Sculpting Polygonal Models using Virtual Tools

James R. Bill
Suresh K. Lodha
Computer and Information Sciences
University of California
Santa Cruz, CA 95064, USA
bill@geomet.com lodha@cse.ucsc.edu

Abstract

We describe a sculpting system for designing free-form polygonal models using virtual sculpting tools and mesh refinement operations. Virtual tools are geometric objects with shapes described by superquadric equations. The user controls a virtual tool to sculpt a free-form polygonal model starting from a mesh. The vertices of the mesh respond to a tool in a semi-realistic manner, allowing the user to push, pull, and deform the mesh in a variety of ways. Mesh refinement operations allow the user to control refinement levels and create smooth objects. Interactive shadows and a virtual trackball considerably enhance the intuitive manipulation and sculpting of models. The strength of the system has been demonstrated by sculpting free-form surfaces such as a dog, a fish, a banana slug, and a dinosaur, all with widely varying levels of detail.

Keywords: computer graphics, computer sculpting, interactive modeling, polygon meshes, shadows, smoothing, subdivision, virtual tools.

1 What is Computer Sculpting?

Ideally, the term computer sculpting should be used to mean a computer modeling system in which the goals and techniques of traditional sculpting are emulated. However, the goals and techniques of traditional sculpting vary widely depending upon the artist and the material used for sculpting. This variety is reflected in the computer sculpting literature as well. The employed models include polygon, point-based, parametric, algebraic and implicit models. The techniques for creating models include the use of virtual sculpting tools, decay functions, physics-based dynamics and variational principles. The user interface varies from traditional inputs such as keyboard and mouse to the use of pressure-sensitive pens and virtual reality equipments. Therefore in this work, the term computer sculpting is used in a very broad sense. Nevertheless, the key goal of computer sculpting is the design of free-form objects. A flexible computer sculpting system allows designers to translate their understanding of form directly into a finished geometric model.

1.1 Overview

The goal of this work is to create an intuitive computer sculpting system. This has been achieved by a novel unification of known techniques, some of which have been adapted to our purposes from use in different contexts. Our work builds upon the techniques of interactive polygon manipulation using decay functions initially proposed by Parent [Par77] and extended by Allan et al [AWW89]. We further enhance these capabilities by incorporating adaptive subdivision and smoothing techniques. Moreover, we have used virtual tools to identify regions of interest, which can then be pushed or pulled by the user in an intuitive way. In previous sculpting systems, virtual tools have been used mostly in conjunction with boolean operations. Finally, interactive shadows and use of a virtual trackball provide the user the intuition necessary to manipulate these objects with ease.

2.0 Previous Work

It is rather difficult to provide a comprehensive survey of existing computer sculpting systems in a very limited space. Therefore we focus on previous research on polygon mesh sculpting and virtual tools. Other sculpting systems are mentioned very briefly. For details, see [Bil94].

The four most significant polygon mesh based sculpting system are those of Parent [Par77], Allan et al [AWW89], Leblanc et al [LPMTT91], and Elson and Malone [Els90b, Els90a]. Parent initiated use of the basic vertex-movement and decay function techniques used in all of these systems. His is the only polygon mesh system that uses virtual tools. Allan advanced the use of decay functions and defined "move-vertex" as the fundamental deformation technique; both concepts are reflected in our work. Leblanc chiefly concentrated on improving the interface to an Allan-like system. Elson's papers illustrate models sculpted with the S-Geometry polygonal modeler created by Malone. This system used winged-edge representation for polygonal meshes and unlike other systems relied heavily on subdivision and smoothing techniques.

While the basic operation of moving a point is central to interactive modeling, virtual tools offer a higher level of deformation control. They allow multiple primitive operations to be applied to a model simultaneously, in an intuitive fashion. Virtual tools have been used in a number of modeling systems, most typically as a method of defining boolean operations to be applied to solid and volumetric models. They are used in this way by Parent (whose polygonal models described solids), Naylor [Nay90], Mingxian et al [MFD93], and in a related way by Galyean [GH91]. Mingxian uses CSG-based virtual modeling tools in a non-free-form CAD system. Pushing and pulling points as we propose to do has been done to some extent by Brewer [BA77], Hsu [HHK92], and Szeliski [ST92]. Brewer pushed parametric surfaces' control points with a planar tool. Szeliski used tools to apply and cancel forces in a particle-based modeler. Hsu used pushing and pulling tools to impart free-form deformation (FFD). Riesenfeld [Rie89] described high level design tools for shaping spline models.

