CaveSculpture:
Creating sculpture from CavePaintings

Andrew W. McClain

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-03-27
May 2003

CaveSculpture:
Creating sculpture from CavePaintings

by
Andrew W. McClain
A.B., Brown University, 2003

A thesis submitted in partial fulfillment of the requirements for Honors
in the Department of Computer Science at Brown University

Providence, Rhode Island
May 2003

This thesis by Andrew McClain is accepted in its present form by
the Department of Computer Science as satisfying the research requirement
for the awarding of Honors.

Date ____________ ________________________________

David Laidlaw, Reader

Date ____________ ________________________________

Richard Fishman, Reader

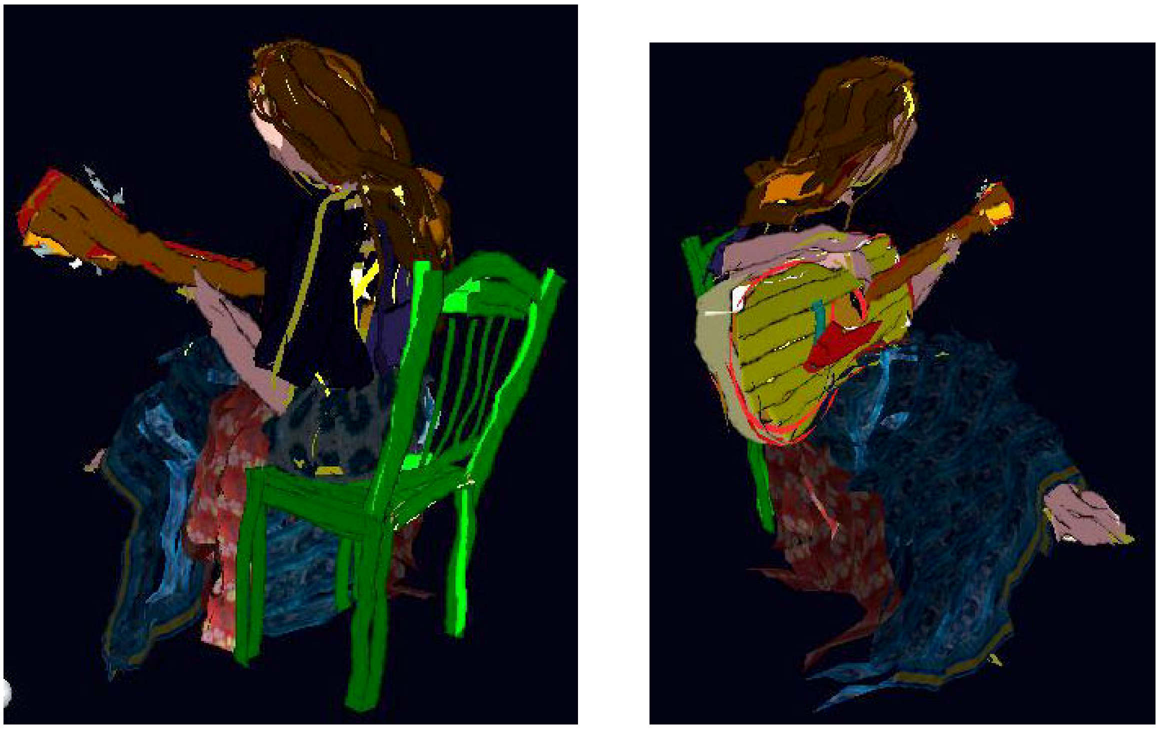
Acknowledgements

Thanks to Chris Bull and Daniel Acevedo for providing me the inspiration for this project, to David Laidlaw for his support, guidance, and insistence on "gifts", to Richard Fishman for placing me on a path that combines art and science, to Dan Keefe and Dave Karelitz for their gracious support with CavePainting, and finally, to Brian Corkum for always taking time to share his knowledge with me.

Table of Contents

Introduction

CavePainting was created as medium that takes the essence of two-dimensional painting and translates it into a three-dimensional virtual space (Keefe et al.[4]). However, there is no convenient method view to these paintings outside of a virtual environment. Virtual reality CAVEs are expensive and few in number and video does not capture the stereographic nature of the paintings. Translating CavePaintings into CaveSculptures through the use of a rapid prototyping machine allows the paintings to be viewed without special equipment. Furthermore, combining virtual reality with rapid prototyping creates a new medium that lies between painting and sculpture. This paper describes the process for creating sculptures from CavePaintings.



