Volume Sculpting and Keyframe Animation System

V. Chandru,* N. Mahesh, † M. Manivannan,‡ S. Manohar,§

Abstract

In traditional animation, keyframes are modeled and standard graphics pipeline is used to animate the scene. In this paper we consider volume animation where the 3D world and its components are represented as a voxel model. The voxel model is annotated with hierarchical feature information and thus facilitates easy volume editing. These annotations are based on regularised Minkowski operators *used in constructing/sculpting the model. The Selective editing of volume models also becomes possible with these annotations. The pure voxel or volumetric approach used from start to finish has the advantage that extremely simple data structures and algorithms enable us to generate complex volume animations.*

1 Introduction

Computer animation, in recent years, has moved beyond the entertainment industry to become a valuable tool in several areas of science and engineering. A parallel development is the emergence of *volume graphics* [7] as a viable alternative to traditional *polygonal graphics*. The cross fertilization of these two areas is the area of volume animation or voxel-based animation. Keyframe animation, the process by which critical frames in an animation sequence are individually crafted by a skilled animator, with all the intermediate frames (the inbetween frames) generated by simple interpolation (either manual or automated), is still the dominant animation paradigm. In this paper we present a voxel-based keyframe animation system, a first of its kind, that allows the animator considerable flexibility in designing the animation.

*Computer Science and Automation, Indian Institute of Science,Bangalore - 560012, India, Email:chandru@seas.upenn.edu

†Sanchez Software Systems, Philadelphia, PA, USA, Email:mahesh@sanchez.com

‡Visiting Scientist, National Institute of Standards and Technology(NIST), Gaithersburg, MD, USA. Email:mmani@nist.gov

§Computer Science and Automation, Indian Institute of Science,Bangalore - 560012, India, Email:manohar@csa.iisc.ernet in

1.1 Animation of Voxel data

The increased availability of volume data from many sources and the diverse applications involving volume data are the reasons for recent developments in volume animation. Different scanners are used to produce *Scanned Volumes* of live patient to study, for example, the dynamics of heart. Any functions over 3D space can be used to generate *procedurally defined* volume models. The volume models in scientific visualization are *computed volume data* such as weather modeling and prediction which are generated by compiling different source of data into 3D grid. *Voxelized geometric models* have emerged in recent years [16] for converting polygonal models into a voxel volume. The quality of such voxelizations can be improved by antialiasing techniques [13]. A naive approach to animating such models is to animate the geometric models and after each time step, voxelize and then volume render the resulting volume. Such an approach can be modified and can indeed be a useful animation method as will be shown later in this paper.

1.2 Related Work

An intuitive method of volume animation is introduced by Nikhil et al. [12] using skeletons. The skeleton which is extracted from the given volume is animated and the volume is reconstructed over the deformed skeleton. A physics based approach, a mass-spring model is used for voxel-based animation by Chen et al. [3]. Muscle volume deformation is done using bio-mechanical FEM model in a voxel based system [18]. Volume metamorphosis considered by Lerios et al. [8] is feature based and allows fine user control and thus ensures realistic looking intermediate volumes. The user can quickly define how an intermediate volume should look like. Young Man project by Prakash et al. [17] involves combining voxel model of a human with geometric models(surface) in a motion authorship system. Different parts of the human body are identified by clustering into boxes of voxels and the boxes are transformed/deformed to give motion or animation effect and then the original volume buffer is voxture mapped into the deformed boxes. Gibson et al. [5] used voxel objects for

1087-4844/00 $10.00

simulating soft tissues. Their work is based on the simple observation that complex system behavior can result when each of large number of elements follow a simple behavior pattern.

In this paper we restrict attention to animations that are created frame by frame. One of the main reasons is that volume rendering itself (for reasonable voxel dimensions) is non-interactive on standard platforms. Thus when we refer to interactivity in creating animations we refer to the interactive modification of one frame of the animation. We find that the current volume animation methods do not support intuitive ways of creating a simple animation, notion of features of an object, or a hierarchy of components within an object, direct control of the animation (for example, physical based systems take away the direct control of the animation from the animator), traditional kinematics and inverse kinematics tools prevalent in current polygon-based 3D, and mostly they require heavy preprocessing which reduces the interactivity of the animation process. The method proposed in this paper attempts to overcome several of the above limitations.