In addition to the polygon-based sculpting systems, there are other well-known computer sculpting systems that employ parametric, algebraic, implicit or point-based representations. The free-form deformation technique initially proposed by Sederberg and Parry [SP86] is based on deforming lattices by means of moving control points, that represent parametric surfaces. Since then several improvements and extensions have been suggested [BB91, HHK92, CJ91]. Hierarchical B-spline refinement techniques [FB88], S-patches [Loo90], spline surfaces over irregular meshes [Loo94] and topological design of sculptured surfaces [FRC92] also use parametric surfaces. Szeliski [ST92] proposed a point-based representation, which has been further developed in the more recent work by Welch and Witkin [WW94], which uses a combination of point-based and parametric representation. The effectiveness of this system in representing artistic models for use in animation is unclear, since the models created by them so far still seem to be highly geometric in nature. Another class of sculpting systems based on physics-based dynamics has been proposed by Terzopoulos [TF88]. Variational principles have been used by Celniker and Gossard [CG91], Moreton and Séquin [MS92], Welch and Witkin [WW92, WW94] and Halstead et al [HKD93] to create free-form surfaces. Implicit and algebraic representations have been used by Bajaj [Baj92], Bloomenthal [BW90] and Wyvill [Wyv92] to design free-form surfaces. Finally, the 3D paint system of Williams [Wil90] uses a technique in which the painting in 2D is translated into 3D sculpting by correlating the colors in a 2D painting with depths in 3D. A technique resembling 3D paint is also used to this end in a physics-based implicit modeling system by Pentland et al [PW89].

3.0 Sculpting System Features

This section presents an overview of the most notable features of the system – polygonal mesh modeling, virtual tools, decay functions, mesh refinement operations, shadows and virtual trackball.

3.1 Polygon Mesh

Polygonal meshes are effective in sculpting applications due to their simplicity and generality; their simplicity makes them easy to understand, manipulate, and implement, while their generality allows them to represent free-form objects to any desired degree of precision. In fact, it is termed *the* most popular type of geometric modeling representation by Paouri et al [PMTT91]. Other benefits include and the availability of graphics machines optimized for the display of polygonal surfaces. Some of the difficulties associated with the polygon mesh representation include large memory requirements and lack of control on geometric continuity.

The system uses the *winged-edge polyhedra* data structure to represent both the mesh and the tools. Winged-edge polyhedra were originally developed by Baumgart [Bau74] to represent solid geometric models. Our implementation, based on the the winged-edge data structure recommended by Hanrahan and Glassner [Han82, Gla91, Bil94], is described in detail in [Bil94].

Initially, the user initially selects a starting mesh or model, which may be read from a file or may be chosen from a number of predefined initial configurations, including plane, sphere, tetrahedron, cylinder, and box shapes. The user also chooses an initial level of refinement. Default meshes are triangular, although the system supports N-faceted polygonal meshes as well.

3.2 Virtual Tools

Virtual tools are 3D geometric objects of different shapes and sizes. Our tools have two main attributes: shape and action. Currently, the supported basic tool shapes are box, cylinder, and sphere, all of which are rendered as polyhedra. Since the tool can be interactively scaled along any principal axis at any time, a variety of tool shapes such as ellipsoids, cubes, and disks can be generated from the three basic forms.

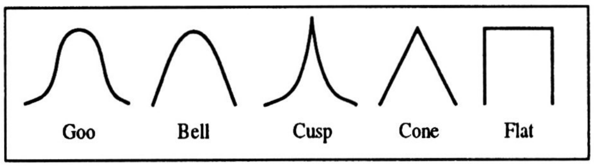


Figure 1: Shapes of decay functions in the basic set.