Figures 1 and 2. Views of a CavePainting, *La Guitarrista Gitana* by Daniel Keefe.

Previous Work

Many systems of virtual sculpture, such as Chen et al.[1] and McDonnell et al.[2], seek to simulate the sensations a physical sculptor would experience by involving a haptic response system that allows the user to engage physically with the sculpture being created. We do not aim to reproduce the act of sculpting in a virtual environment. Instead, we seek to reinterpret painting as sculpture. Rather than transforming a virtual sculpture into a physical sculpture, we intend to transform a virtual painting with sculptural attributes into physical sculpture with painterly attributes.

Description of process

The rapid prototyping machine used in the creation of CaveSculptures creates models out of plaster in an additive process. Consecutive layers of plaster are "printed" with colored binder. As the layers accumulate, the model is built. Once the model is created, excess plaster is removed, and the rendered sculpture is coated in wax.

There are limitations to this process, however. The model given to the machine must be closed and water-tight. The machine has a difficult time prototyping models with co-planar surfaces, self-intersections, or non-regular surface normals. Additionally, many rapid prototyping machines cannot render models with multiple, intersecting pieces. Physically, the model must be thick enough to withstand handling. Due to the fragility of plaster, models that are very thin tend to disintegrate during the post-build process.

CavePaintings consist of a collection of "strokes" in 3D space, most often ribbons and tubes. Each stroke is stored as a polygonal mesh that has no thickness. In order to create a single, watertight mesh that is hardy enough to withstand the fabrication process, we must thicken and join all the strokes in the painting, without losing the painterly quality of the strokes.

"Thickening" and "Joining"

Distance fields provide an elegant solution to both "thickening" and "joining" the strokes. A distance field is generated by sampling the distance to the closest point on the mesh at regular intervals within a volume surrounding the mesh, creating a grid of distance values. Polygonal meshes can be combined by volumizing the meshes into distance fields. The union of two distance fields is easily calculated by taking the minimum of each value in the field (Frisken et al.[3]). Once the composite distance field for each stroke has been generated, we can generate an offset isosurface for the union of the strokes which is watertight and has no self-intersections. Increasing the size of the offset increases the "thickness" of the strokes.

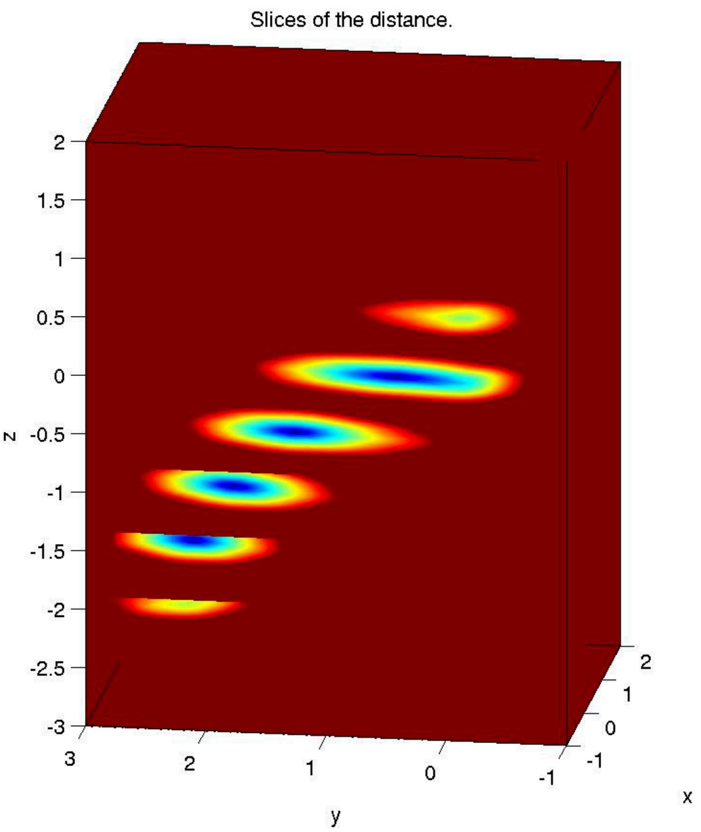


Figure 3. The Distance Field of a Tube Stroke

However, many algorithms that generate distance fields require a closed mesh, that is, the mesh must define a space that is "inside" and "outside" the mesh. Therefore, it is necessary to extrude the strokes to give them some volume. Extruding, along with isosurface extraction, thickens the strokes.

