EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2008)
C. Alvarado and M.- P. Cani (Editors)

Rapid sketch modeling of clouds

Jamie Wither, Antoine Bouthors, Marie-Paule Cani
University of Grenoble, LJK, INRIA, France

Abstract

*Clouds are an important visual element of any natural scene and computer artists often wish to create specific cloud shapes (for example in the film Amélie, as depicted in Fig 1). We describe a sketch based interface for modeling cumulous clouds. This interface allows rapid construction of a 3D cloud surface representation (mesh) using an underlying point based implicit surface representation. This mesh is rendered using the technique [BNM*08], resulting in a realtime cloud modeling and rendering system.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Modeling packages

1. Introduction and previous work

Modeling virtual clouds is a difficult process. Cloud formation follows the laws of fluid mechanics and thermodynamics. It is a chaotic process which depends strongly on initial conditions. Solutions based on the direct simulation of these laws [HBSL03] or even simpler cellular automata [DKY*00] are difficult to control and thus hard for computer artists to influence in order to attain a desired shape. Recent methods have been proposed to control fluid simulations [TMPS03, MTPS04] but they still require the user to supply keyframes. Moreover, these methods only simulate fluid dynamics but not thermodynamics which play an important role in cloud dynamics.

An alternative is to use procedural methods. These methods allow users to control the general shape of the clouds through large-scale tools. Stratiform clouds are easily modeled by horizontal layers of varying thickness [Gar85, BNL06], while cumuliform clouds are generally modeled as sets of ellipsoids [Gar85, Ney97, TB02, SSEH03, BN04]. Because of the puffy nature of convective clouds this ellipsoid set representation is well suited and has been used at several scale levels [Ney97, BN04] to mimic the fractal aspect of clouds.

To add details to these large-scale models (or make up for the coarse resolution of simulations) it is common to add high-frequency procedural noise [Gar85, SSEH03]. The resulting shape is then passed to a renderer, either in the form of volume densities [SSEH03] or a mesh [Gar85, TB02, BNM*08] (a discussion of cloud rendering is beyond the scope of this paper, for a survey of rendering techniques see [SSEH03, BNM*08]). Implicit surfaces are sometimes used to convert the set of ellipsoids into a manifold mesh [SSEH03, BN04].

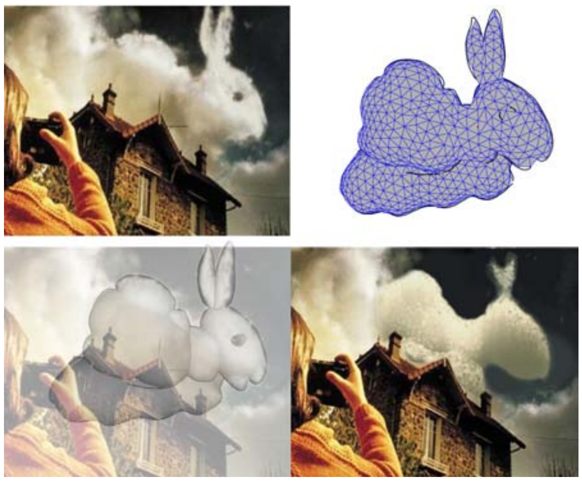
Figure 1: *Top left is an image from the film Amélie, we use it as a guide to create the mesh at the top right and the rendered result at the bottom right. The whole process took 3 minutes.*

However, controlling the overall shape which emerges from these procedural methods is not trivial. Either not enough control is given to the user or he/she has to go through a long session of 3D placement of ellipsoids. Easy and intuitive control is important in applications such as

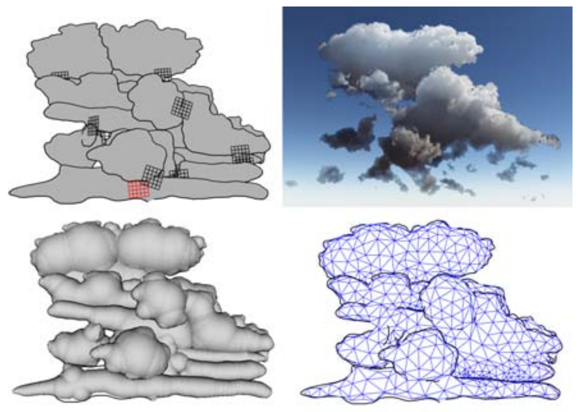
Figure 2: *A natural cloud formation modeled in three minutes and result rendered using the method of [BNM*08].*