Each virtual tool is represented internally in two ways: as an analytical superquadric equation [Bar81, FB81, PW89], and as a winged-edge structure of faces, vertices, and edges corresponding to the equation. Superquadrics are extensions to the standard *quadric* equations, which describe the familiar ellipsoid, torus, hyperboloid of one sheet, and hyperboloid of two sheets. The equation is used in collision detection calculations, while the winged-edge structure is used for the rendering of the tool.

The three tool actions supported are termed define-region, push, and pull. *Define-region* tools simply define regions of the mesh for the mesh refinement operations to be applied to. *Push* tools act like a solid and push away vertices they come in contact with. *Pull* tools pull vertices in contact with the tool as it is moved.

3.3 Decay Functions

The user must also define a currently active *decay function* for the push and pull actions. The decay function determines the way a vertex translation propagates to surrounding vertices. All decay functions possess a range and a type. The range determines the number of surrounding vertices that can be affected. Figure 1 illustrates the basic shapes of the standard set of decay functions; these include Goo, Bell, Cusp, Cone, and Flat. The corresponding equations can be found in the original thesis [Bil94].

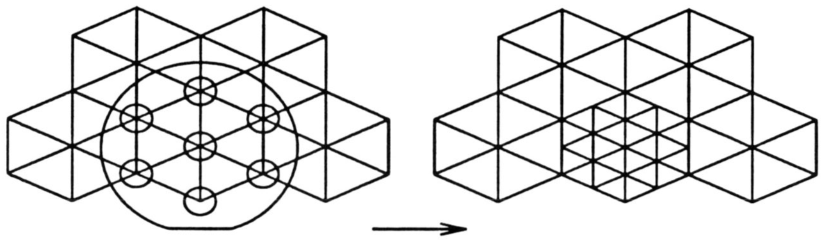
4.1 Mesh Refinement Operations

Supported mesh refinement operations include adaptive mesh subdivision, smoothing, hole-filling, and deletion of subparts of a mesh. *Subdivision* allows the region to be subdivided to whatever degree is desired. Furthermore, a region can be *smoothed*, which transforms the mesh subregion into one that appears smoother. Depending on the type of smoothing done, the number of vertices in the region may or may not increase. *Hole-filling* allows a hole in the mesh to be replaced with a face. *Deletion* removes all vertices and incident edges in a a region, and can be used to either introduce holes to the mesh or to replace mesh subregions with a single face.

3.4.1 Adaptive Subdivision

The system supports three types of subdivision, which we refer to as subdivision with propagation, subdivision without propagation, and triangulation. Figure 2 illustrates the difference between subdivision with propagation and subdivision without propagation. In subdivision with propagation, the faces adjoining the selected region are also subdivided such that no faces with more than 3 sides are created as a result of the subdivision. In subdivision without propagation, no subdivision occurs outside of the selected region, so faces outside the selected region may increase in number of vertices. *Triangulation* refers to subdividing only those selected faces that are not already triangular. The two distinct methods of subdividing individual faces are illustrated in Figure 3 and are described below.

Single triangular faces are subdivided using the *quaternary* method described by Samet [Sam90]. In this method, a single triangle is replaced with four triangles by connecting the midpoints of each edge of the triangle to form a new central face and three new faces adjacent to the central triangle and incident to the original triangle's corners (see fig. 3). This method maintains triangularity, limits vertex degree (new vertices all have five or six neighbors) and breaks up existing faces such that the new faces have a fairly regular shape. The previous method cannot be used for N-sided faces for several reasons: the central face created will not be triangular, the new faces may have widely varying areas, and the method is not well-defined for concave faces. Therefore, N-sided faces are divided by connecting each corner of the face to a single new central vertex. Currently the face's vertices are averaged to calculate the new vertex. The face's *centroid*, or center of area (a notion similar to a physical object's center of mass), would lead to a more regular


Vertices selected by intersection with tool
Selected faces subdivided without propagation

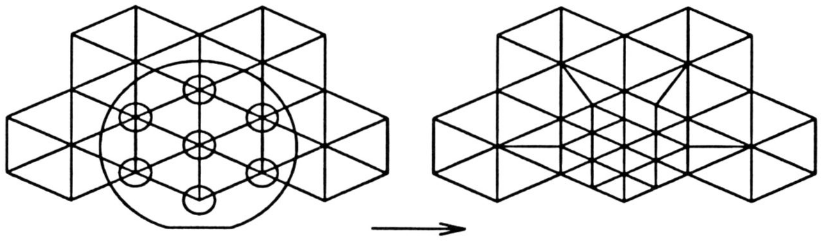

Vertices selected by intersection with tool
Selected faces subdivided with propagation

Figure 2: Triangular subdivision, with and without propagation.

subdivision. Since this method does not introduce additional vertices to adjacent faces, there is no notion of propagation of this type of subdivision; faces can be subdivided with no consideration of adjoining faces. The implementation of this method is therefore much simpler than that of the previous technique.