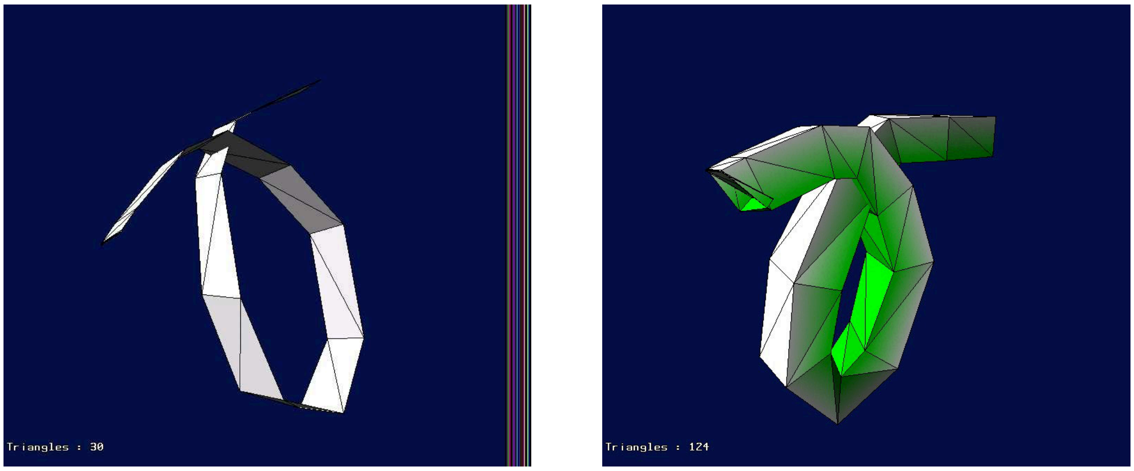
By enlarging the amount of extrusion, we give each stroke more "depth" without sacrificing sharp edges that characterize individual strokes. Selecting larger isosurface values creates "blobbier" strokes, allowing strokes that are fairly close together to form one surface, at the expense of edge quality.

Implementation

Our system consists of a multi-step process. First, the vertices of each stroke are flattened for compatibility with the depth field algorithm. Then the "flat" zero thickness stroke is extruded to give it depth. Once extruded, the distance field algorithm is run. The corresponding distance field for each stroke is merged in a composite distance field, and color information is stored. Finally, an isosurface is extracted from the distance field, and each vertex from the resulting mesh is colored.

First, the stroke's mesh is "flattened". In order for the depth field algorithm to work correctly, every vertex in the mesh must be unique. Multiple vertices with the same coordinates are collapsed into a single vertex. As the stroke is read, each vertex is hashed and each subsequent vertex with the same coordinates is collapsed into the first vertex. For example, in a cube represented by six planar patches, each corner shares three vertices, each with a different vertex normal. These three vertices are collapsed into one.