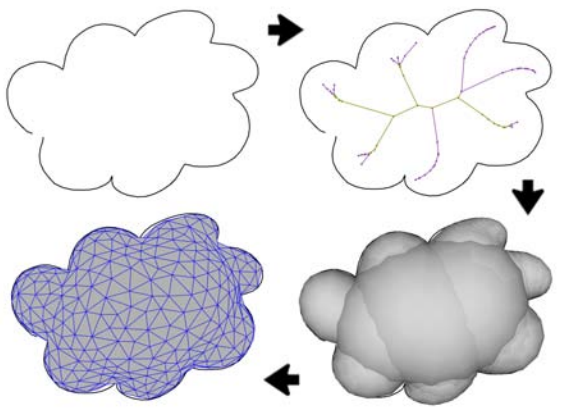
Figure 3: *The user draws a silhouette. We infer a skeleton, spheres along that skeleton and finally create a mesh.*

video games, 3D feature films and special effects (see Fig 1) where the artist may wish to achieve a very specific cloud shape.

Our aim is to allow rapid modeling of cloud shapes via a simple to use interface. We target fluffy, cumulous type clouds in particular since they carry the most visibly appealing features. Our method allows modeling of cloud like meshes in a few seconds using a sketch based metaphor. These meshes can then be rendered by a real-time technique such as [BNM*08]. Such meshes could also be used directly as animation keyframes in recent fluid control methods [MTPS04]. Using meshes allows us to simplify the modeling process and later control the rendering through further sketch based annotation.

Sketch based modeling was demonstrated to great effect in Teddy [IMT99]. More recently both variational and skeleton based implicit surfaces were used for smooth shapes in systems like [DAJ03], [ABCG05] and [SWSJ05]. The FiberMesh system [NISA07] uses differential representations of meshes to incorporate both sharp and smooth edges.

Due to the nature of clouds and to the rendering system we use [BNM*08], which adds procedural detail to a coarse mesh, we do not need such extensive shape control. Our system is rather based on some a-priori knowledge of clouds [Gar85,Ney97], telling us that the shape of cumulous clouds can be approximated by a union of spherical primitives. Thus we propose techniques to infer spherical primitives from sketched 2D outlines, and to automatically generate 3D surface detail while retaining the 2D outline. This makes our method easier to understand and faster to learn than a general sketching system or a traditional modeling package.

To our knowledge there has been no previous work combining sketch based modeling and rendering of clouds. A comprehensive cloud modeling system was presented in [SSEH03]. It used a traditional modeling environment consisting of resizable implicit ellipsoid primitives and a hierarchy of user modifiable parameters. Our approach is not as comprehensive as this and does not address cloud animation (though it is discussed briefly in the conclusion). It is rather a complementary interface which could be integrated into such a system to speed up the shape modeling.

2. User scenario

The user draws a stroke representing the whole or a section of a cloud silhouette (Fig 5). The user may alter or extend his stroke (using the technique of [Ale05]) and when he's happy with it he presses 'space' to interpret the stroke. A cloud volume (a union of spheres) matching the silhouette is then inferred. Many silhouettes can be drawn, with overlapping volumes being attached to the underlying ones. The camera can be positioned anywhere and repositioned at will.

Each interpreted stroke is stored on a separate drawing plane in the scene (Fig 4). The spheres which represent the volume of the cloud are generated along the skeleton (chordial axis transform) of the closed stroke (See §3.1).

Each drawing plane can be selected and then moved, rotated, duplicated, inflated/deflated or deleted. The user can begin a new section of cloud on a new drawing plane by drawing a new outline. If the new outline overlaps existing spheres, then the new drawing plane is defined parallel to the view plane and passes through the initial point of the silhouette projected onto the existing surface.

The user can thus quickly build up the layers within a cumulous cloud (without manually defining or moving drawing planes) by simply sketching each layer from the back to the front. To aid this process an imported image may be used as a guide (Fig 1).