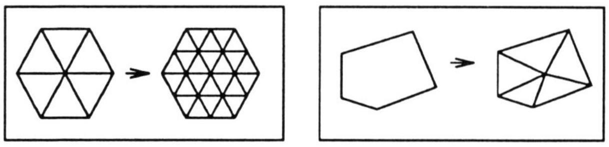


Figure 3: Quaternary triangular face subdivision and N-sided face subdivision.

3.4.2 Adaptive Smoothing

The best-known smoothing algorithm is perhaps Chaikin's corner cutting algorithm for biquadratic tensor-product B-spline surfaces and its generalization to bicubic tensor-product B-spline surfaces [Cha74]. Unfortunately, Chaikin's algorithm only works for rectangular meshes. The two other best-known smoothing techniques, those of Doo and Sabin [Doo78] and Catmull and Clark [CC78] are generalizations of Chaikin's algorithm to arbitrary meshes but still work best for rectangular and nearly rectangular meshes. Our system uses an adaptive version of the subdivision algorithm proposed by Loop [Loo87]. Loop's subdivision algorithm seems to be the most judicious choice because it is a generalization of the box-spline smoothing algorithm from *triangular* meshes to arbitrary meshes. The models created using Elson and Malone's S-Geometry system suggest that they have used the Doo-Sabin algorithm, though this is not explicitly stated in their publications [Els90b, Els90a]. Allan et al also use a "smoothing" decay function which appears to do a sort of averaging with no subdivision step [AWW89], although no concrete details are provided.

3.4.3 Filling Holes

Users may refine meshes further by filling and creating holes and to replacing mesh regions with single faces. To fill holes, we define the average point of the vertices on the perimeter of the hole, and connect each vertex to this new vertex, replacing the hole as if it was an N-sided face.

3.4.4 Deleting Mesh Regions

A two-step algorithm is used both to replace regions with a single face and to create holes. First, all edges which border two selected faces are deleted, which eventually results in all selected regions being reduced to a single selected face. If the intent was to replace all selected mesh regions with single faces, we are finished at this point. If the intent was to create holes, we then delete the remaining singular selected faces and update the edges around them to point to outside world.

3.5 Shadows

In our system, a single perspective view of work space is maintained at all times. In order to indicate the spatial relationship between the tool and the

