $\mathbf{R}_2$

In practice, it may not be easy for users to ensure that the radial vector $\mathbf{R}$ of each disk field is perpendicular to the normal $\mathbf{N}$ of the disk; and it is helpful for an implementation of the algorithm not to impose this restriction upon users' input, but allow the use of an arbitrary vector $\mathbf{P}$, as far as $\mathbf{P}$ is not parallel to $\mathbf{N}$.

Given the centre point $\mathbf{c}$ of a disk, $\mathbf{N}$ can also be initialised (or set by default) to the normal of the iso-surface at $\mathbf{c}$ by computing a gradient vector using the neighbouring voxels of $\mathbf{c}$ [11]. This further reduces the users' tasks in the specification stage.

To obtain an inbetween disk field ($\mathbf{c}_i$, $\mathbf{N}_i$, $\mathbf{R}_i$) from a pair of starting and finishing disk fields, ($\mathbf{c}_a$, $\mathbf{N}_a$, $\mathbf{R}_a$) and ($\mathbf{c}_b$, $\mathbf{N}_b$, $\mathbf{R}_b$), it is desirable to determine $\mathbf{N}_i$ by linearly interpolating the length and angles, instead of the coordinates, of $\mathbf{N}_a$ and $\mathbf{N}_b$. When the angle between the two vectors is 180˚, $\mathbf{R}_a$ or $\mathbf{R}_b$ can be used to determine the path of rotation. The inbetween radial vector $\mathbf{R}_i$ can be obtained in a similar manner.

4. Results and Remarks

Figure 5 shows the intermediate and final results of a morphing between two volumes, which contain a cube and a sphere respectively. Figure 6 gives a sequence of transformation from the sphere volume to a volume containing a CT (Computed Tomography) scan of a head. Fourteen pairs of disk fields were used to control the cube-to-sphere morphing process, and as shown in Figure 7, the same number of disks were used in the sphere-to-head morphing. The numbers of disk fields used in both cases are considered to be quite small, and the results are compared favourably with those produced by the existing algorithms [4].