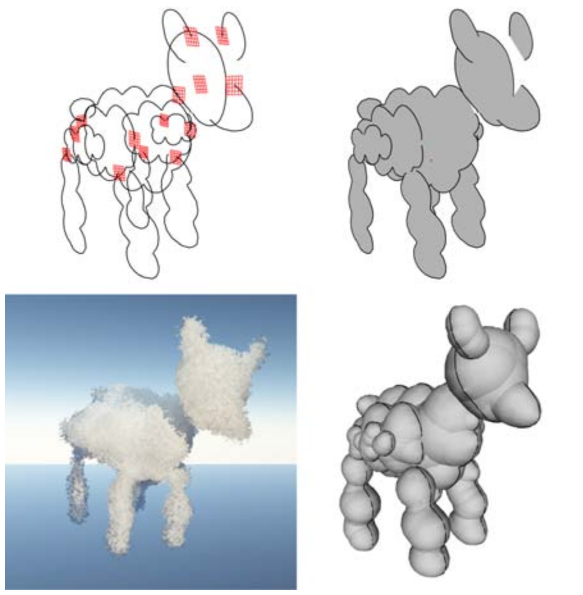

Figure 4: *This sheep took two minutes to model. Each red grid in the top left image represents a drawing plane containing a silhouette*

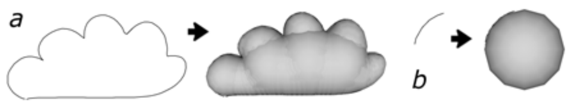

Figure 5: *a) closed stroke and interpretation. b) open stroke and interpretation.*

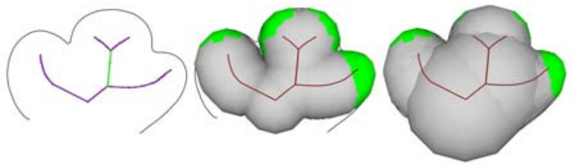

Figure 6: *Skeleton segmentation (left). Terminal branches in purple and body branches in green. Sphere inflation minimum (centre) and maximum (right). Spheres with radii below r_f are shown in green (see §3.1)*

As the spheres generated in a given drawing plane follow a planar skeleton the resulting cloud section could be too flat. The user may of course add more volume through further sketching, but we also propose a novel method for generating a plausible 3D cloud from only the 2D silhouette information (Fig 7, see §3.2).

Once the user is happy with the cloud volume a surface mesh is produced by interpreting the spheres as point based implicit primitives and blending them (see §3.3). The resulting surface is used for rendering the cloud.

A common feature of cumulous clouds is the flat bottom. The user may quickly add this feature by drawing a cut stroke which will ensure no part of the cloud mesh extends beyond the stroke (Fig 8, see §3.3).

3. Cloud shapes from sketching

3.1. Skeleton from 2D silhouette

Although we do not need to use complex implicit surfaces generated by skeletal structures, computing a geometric skeleton from the silhouette will help in reconstructing the silhouette using spherical primitives. To generate a shape skeleton we first close the outline stroke by repeating the initial point. The skeleton is then generated from the outline using a chordial axis transform (CAT) derived from a constrained delaunay triangulation using the technique of [Pra97]. This technique also defines an area metric which we use to prune insignificant branches in skeleton (we use 2% of the outline polygon area). While pruning we maintain a mapping of outline edge segments to skeleton segments. This mapping is required for later silhouette definition (see §3.2). The skeleton is a network of polylines, and for each vertex in the polylines the radius of the maximal ball (touching the outline) is stored. These points and the associated radii are used to define the spheres in the cloud volume.

To reduce the skeleton resolution we can simplify the outline stroke before generating the skeleton. We use the technique of [HB97] (realtime piecewise linear approximation using a function of chord and arc length to form a set of approximating points) and offer the user a single parameter to control the degree of simplification.

We segment the skeleton into terminal branches (sequences of skeletal segments which are connected at one end only) and body branches (connected at both ends). We observe that terminal branches appear to point towards curvature maxima in the outline, and thus can be thought of as defining shape features in the cloud outline.

When drawing large outlines the central (body) spheres of the skeleton are necessarily large as they must touch the outline. However the user may wish a flatter shape. We wish to allow the user the option of shrinking the largest spheres, without affecting the smaller spheres (which contribute to the finer details of the cloud). To provide this we offer an inflation/deflation slider control to the user, described below.