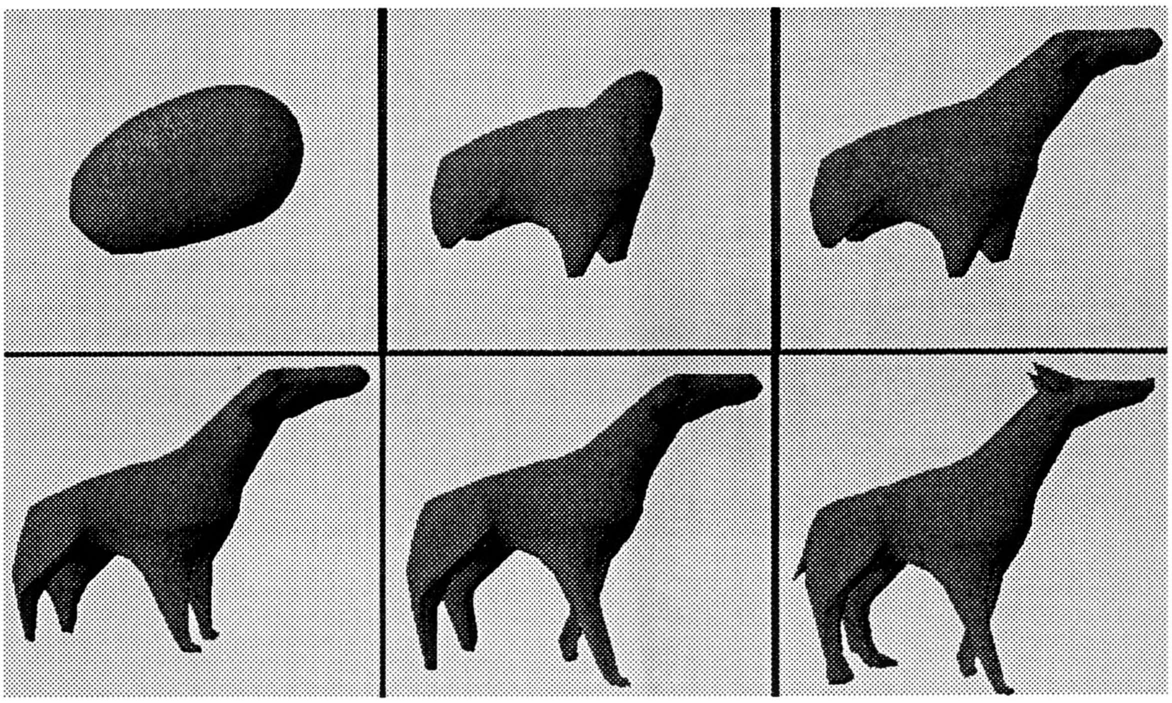


Figure 4: Major steps in the creation of a dog model.

model better, the tool can cast a shadow [Wag92] on the model, and the positions of the two light sources can be freely modified. The algorithm used is an adaptation of the *shadow volume* technique initially proposed by Crow [Cro77] and updated by Hudson [Hud92]. Our algorithm improves upon these methods through use of the stencil bitplanes available on modern higher-end Silicon Graphics workstations. Full details are provided in [Bil94].

3.6 Virtual Trackball

Each object has either a local center of rotation (COR), which moves with it as the object is translated, or a global COR that remains stationary. Rotations around the COR can be in any direction and are controlled with a virtual trackball of the type described by Chen [CMS88]. We increase the power of trackball-based rotation by allowing users to freely edit the COR of each object. At any time, users may choose to define a point in world space by using a 3D crosshair that is manipulated in the same manner as the mesh and tools. That point can than be declared to be the new COR for the tool or the mesh. Objects' CORs can be declared global or local at any time as well; they are local by default, as this is generally more intuitive and useful. Global scaling of the mesh and tools, using sliders, is also possible.

4.0 Models Created

This section illustrates a few models created with the system to demonstrate its capabilities. We first give a detailed description of the creation of a dog model. A collection of other interesting models created using the system is then presented. Each model will be described by both an illustration and complexity statistics (number of vertices, edges, and faces).

Figure 4 shows six stages in the sculpting of a dog, supposed to be of the Doberman Pinscer variety. The initial ellipsoid-shape, shown in the upper left corner, was created through two applications of subdivision with smoothing to an initial box shape. In the top center image, the beginnings of the legs and neck have been pulled out of the torso. A Goo decay function was used for the rear legs, a Cusp function for the front legs, and a Bell function for the neck. In the third frame, the neck region was subdivided and a crude head shape formed. By the fourth frame, the tips of the legs have been pulled out and front paws added, and in the following frame the legs have been stretched further and pulled into a walking pose. The final (lower right) frame shows the resulting model after the head, paws, and tail regions, respectively, have been subdivided and detailed. The model in this image is made up of 2523 vertices, 4746 faces, and 7267 edges. Figure 5 shows a detail after the entire model was smoothed with subdivision one final time, resulting in a model with 9927 vertices, 19850 faces, and 29775 edges.