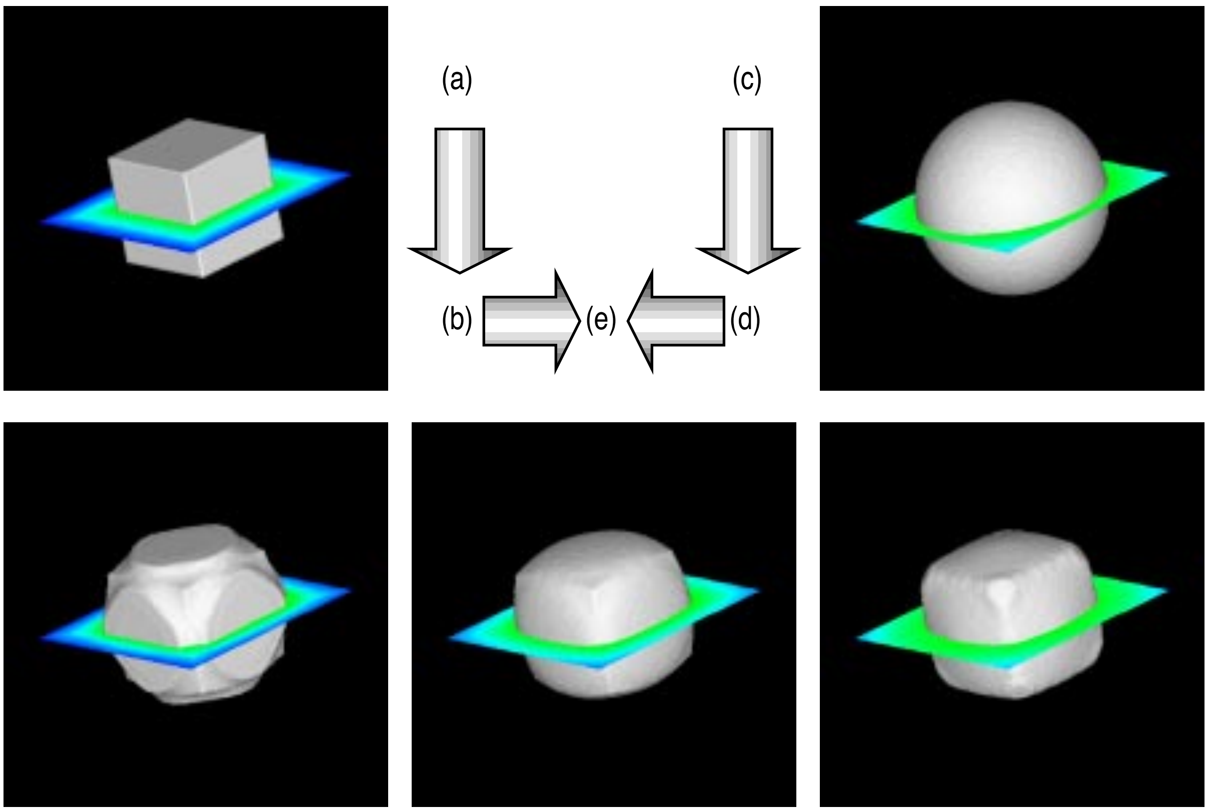

Figure 5: A morphing between two volumes: (a) the starting volume; (b) the distorted starting volume at t=0.5; (c) the finishing volume; (d) the distorted finishing volume at t=0.5; (e) the inbetween volume at t=0.5.

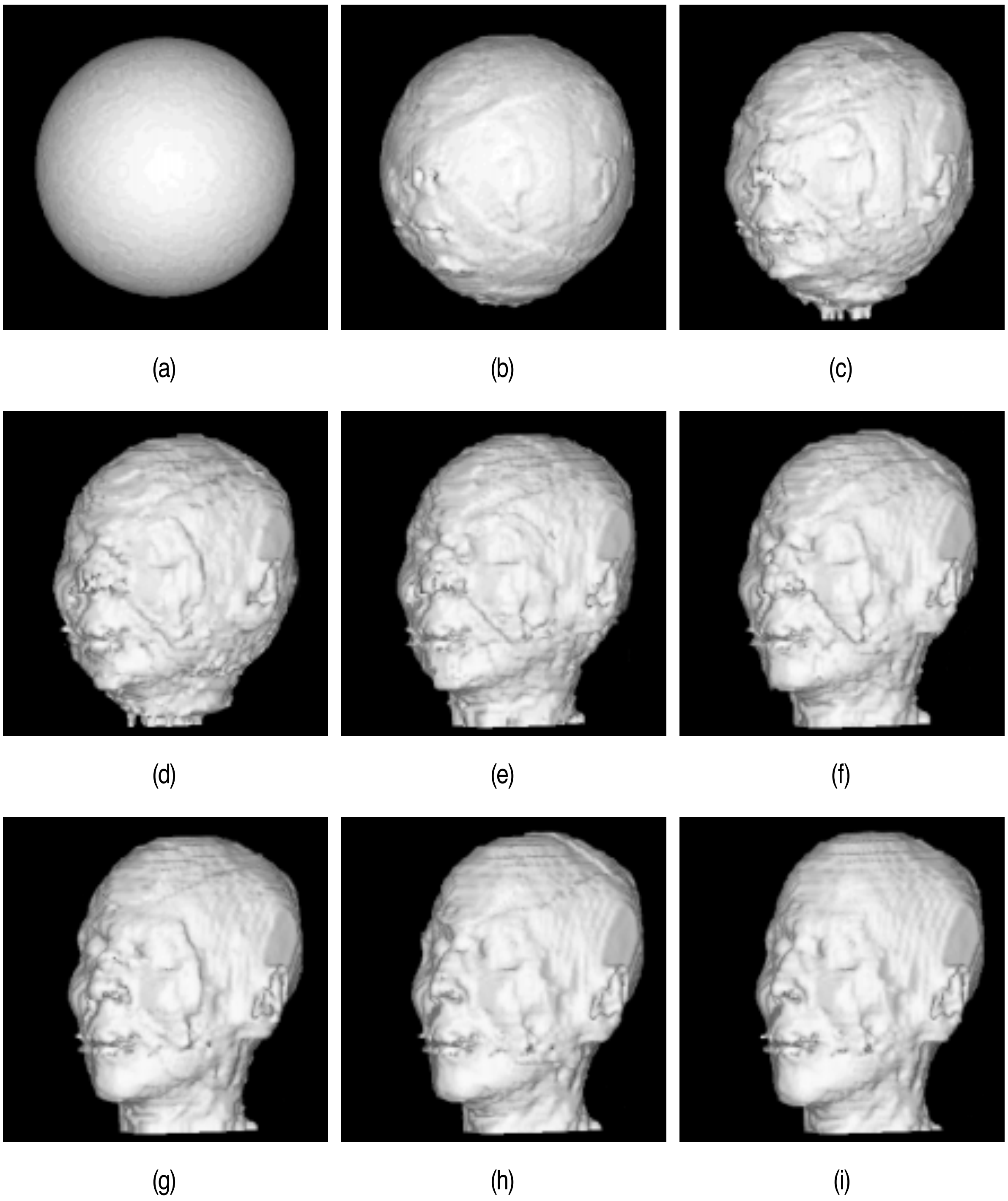

Figure 6: A morphing sequence: (a) the starting volume; (b-h) the inbetween volume at ***t***=1/8, 2/8, 3/8, 4/8, 5/8, 6/8, 7/8 respectively; (i) the finishing volume.