We define the minimal feature radius r_f. This radius will represent the smallest size any sphere can be shrunk to, and should be representative of the smaller spheres at the end of the skeletal branches in order to preserve fidelity to the drawn outline in these areas. We determine r_f as the maximum radius from the set of the spheres contributing to the

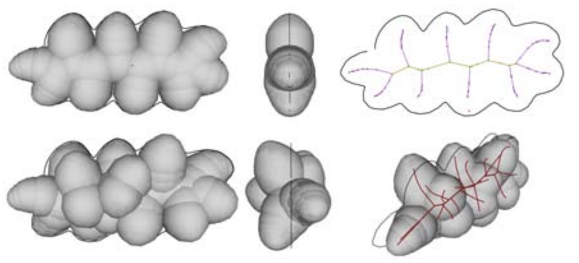

Figure 7: *The 2D skeleton is automatically placed in 3D, generating new, similar detail while preserving the existing silhouette (See §3.2)*

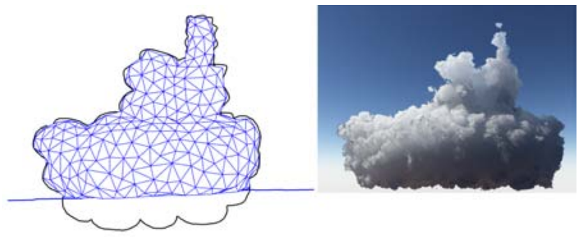

Figure 8: *The mesh can be clipped by cut strokes (left) which suppress the implicit surface below the line (See §3.3). Our rendering of this cloud using the method of [BNM*08] (right)*

last 50% of all terminal branch segments in the skeleton (Fig 6). 50% was chosen through experiment with typical outlines as retaining the spheres contributing to the smaller outline features while excluding the larger body spheres. The user may interactively specify an interpolation parameter $t \in [0,1]$ using a slider. We adjust the size of all spheres with radii above r_f using t, the minimum radius is limited to r_f and the maximum radius is the radius of the maximal ball for that point.

One disadvantage of this approach is that the fidelity to the drawn outline is reduced for spheres above the r_f radius. This could be addressed in future work by allowing ellipsoid primitives which could then be shrunk along the direction perpendicular to the drawing plane. Also defining a minimal feature radius for each skeletal branch would improve the result for outlines consisting of a wide variety of feature sizes. Another disadvantage is that the body of the cloud could get disconnected into several disjoint components if the skeleton isn't sampled densely enough. Automatic resampling could be implemented to avoid this.

3.2. Extending a flat cloud to a 3D structure

If a cloud is constructed from just one outline, it will be unrealistically flat (with a skeleton laying in its original plane) and unlike a real cloud. We present a technique which automatically generates a plausible 3D structure in case the user doesn't wish to build it up manually in layers.

To do this we consider the terminal branches of the skeleton as defining volume features that should be duplicated over the cloud shape. We segment the set of body branches in the skeleton into 'trunks' (wherever three body branches meet) and duplicate terminal ('feature') branches and reposition and rotate them around the trunks in the manner of branches around a tree, scaled to the surface of revolution of each trunk silhouette (constructed using the edge segment to skeleton segment mapping constructed in §3.1). The existing branches are allowed a small 'jitter' rotation from their initial plane, but their projection to the viewplane is maintained and the new branches are placed so as to try and maximise the distance between branches on the surface of the 'trunk'. This is similar to the approach used for sketching trees in [WBCG].

3.3. Mesh generation

Once the user has finished creating and editing his drawing planes (thus having modeled a union of spheres) a surface representation is generated by considering each sphere as a point based implicit primitive. To blend these primitives we represent them using the blending function of [Wyv92] (cubic in r^2). We convert this implicit surface to a manifold mesh using the CGAL library [CGA]. This flexible surface meshing approach [BO05] is based on restricted delaunay triangulation of the zero level set of the implicit surface. It offers a number of mesh quality parameters the user may alter. For a typical cloud this takes around 10 seconds (Table 1). Note that faster polygonisation methods exist (e.g. [SWG05]) but as we need only a simple mesh with no special surface properties using a library was appropriate.

Finally we observe that most cumulous clouds have a flat bottom. We enforce this by allowing the user to specify a large 'clipping' sphere using a stroke. This sphere acts as a large negative primitive in the implicit function which ensures the implicit surface generated is flat along the bottom.

4. Results and discussion

The test system is an Intel Xeon Quad Core 2.8GHz, 2GB RAM, with NVidia Quadro FX 1400. For performance times for models created by the authors see Table 1.

The modeling times are typically less than 2 minutes even for fairly complex looking clouds. The correspondance between the final rendered clouds and the input strokes is good, as can be seen from the resulting images.