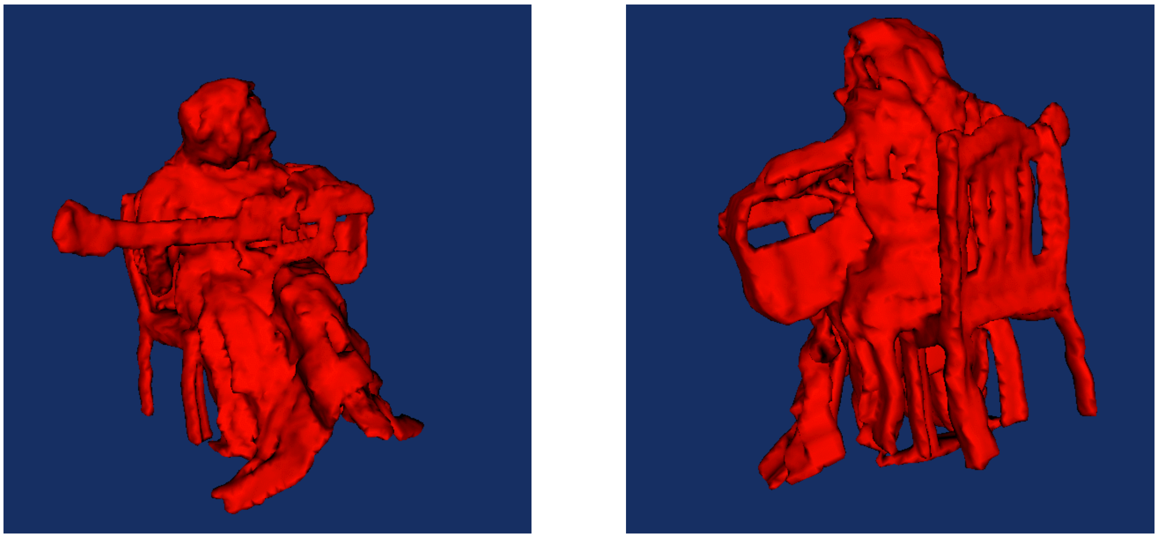
Next, the flattened mesh is extruded to give it volume, both for compatibility with the distance field algorithm, and to thicken the strokes. First, an extruded patch is created. Each vertex in the mesh is duplicated and moved along the extrusion normal. This extrusion normal is determined by calculating a weighted average of the face normals surrounding a given vertex. The edges of the extruded patch and the original mesh are then stitched together to form a closed surface.



Figures 4 and 5. A ribbon stroke that has been extruded. The extruded patch is shown in green.

Next, the distance field is generated. CaveSculpture uses Sean Mauch's Fast Algorithm for Computing the Closest Point and Distance Transform [6] to compute the distance fields for each stroke. Although other methods for volumizing a mesh exist, including Sethian's Level Set and Fast Marching Methods [9], or Nooruddin and Turk's Ray-Stabbing Technique [8], we chose Mauch's algorithm for its speed. As each stroke is added into the composite distance field, the color of the closest point is sampled and stored in a separate grid structure.

Then, the composite distance field is fed into the Visualization Toolkit's implementation of the Marching Cubes Algorithm [5]. The Marching Cubes Algorithm generates a waterproof mesh based on an isosurface value. Finally, each vertex in the resulting mesh is colored using the grid structure saved during distance field generation.



Figures 6 and 7. Views of the uncolored isosurface.