In Section 2, we present the motivation for developing voxel-based animation. In Section 3, the notion of voxel features, the central idea in our voxel-based animation system is described. The algorithms and data-structures for implementing features in voxel models are also discussed in this section. Our prototype voxel-based keyframe animation system is presented in Section 4. The potential of our approach is demonstrated in Section 5 by means of an animation sequence designed using our system. The last section outlines future research direction.

2 Voxel-based keyframe animation

In an animation system the two key requirements namely, design and generation of keyframes of a dynamic scene and interpolation between keyframes should be satisfied. In volume graphics the keyframe would be volume keyframe, or a voxel keyframe.

Typically, the voxel-based animation scenario involves the following aspects:

1. creation of keyframes, either from scratch or by modifying previous keyframes.
2. iterative and interactive modification and rendering of the keyframes till the animator is satisfied with the visual aspects of the frame
3. in-betweening: the process of starting with the sequence of keyframes and generating the required frames so that the entire animation sequence is smooth.

The creation of a voxel keyframe from scratch, or by the modification of an existing keyframe is nothing more than interactive voxel modeling. Voxel-based modeling has several advantages such as, support for heterogeneity, insensitivity to object complexity and topology, ease of implementation of Boolean operations, view independency etc. Voxel modeling has excellent local editability and certain global operations (analogous to filtering) that alter entire model can be implemented with ease. We can select individual voxel from a voxel model and edit it. Voxel models while enjoying these many virtues, lacks component information of an object, that is no information about the four legs of a table for example. This particular inability has hitherto restricted the use of voxel modeling as a major volume modeling paradigm. Volume modeling community has overlooked voxel models because of this particular in-ability.

Central idea of this paper is the notion of *voxel features* that overcomes the existing drawbacks of voxel based modeling and enables us to meet the three requirements of a voxel based keyframe animation system listed above. We first outline our notion of voxel features and then proceed to describe how this can be incorporated into a keyframe animation system.

3 Features in Voxels

By features we mean any *collection of voxels that has some specific meaning to the modeler.* For example, the voxels that define the eyes of a head being sculpted or the voxels that describe a sphere are deemed to be features. This idea is more general than the traditional notion of features held in the CAD community [11]. When features are represented in voxel models, complex (free form) features can be represented which is indeed necessary for animation. Feature related algorithms (like feature-feature interaction) can be handled with ease and heterogeneous features (composites) can be easily represented [2]. When voxel models are annotated with feature notions editability improves and hierarchical nature of objects and object oriented advantages can be combined with voxel models.

Our approach to voxel features

Our approach to incorporating features in voxel models is by [2] storing the sequence of sculpting operations along with the voxel data, and defining *feature operators* which will operate on features. By *"storing sequence of operations"* we mean that along with voxel models we store the sequence of operations that created the voxel model. This is based on the observation that the model is the result of sequence of sculpting operations. This problem is studied by many researchers [14, 6, 11]. Following is the excerpt from [14].

> *"...unless all original models and the sequence of operations are stored then any modeling scheme*

with some sophistication in its modeling space would be lossy..."

The result of each sculpting operation changes a set of voxels. This group of voxels we call as a feature. Instead of forcing a voxel model to store all the feature information as reported in the voxel literature, we introduce the notion of *feature operators* which will take care of the presence of features and interactions between features. This reduces the memory burden and complexity of the algorithms which otherwise would have been very high. We have demonstrated these ideas in our prototype system Sirpi described below.