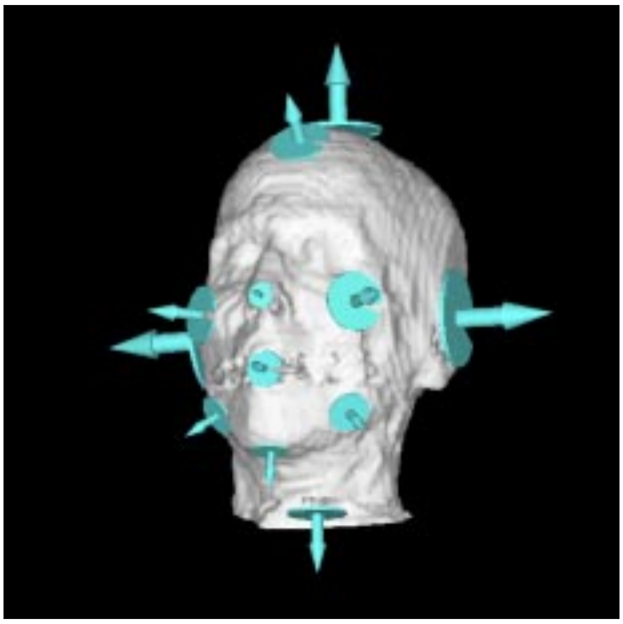

Figure 7: The sphere-to-head morphing was controlled by fourteen disk fields including the two on the nape and the back of the head.

For each inbetween volume, the algorithm requires $O(\boldsymbol{n} \cdot \boldsymbol{m})$ run time and $O(\boldsymbol{n}+\boldsymbol{m})$ space, where $\boldsymbol{n}$ is the size of the volume and $\boldsymbol{m}$ is the number of pairs of disk fields. In most cases, $\boldsymbol{m}$ is a much smaller number than $\boldsymbol{n}$. On the contrary, a mesh warping algorithm may require only $O(\boldsymbol{n}+\boldsymbol{m})$ run time, but would have to employ a large number ($\boldsymbol{m}$) of elements to achieve a reasonable result. For example, to generate a morphing sequence similar to the one in Figure 5, the tetrahedral warping algorithm would require 96 tetrahedra which are resulted from a triangulation with 29 points (the central points of the 14 disk fields, 14 boundary points and the centre point of the volume), but produce results in a relatively lower quality [4]. The frequence-based methods would require $O(\boldsymbol{n} \cdot \log \boldsymbol{n})$ run-time [12, 6], and therefore a small $\boldsymbol{m}$ is compared favourably with log $\boldsymbol{n}$.

The disk field morphing algorithm has been implemented in C as one of the tools in a volume manipulation toolbox. It also operates as a distortion tool when there is only one input volume (either $\mathbf{V}_a$ or $\mathbf{V}_b$). The algorithm has been tested against a variety of volumes and disk fields. Some of the run times in seconds obtained during the testing are given in Table 2, where each test is a complete morphing process applied to a single inbetween volume. As shown in Figure 8, the execution times of the algorithm match with its time complexity. The timing of the algorithm was carried out on a DEC Alpha workstation, and volumes were visualised using Advanced Visualisation System (AVS).

Table 2: The run-times (sec.) of the algorithm.

Fields	**Volume Size — n**			
m	**128x128x32**	**128x128x64**	**128x128x96**	**128x128x128**
2	85.53	167.78	251.78	334.88
6	210.97	421.22	631.28	842.22
10	333.33	665.88	1,005.08	1,332.15
14	471.58	934.03	1,411.10	1,865.53