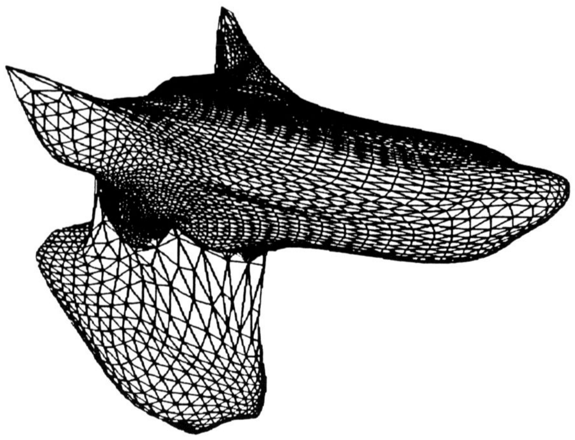


Figure 5: Detail of dog face after smoothing with subdivision.

Figure 6 presents a simple fish sculpted out of a sphere and rendered in hidden-line mode. Note the adaptive subdivision in regions where more detail is needed, such as the eye, fins, and tail. The fish is made up of 525 vertices, 1489 edges, and 966 faces, and originally was a sphere.

In Figure 7 we see a 3D version of our campus mascot, the Fighting Banana Slug. This model was created during a live demo of the program and illustrates what may be done in a very short time using the system (the modeling time here was only about 5 minutes). The slug was created from a sphere and consists of 1036 vertices (a large proportion of which are grouped in the highly subdivided eyestalks region), 1966 faces, and 3000 edges.

A model of the famous dinosaur triceratops is seen in Figure 8. The triceratops was initially a cylinder; the final shape consists of 1700 vertices, 3181 faces, and 4879 edges.

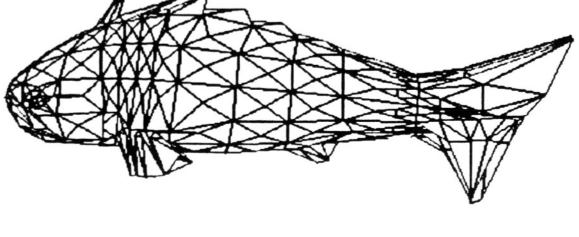


Figure 6: A fish.

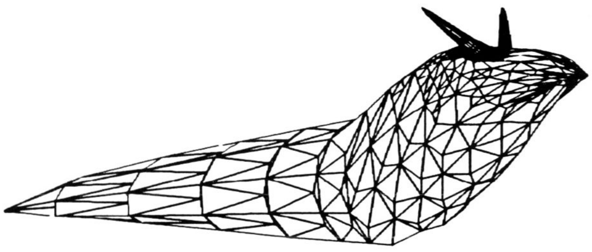


Figure 7: A banana slug.

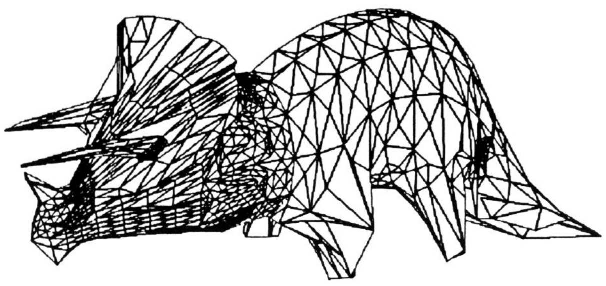


Figure 8: A dinosaur: triceratops.

5.0 Conclusions and Future Work

We described an intuitive sculpting system for designing free-form polygonal models. Its notable features include a novel combination of virtual tools, decay functions, adaptive subdivision and smoothing, interactive shadows, and virtual trackball. We have demonstrated the effectiveness of the system by deigning several complex models that could be used in animation.

The two major difficulties of the current system are excessive storage requirements and the difficulty in aligning the deformation along the tool axes. We plan to incorporate mesh optimization [Tur92, HDD$^+$93] and multiresolution techniques [LDW94, EDD$^+$95] to compress polygonal meshes and use them more effectively in subsequent applications. We would also like to add the facility of constraining the tool movements to be tangential or normal to the surface. Other possible improvements and enhancements include spatial decomposition techniques to determine collision with the tools, improved winged-edge data structure implementation to reduce memory requirements, incorporation of paint and texture tools, and addition of virtual reality input devices.

Acknowledgments

We would like to thank Jane Wilhelms, Alex Pang and the anonymous referees for many thoughtful comments which has helped us to improve our presentation. This research was partially supported by National Science Foundation grant CCR-9309738.