4 Sirpi

Sirpi[1] (Sculpting Interface for Rapid PrototypIng) is an implementation of our interactive sculpting system. It uses *Interactive Virtual Machining* (IVM) as an interface to interactive sculpting. Sirpi is based on voxel modeling approach [9]. Our work in interactive sculpting is in the context of a geometric modeling framework for rapid prototyping(RP). Interactive sculpting is the process by which a designer makes free-form changes on a volume model interactively [15, 9]. It is the extension of 2D paintbrush programs to 3D. IVM is a conceptual interface to sculpting and our unique approach to interactive sculpting. Simple milling, contour milling, turning, thread cutting and surface machining are the IVM tools Sirpi implements. Sirpi supports four primitive tool shapes: Cuboid, Cylindrical, Conical, and Spherical. The tool also is a voxel model, and Sirpi sculpted models itself can be used as a tool, thus expanding Sirpi's design space.

4.1 Minkowski Operators

Minkowski operators are shape operators using which we can design almost all real-world objects [4]. IVM is implemented using Minkowski operators for cutting (or pasting) clay along a path using a tool. A brief description of these operators and their role in Sirpi are given below.

Minkowski addition of two sets A and B in R^d is defined as the union of sets obtained by positioning one of them, say B, at every point of the other, say A and vice versa, i.e., the set of points obtained by vectorially adding each point in A with each point in B. Mathematically, if A_p denotes the translate of a set A by the vector p, i.e., A_p = A $\oplus$ {p}, then,

$$A \oplus B = B \oplus A = \bigcup_{a \in A} B_a = \bigcup_{b \in B} A_b$$

which is same as,

$$A \oplus B = \{a + b : a \in A, b \in B\}$$

where $\oplus$ stands for Minkowski addition.

Minkowski decomposition of two sets A and B in R^d is defined as

$$A \ominus B = \bigcap_{b \in B} A_{-b} = \bigcap_{b \in B'} A_b$$

where $\ominus$ stands for Minkowski decomposition operation. The set B' = { -b : b $\in$ B } is generally known as the symmetrical set of B with respect to the origin.

Computation of Minkowski operations when both the objects are general polyheadra is hard. Implementing Minkowski operators in B-Rep restricts the two object to be polyhedrons. The problem of finding algorithms for multiply connected B-Rep objects is still open. We avoid these problems by taking a voxel representation for both the tool and the clay. Implementing Minkowski operators in voxel representation has special advantages such as topology insensitivity, trivial algorithms for Boolean operations etc. [16]. Different algorithms for implementing Minkowski operators in voxel representation are detailed in [15].

Cutting and pasting operations are implemented using Minkowski sum of the tool along its trajectory. We approximate the trajectory by a polyline. Hence, the task reduces to finding the Minkowski sum of the tool with a straight line segment. Let S be the volume swept by the tool along the straight line segment, and C be the clay. Then, in set theoretic terms,

$$Result = \begin{cases} C - S & \text{for cut operation} \\ C \cup S & \text{for paste operation} \end{cases} \quad (1)$$

The following lemma gives an algorithm to calculate the swept volume by considering boundary voxels that lie along the sweep line alone.

$$A \oplus L = (A_s \bigcup (\cup_{v \in \delta A} v \oplus L)) \quad (2)$$

where L sweep line is the line segment with end points s and t, A is the tool, A_s is the tool placed at s and δA is the boundary voxels visible to the sweep line. Since number of voxels in δA is $O(N^2)$, the time complexity of the algorithm is $O(N^3)$ where N is the size of the voxel array.

4.2 Feature based Sirpi

We consider the result of each Minkowski operation to be a voxel feature. Thus our notion of a feature is much more general than the traditional feature. For example, we consider the swept volume of a tool along a trajectory as a feature. Some of the examples are shown in the Figure