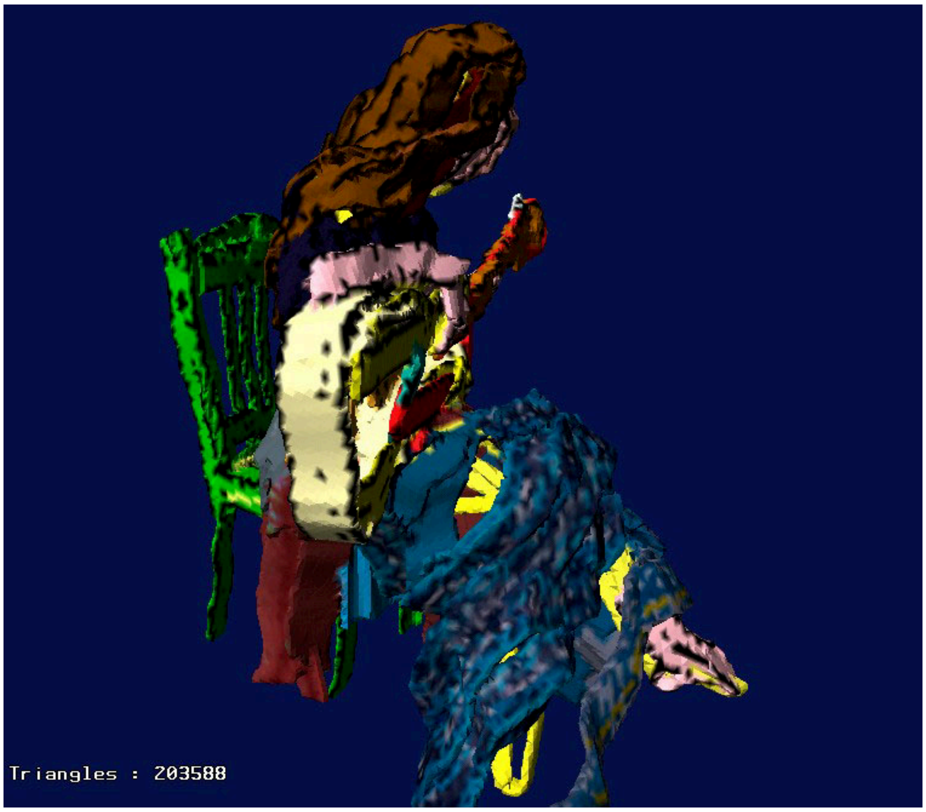


Figure 8 and 9. Views of completed watertight mesh, colorized.

Conclusion and Future Work

While CaveSculpture has been successful in creating physical models from a CavePainting, many CavePaintings cannot be rendered due to dropped strokes. The largest problem with our process is the tendency for the distance field algorithm to fail during a stroke, causing that stroke to be absent from the sculpture. The process works well with models which have large numbers of similarly sized strokes in close proximity to each other, where one stroke failing will not jeopardize the structural integrity of the sculpture.

Several factors contribute to the omission of a stroke. Because grid spacing for the distance field is determined by the extents of the painting, a stroke that is very small in relation to the painting may fall between grid spaces. Barring adaptive resolution distance fields, the only solution for this is to increase the resolution of the grid. Self-intersections may also play a role in dropped strokes. Exceedingly complex strokes seem to cause the distance algorithm to fail, as does increasing the thickness of the stroke, either by extrusion or by increasing the maximum distance the algorithm calculates. Using a different algorithm that deals more robustly with self intersections (such as Sethian's [9]) may resolve this problem.

CaveSculpture is currently being integrated into CavePainting, so that a user creating a painting can interactively see how strokes will look when they are thickened and joined. Future work should also focus on dealing with orphaned strokes (strokes that are not joined to other strokes), implementing higher quality texturing through mip-mapping, automatically determining the optimal thickness through extrusion and isosurface extraction, and reducing, if not eliminating, dropped strokes.

Bibliography

[1] CHEN, H. AND SUN, H.. 2002. Real-time haptic sculpting in virtual volume space. In *Proceedings of the ACM symposium on Virtual reality software and technology 2002,* 81—88.

[2] CUTLER, B., DORSEY, J., MCMILLAN, L., MÜLLER, M., AND JAGNOW, R. 2002. A Procedural Approach to Authoring Solid Models. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques 2002,* Computer Graphics Proceedings, Annual Conference Series, 302–311.

[3] FRISKEN, S. F., PERRY, R. N., ROCKWOOD, A. P., AND JONES, T. R. 2000. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, 249–254.

[4] KEEFE, D.F., FELIZ, D.A., MOSCOVICH, T., LAIDLAW, D.H., AND LAVIOLA, J.J. 2001. CavePainting: a fully immersive 3D artistic medium and interactive experience. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 85–93.

[5] LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *Computer Graphics (Proceedings of ACM SIGGRAPH 87)*, 21(4), 163–169.

[6] MAUCH, S. 2000. A Fast Algorithm for Computing the Closest Point and Distance Transform. http://www.acm.caltech.edu/~seanm/software/cpt/cpt.pdf

[7] MCDONNELL, K.T., QIN, H., AND WLODARCZYK, R.A. 2001. Virtual clay: a real-time sculpting system with haptic toolkits. In *Proceedings of ACM SIGGRAPH 2001* Computer Graphics Proceedings, Annual Conference Series, 179—190.

[8] NOORUDDIN, F. S., AND TURK, G. 2000. Interior/exterior classification of polygonal models. In *IEEE Visualization 2000*, 415–422.

[9] SETHIAN, J. A. 1999. *Level Set Methods and Fast Marching Methods*, 2nd ed. Cambridge University Press, Cambridge, United Kingdom.