References

[AWW89] J. Allan, B. Wyvill, and I. H. Witten. A methodology for direct manipulation of polygon meshes. In R.A. Earnshaw and B. Wyvill, editors, *New Advances in Computer Graphics*, pages 451–469, Tokyo, Japan, June 1989. CG International, Springer-Verlag.

[BA77] J. A. Brewer and D. C. Anderson. Visual interaction with overhauser curves and surfaces. *Proceedings of SIGGRAPH '77*, pages 132–137, 1977.

[Baj92] C. Bajaj. Smoothing polyhedra using implicit algebraic splines. *Proceedings of SIGGRAPH '92*, pages 79–88, 1992.

[Bar81] A. H. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications*, 1(1):11–23, January 1981.

[Bau74] B. G. Baumgart. Geometric modeling for computer vision. Technical Report CS-463, Dept of Computer Science, Stanford University, 1974.

[BB91] P. Borrel and D. Bechmann. Deformation of n-dimensional objects. *International Journal of Computational Geometry and Applications*, 1(4):427–453, 1991.

[Bil94] J. R. Bill. Computer sculpting of polygonal models using virtual tools. Master's thesis, Department of Computer and information Science, University of California – Santa Cruz, Santa Cruz, CA 95064, 1994.

[BW90] J. Bloomenthal and B. Wyvill. Interactive techniques for implicit modeling. *Computer Graphics*, 24(2):109–116, March 1990.

[CC78] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer Aided Design*, 10(6):350–355, November 1978.

[CG91] G. Celniker and D. Gossard. Deformable curve and surface finite-elements for free-form shape design. *Proceedings of SIGGRAPH '91*, 25(4):257–266, 1991.

[Cha74] G. M. Chaikin. An algorithm for high-speed curve generation. *Computer Graphics and Image Processing*, 3:346–349, 1974.

[CJ91] S. Coquillart and P. Jancene. Animated free-form deformation: An interactive animation technique. *Proceedings of SIGGRAPH '91*, pages 23–26, 1991.

[CMS88] M. Chen, S. J. Mountford, and A. Sellen. A study of interactive 3-d rotation using 2-d control devices. *Proceedings of SIGGRAPH '88*, pages 121–129, 1988.

[Cro77] F. C. Crow. Shadow algorithms for computer graphics. *Proceedings of SIGGRAPH '77*, pages 242–247, 1977.

[Doo78] E. Doo. A subdivision algorithm for smoothing down irregular shaped polyhedrons. In *Proceedings of the International Conference on Interactive Techniques in Computer Aided Design*, pages 157–165, New York, 1978. IEEE.

[EDD$^+$95] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multiresolution analysis of arbitrary meshes. *Manuscript*, 1995.

[Els90a] M. Elson. Displacement animation: Development and application. In *Character Animation by Computer: SIGGRAPH 1990 Course Notes #10*, pages 14-36. August 1990.

[Els90b] M. Elson. Winged edge polyhedral models: Their use in the construction of characters, speech, emotion, and body language. In *State of the Art in Facial Animation: SIGGRAPH 1990 Course Notes #26*, pages 21–42. August 1990.

[FB81] W. R. Franklin and A. H. Barr. Faster calculation of superquadric shapes. *IEEE Computer Graphics and Applications*, 1(3):41–47, January 1981.

[FB88] D. R. Forsey and R. H. Bartels. Hierarchical B-spline refinement. *Proceedings of SIGGRAPH '88*, pages 205–212, 1988.

[FRC92] H. Ferguson, A. Rockwood, and J. Cox. Topological design of sculptured surfaces. *Proceedings of SIGGRAPH '92*, pages 149–156, 1992.

[GH91] T. A. Galyean and J. F. Hughes. Sculpting: An interactive volumetric modeling technique. *Proceedings of SIGGRAPH '91*, 25(4):267–274, July 1991.

[Gla91] A. Glassner. Maintaining winged-edge models. In J. Arvo, editor, *Graphics Gems II*. Academic Press, Boston, MA, 1991.

[Han82] P. M. Hanrahan. Creating volume models from edge-vertex graphs. *Proceedings of SIGGRAPH '82*, pages 77-84, 1982.