[1] Sirpi means sculptor in the Dravidian language Tamil

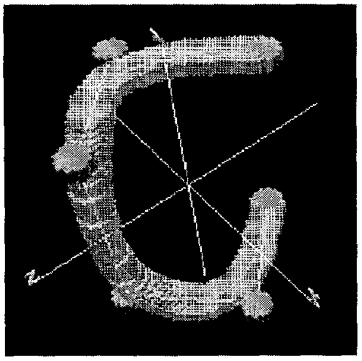

Figure 1. A contour feature

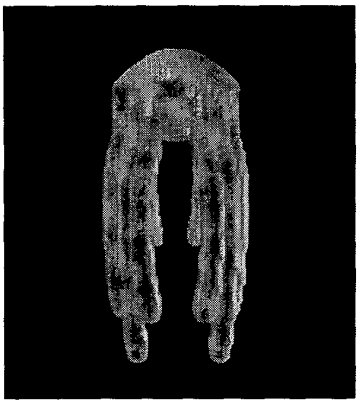

Figure 2. A sculpted jelly fish

1, 2. Figure 1 shows the swept volume of contour milling with a cylindrical tool. Figure 2 shows a tortoise sculpted in Sirpi. The body of the tortoise is simple milled using an ellipsoid shaped tool scan-converted outside Sirpi. Similarly, the nose of the tortoise is simple milled using a pyramid shaped tool scan-converted outside Sirpi. The legs are contour milled using a cuboidal tool. Thus, Sirpi allows objects modeled using its tools to be in turn loaded as tools and used for further sculpting. Minkowski operators can be used to describe traditional features.

We call a sequence of Minkowski operations used to create a voxel model as a Minkowski model. We see that the Minkowski model and a voxel model are complementary because, voxel model is a definite, explicit and memory intensive, whereas Minkowski model is procedural, implicit and concise [2]. So, we see advantage in storing the Minkowski model along with voxel model. For example, in the Minkowski model the BSpline curve is captured as the defining control polygon, then it is approximated to line segments for voxel models.

4.3 Regularised Minkowski Operators

We introduce a new notion of a regularised Minkowski operation as follows: Suppose there are n features in a voxel model and n+1th feature is the one we would like to edit, then,

$$\begin{aligned} f_{n+1} &= A \oplus L \\ r(f_{n+1}) &= A \otimes L \\ &= f_{n+1} - \bigcup_{i \in n} (f_i \cap f_{n+1}) \end{aligned} \tag{3}$$

where $\otimes$ is the regularised Minkowski operator and $r(f_{n+1})$ is the regularised $n + 1^{th}$ feature. The result of regularised Minkowski operators will be regularised features. A regularised feature is nothing but, the feature itself except the intersecting volumes of other features. Thus, when the regularised features are edited, other features in the model are not disturbed. Note that a similar regularised feature can be used in Minkowski decomposition. An example of regularised Minkowski addition is shown in the Figure 3.

The notion of regularised Minkowski operators is not new in the literature. Menon et al. [10] use regularised Minkowski operators in the same sense as used in regularised set operators for CSG modeling (ie., to remove dangling edges and faces). Our regularisation of Minkowski operators is with respect to features.

4.4 Algorithms for regularised Minkowski operators

The algorithms described in this section assume the voxel representation of the objects.

4.4.1 Naive approach

A naive algorithm would first create a feature using Minkowski operator, then compute its intersection with the existing features, and finally compute the set difference of the corresponding voxel sets. The pseudo code is given below.

```
RegularisedMSum(tool, toolPath, listOfFeatures)
{
 rMSum = RegularisedMSum(tool, toolPath, NULL);
 foreach(feature in listOfFeature)
 {
  rMSum = rMSum - intersection(rMSum, feature);
 }
}
```

Using an $O(N^3)$ algorithm for computing the Minkowski sum, this algorithm would take $O(rN^3)$ where r is the number of features.

4.4.2 The Reference Count approach

The naive approach can be improved to an $O(N^3)$ as described below. A counter called reference count is maintained for each voxel as shown in Fig.4. When a voxel is referred to by a feature its reference count is incremented. When the Minkowski operation is that of subtraction, the reference count is decremented. Thus,

regularised i^{th} feature =
all the voxels belong to i^{th} feature -
voxels having reference count more than one. (4)

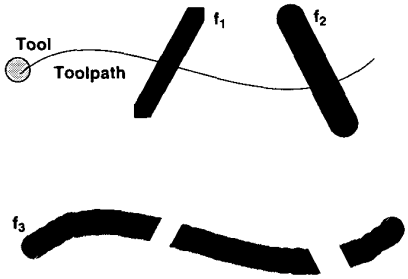

Figure 3. Regularised Minkowski Summation

A regularised Minkowski operator can now be implemented using the following procedures. Though the pseudo code is for binary volumes, the algorithm that has been implemented works for non-binary volumes as well. Note that the improvement in complexity has been achieved by avoiding the explicit computation of all pairwise intersections.

```
setVoxel(pt, state)
{if(state = ON)
  incrementRefCount();
 if(state = OFF)
  decrementRefCount();}
incrementRefCount(pt)
{refCount ++; }
decrementRefCount(pt)
{refCount --;
 if(refCount =< 0)
  { refCount = 0;
  setOff(pt);}
}
```