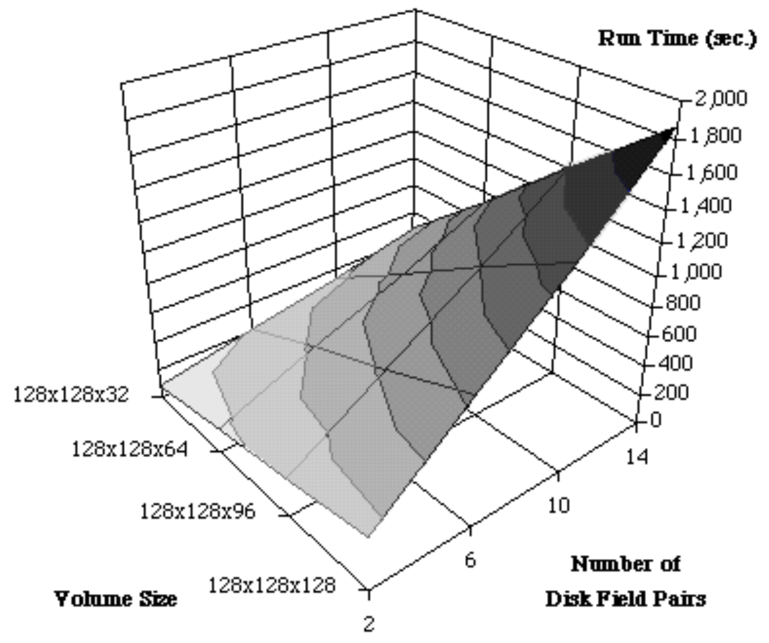

Figure 8: A surface ***F(n, m)*** representing the run times given in Table 2.

The main concern of a field morphing method would be that each voxel has to be computed against every field regardless of the amount of influence that it receives from the field [6]. By associating each field with an appropriate bounding volume, for example a cylindrical or spherical volume in the case of a disk field, the speed of a field morphing process can be improved, though it must be ensured that every voxel is included in at least one bounding volume. Alternatively a threshold ε can be used eliminate some coordinate transformation when $\frac{\text{(the weight of a field at } \mathbf{v})}{\text{(the current total weight at } \mathbf{v})} < \varepsilon.$

In general, the method, which can be used for both volume distortion and morphing, produces good quality results at a reasonable speed without demanding complex control information. Its flexibility and controllability are apparent, and it can be applied to the general form of volumetric data, which is a collection of scattered voxels each associated with a set of values. In comparison with other field morphing and mesh warping methods, disk fields are relatively easier to specify and manipulate in an interactive environment. The method, together with an appropriate graphical user interface, can be developed into a useful tool for many applications where geometric transformation of volumetric data are necessary. The work of parallelising the algorithm is currently being undertaken at Swansea. A graphical user interface has also been designed based on the disk field concept, and its development is being planned.

REFERENCES

[1] T. Beier and S. Neely, Feature-based image metamorphosis, *SIGGRAPH Computer Graphics*, **26**(2), 35-42 (1992).

[2] G. Woldberg, *Digital Image Warping*, IEEE Computer Society Press (1990).

[3] A. Kaufman, D. Cohen and R. Yagel, Volume Graphics, *IEEE Computer* **26**(7), 51-64, (1993).

[4] M. Chen, M. W. Jones and P. Townsend, Methods for Volume Metamorphosis", In *Image Processing for Broadcast and Video Production*, Y. Paker and S. Wilbur (Eds.), Springer-Verlag, London (1995).

[5] B. A. Payne and A. W. Toga, Distance Field Manipulation of Surface Models, *IEEE Computer Graphics & Applications* **12**(1), 65-71 (1992).

[6] J. F. Hughes, Scheduled Fourier volume morphing, *SIGGRAPH Computer Graphics* **26**(2), 43-45 (1992).

[7] T. He, S. Wang and A. Kaufman, Wavelet-based volume morphing, In *Proceedings of IEEE Visualization' 94*, Washington, D.C., 85-92 (1994).

[8] D. Shepard, A two-dimensional interpolation function for irregularly spaced data, In *Proceedings of 23rd National Conference of ACM*, 517-524 (1968).

[9] D. Ruprecht, R. Nagel and H. Müller, *Spatial free form deformation with scattered data interpolation methods*. Research Report: 539, Department of Computer Science, University of Dortmund, Germany (1994).

[10] A. Lerios, C. D. Garfinkle and M. Levoy, *Feature-based volume metamorphosis*, Proc. SIGGRAPH 95, Los Angeles, California, August 6-11, 1995, 449-456 (1995).

[11] M. Levoy, Display of surfaces from volume data, *IEEE Computer Graphics & Applications* **8**(3), 29-37 (1988).

[12] T. H. Cormen, C. E. Leiserson. and R. L. Rivest, *Introduction to Algorithms*, The MIT Press, Cambridge, Massachusetts (1990).