[HDD+93] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. *Proceedings of SIGGRAPH '93*, pages 19–26, 1993.

[HHK92] W. M. Hsu, J. F. Hughes, and H. Kaufman. Direct manipulation of free-form deformations. *Proceedings of SIGGRAPH '92*, pages 177–184, 1992.

[HKD93] M. Halstead, M. Kass, and T. DeRose. Efficient fair interpolation using Catmull-Clark surfaces. *Proceedings of SIGGRAPH '93*, pages 35–44, 1993.

[Hud92] S. E. Hudson. Adding shadows to a 3d cursor. *ACM Transactions on Graphics*, 11(2):193–199, April 1992.

[LDW94] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. Tech Report 93-10-05, University of Washigton, Seattle, 1994.

[Loo87] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987.

[Loo90] C. Loop. Generalized B-spline surfaces of arbitrary topology. *Proceedings of SIGGRAPH '90*, pages 347–356, 1990.

[Loo94] C. Loop. Smooth spline surfaces over iiregular meshes. *Proceedings of SIGGRAPH '94*, pages 303–310, 1994.

[LPMTT91] A. LeBlanc, P.Kalra, N. Magnenat-Thalmann, and D. Thalmann. Sculpting with the 'ball and mouse' metaphor. *Proceedings of Graphics Interface 1991*, pages 152–159, June, 1991.

[MFD93] F. Mingxian, T. Fernando, and P.M. Dew. Direct 3d manipulation techniques for interactive constraint-based solid modelling. *Computer Graphics Forum*, 12(3):C237–C248, 1993.

[MS92] H. P. Moreton and C. Séquin. Functional optimization for fair surface design. *Proceedings of SIGGRAPH '92*, pages 167–176, 1992.

[Nay90] B. Naylor. Sculpt: an interactive solid modeling tool. *Proceedings of Graphics Interface 1990*, pages 138–148, May, 1990.

[Par77] R. Parent. A system for sculpting 3d data. *Proceedings of SIGGRAPH '77*, pages 138–147, 1977.

[PMTT91] A. Paouri, N. Magnenat-Thalmann, and D. Thalmann. Creating realistic three-dimensional human shape characters for computer-generated films. In N. Magnenat-Thalmann and D. Thalmann, editors, *New Trends in Animation and Visualization*, pages 89–99. Wiley, New York, 1991.

[PW89] A. Pentland and J. Williams. Good vibrations: modal dynamics for graphics and animation. *Proceedings of SIGGRAPH '89*, 23(3):215–222, July 1989.

[Rie89] R. F. Riesenfeld. Design tools for shaping spline models. In T. Lyche and L. L. Schumaker, editors, *Mathematical Methods in Computer Aided Geometric Design*, pages 499–519. Academic Press, 1989.

[Sam90] H. Samet. *Applications of spatial data structures : computer graphics, image processing, and GIS*. Addison–Wesley, Reading, Massachusetts, USA, 1990.

[SP86] T. W. Sederberg and S. Parry. Free-form deformations of solid geometric models. *Proceedings of SIGGRAPH '86*, pages 151–160, 1986.

[ST92] R. Szeliski and D. Tonnesen. Surface modeling with oriented particle systems. *Proceedings of SIGGRAPH '92*, pages 185–194, 1992.

[TF88] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. *Proceedings of SIGGRAPH '88*, pages 269–278, 1988.

[Tur92] G. Turk. Re-tiling polygonal surfaces. *Proceedings of SIGGRAPH '92*, pages 55–64, 1992.

[Wag92] Leonard Wagner. The effect of shadow quality on the perception of spatial relationships. *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, pages 39–42, 1992.

[Wil90] L. Williams. 3d paint. *Computer Graphics*, 24(2):225–233, March 1990.

[WW92] W. Welch and A. Witkin. Variational surface modeling. *Proceedings of SIGGRAPH '92*, pages 157–166, 1992.

[WW94] W. Welch and A. Witkin. Free-form shape design using triangulated surfaces. *Proceedings of SIGGRAPH '94*, pages 247–256, 1994.

[Wyv92] B. Wyvill. Building models with implicit surfaces. *IEEE Potentials*, 11(3):23–26, October 1992.