4.4.3 The Stack-of-Bits approach

In the Sirpi environment, the user can either start with the full clay (we call this approach *additive sculpting*), or with null clay (we call this approach *subtractive sculpting*). Reference count approach will not allow this freedom to the user, ie., additive and subtractive sculpting can not be mixed. The algorithm described in this section provides this flexibility. The approach is again simple. Each voxel has a stack, where bits (for binary voxels, for heterogeneous environment more bits per voxel has to be allocated) can be pushed or popped. Whenever there is request for change of the voxel state the voxel tries to store the old state in a *sequence of states* by pushing the old state in the stack. Unlike the reference count approach, this approach follows the *sequence* strictly. A change in sequence is simply not allowed, if allowed, the result would not be the regularised Minkowski summation or decomposition. Sirpi implementation of stack-of-bits approach supports two modes of sculpting feature mode and edit mode. In the feature mode, the voxel state is pushed onto the stack. In the edit mode the voxel stack is popped to bring the voxel to its preceding state. The modified outline of Sirpi is shown in Fig.5.

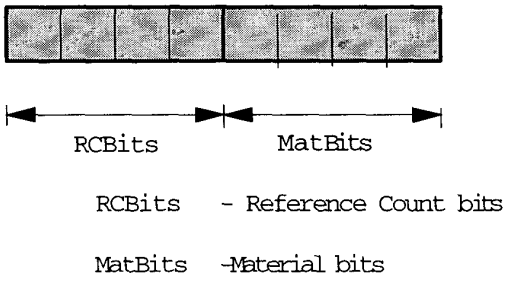

Figure 4. Byte di-section for reference counts

5 Volume animation

We demonstrate that Sirpi is capable of handling all the three steps involved in a voxel-based keyframe animation system (Section 2) by sculpting and animating a synthetic creature (looks like jelly fish) Fig.2. Each volume

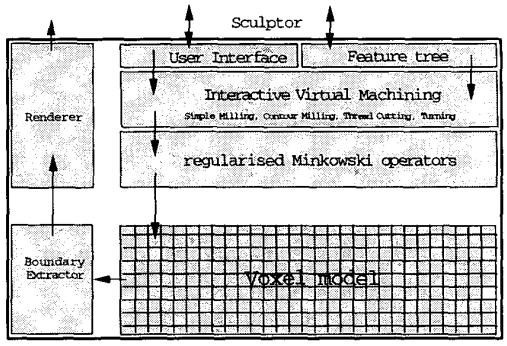

Figure 5. Sirpi architecture

keyframe is of 200x200x200 size. Only one frame (Frame1 shown in Fig.2 is sculpted. The other four frames are resculpted. For resculpting a feature it takes about 16 sec for a 200x200x200 size Sirpi model, in a SGI's Indigo system with a single R4400 CPU at 150 MHz clock speed. The images are rendered using VolVis [1]. The body and belly of the jelly fish are ellipsoids generated outside the Sirpi environment, imported, and simple milled using IVM. Each leg is sculpted using contour milling of our IVM system and represented as a feature which retains the control polygon of the contour in a hierarchical order. When we want to change the legs' contour the corresponding icon in the feature graph is chosen and the control polygon is changed appropriately. It is to be noted that while resculpting the legs, the features belonging to other features (in this case for example body of the creature) are not disturbed: regularised Minkowski operation ensures this. The hierarchical nature of feature graph takes care of regenerating all the child features of an edited feature. The inbetweens are obtained by interpolating the volume keyframes. For demonstration purpose, Sirpi implements linear interpolation of the volume keyframes. Interpolation is done between each sequence of the above five volume keyframes. The interpolation is linear interpolation of the control points of the defining BSpline curve (in case of contour milling) or end points of a straight line (in case of simple milling). This interpolation can be considered as *skeleton based parametric volume keyframe animation*, because in Sirpi, the skeletons are parametrized and interpolated.

The mpeg movie of the animation sequence shown in URL http://www.csa.iisc.ernet in/~manohar/jelly.mpg has 40 frames. **It is to be noted that, only one keyframe is sculpted, four keyframes are resculpted, and all other frames are interpolated**. Unlike volume metamorphosis discussed by Lerios et al. [8] our method outputs the volume keyframe, from which interpolation can be done offline and independently. But the notion of feature is richer than considered by the Leoris et al.

6 Conclusion

We have demonstrated a voxel-based animation system that can create volume keyframes either from scratch or by modifying previous keyframes that can modify the keyframes iteratively at the interactive rates and that can create a smooth animation sequence from a sequence of keyframes. The new notions of voxel features and regularised Minkowski operators enabled us to implement the above animation system. We note, however, that these two notions have significant impact much beyond animation systems. For instance voxel-features can be combined with constraints resulting in powerful constraint-based modeling paradigm. The exploitation of the power of these notions both for other traditional animation techniques such as constraint based inverse kinematics and dynamics, as well as for interactive sculpting are currently being explored.

References

[1] R. Avila and et. al. Volvis: A diversified volume visualization system. *IEEE Visualization Proceedings*, pages 31–38, October 1994.

[2] V. Chandru, N. Mahesh, M. Manivannan, and S. Manohar. Voxel features. *ASME Design Automation Conf*, Aug 1999.

[3] Y. Chen, Q. Zhu, and A. Kaufman. Physically based animation of volumetric objects. *Proc. of Comp.Animation conference*, May 1998.

[4] P. K. Ghosh. A mathematical model for shape description using minkowski operators. *CGIP*, 44:239–269, 1988.

[5] S. Gibson and et.al. Volumetric object modeling for surgical simulations. *Intl. Journal of Medical Image Analysis*, Spring/Summer 1998.

[6] C. Hoffmann. Editable representation. *Tech. Report, Dept. of Computer Science, Purdue University*, 1993.

[7] A. Kaufman, D. Cohen, and R. Yagel. Volume graphics. *IEEE Computer*, 26(7):51–64, 1993.

[8] A. Lerios, D. G. Chase, and M. Levoy. Feature-based volume metamorphosis. *Proc. SIGGRAPH*, pages 449–456, 1995.

[9] N. Mahesh. Sirpi: Interactive sculpting system. *Master thesis: Indian Institute of science, India*, 1998.

[10] J. Manon, R. J. Marisa, and J. Zagajac. More powerful solid modeling through ray representation. *IEEE CG&A*, pages 22–35, May 1994.

[11] M. Mantyla, N. Dana, and J. Shah. Challenges in feature-based manufacturing research. *Comm. of ACM*, 39:77–85, February 1996.

[12] G. Nikhil, D. Kenchammana-Hosekote, and D. Silver. Volume animation using the skeleton tree. *Proc. IEEE Symposium on Volume visualization*, pages 47–53, October 1998.

[13] C. E. Prakash and S. Manohar. Error measures and 3d antialiasing for voxel data. *Proc. pacific graphics*, 1995.

[14] A. Rappoport. Geometric modeling: a new fundamental framework and its practical implications. *Proc. Solid Modeling*, pages 31–41, 1995.

[15] S. U. Sethia and S. Manohar. Minkowski operators for voxel based sculpting. *Comp. and Graphics*, 22(5):593–600, 1998.

[16] S. W. Wang and A. Kaufman. Volume sample voxelization of geometric primitives. *Proc. of IEEE visualization*, 1993.

[17] W. Zhongk and C. E. Prakash. Visible human walk: Bringing life back to the dead body. *Volume Symposium*, 1999.

[18] Q. Zhu, Y. Chen, and A. Kaufman. Real-time biomechanically-based muscle volume deformation using fem. *Proc. of Eurographics*, 1998.