We invited a computer artist with experience in modeling clouds to use the system. Previously she had modeled 3D

Table 1: *Result timings. The mesh parameters are a distance bound and radius bound, see [CGA]*

Name	Modeling (min)	Mesh gen. (sec)	Spheres	Facets	Param.
Sheep	2	18	896	2838	0.1, 0.3
Rabbit	2	14	486	1682	0.05, 0.15
Ship	1	2	398	852	0.1, 0.3
Fig 2	3	26	1866	2248	0.1, 0.3

clouds using traditional modeling packages by manually creating, deforming and positioning spheres within the scene. The work was tedious and required hours of effort. With our system she created in a few minutes clouds that had previously taken her a few hours.

Although in theory clouds could be modeled using a standard sketch-based modeling system for free form shapes, our method saves the user time by relying on specific a-priori knowledge. The use of spherical primitives and the automatic extension to a 3D structure makes the shape cloudlike. The flat bottom and the ability to shrink internal primitives while preserving the size of those on the outline helps constructing clouds in layers.

The method of generating a 3D cloud structure §3.2 works well for simple cloud outlines, but could be improved for more complex shapes. We would like to develop more sophisticated ways of segmenting the skeleton into trees and their child feature branches, and thus extend the volume new cloud detail may occupy, while still being constrained to the user's input. The procedural cloud structure approach of [BN04] is another way to generate 3D structure and may be well suited for animation, however it doesn't offer the user detailed control over the shape, as only the highest level of the shape hierarchy is user defined. Branches offer natural handles for the user to move surface features around. Our method could be used to generate the level zero structure for [BN04] and interesting future work would be to constrain such an approach to the user supplied silhouette.

Our prototype implementation isn't optimized, and mesh generation times could be improved by removing redundant spheres from the skeleton definition. We already offer outline simplification as a method for reducing the number of spheres required, but additionally we could pre-cull many spheres which don't contribute significantly to the surface definition.

5. Conclusion and future work

We have presented a system for rapid modeling of clouds using a sketch based interface. Quality meshes can be created and rendered within a minute. We proposed a method for automatically generating 3D surface detail from a 2D silhouette.

We would like to enhance our system by using more a-priori knowledge of cloud structure. In particular rendering parameters such as lacunarity could be set through the use of sketched annotation.

The skeletons defined in different drawing planes are currently completely independent. An interesting extension would be investigating ways to link the individual skeletons to give one complete 3D skeleton. This would also have implications for animation.

We have not addressed animation in this paper and it would be interesting future work to consider how different sketches could be used as keyframes in an animation. Two approaches come to mind. The simplest approach would be to use the generated mesh directly as a keyframe for a technique such as [MTPS04]. More complex would be to find sensible mappings between skeletons in different keyframes and then animating the resulting sets of spheres.

The rendering system we use [BNM*08] interprets thin sections of the mesh as having low density - hence the ears in Fig 1 are smaller than the user may have expected. Annotation could be used to add hints about density to the rendering system used.

Our 3D surface detail system could be improved for more complex cloud shapes by more sophisticated segmentation of the original skeleton before distributing copied details. Interesting future work would be to incorporate an approach such as [BN04] but modifying it to support user supplied silhouette constraints.

Our implicit reconstruction currently only supports spherical primitives. Adding support for more general primitives such as ellipsoids would be a natural enhancement.

Acknowledgements

The authors would like to thank the reviewers for their valuable comments. Also thanks to Laurence Boissieux for evaluating the system, Fabrice Neyret for his comments and Fred Boudon for his code contributions to §3.2. Jamie Wither is supported by a grant from the European Community under the Marie-Curie project VISITOR. The project was supported by the ANR NatSim grant.

References

[ABCG05] ALEXE A., BARTHE L., CANI M.-P., GAILDRAT V.: Shape modeling by sketching using convolution surfaces. In *Pacific Graphics* (Macau, China, 2005), Short paper.

[Ale05] ALEX J.: *Hybrid Sketching: A New Middle Ground between 2- and 3-D*. PhD thesis, Massachusetts Institute of Technology. Dept. of Architecture, 2005.

[BN04] BOUTHORS A., NEYRET F.: Modeling clouds shape. In *Eurographics (short papers)* (august 2004), M. Alexa E. G., (Ed.).

[BNL06] BOUTHORS A., NEYRET F., LEFEBVRE S.: Real-time realistic illumination and shading of stratiform clouds. In *Eurographics Workshop on Natural Phenomena* (sep 2006).

[BNM*08] BOUTHORS A., NEYRET F., MAX N., BRUNETON E., CRASSIN C.: Interactive multiple anisotropic scattering in clouds. In *ACM SIGGRAPH Symposium on Interactive 3D graphics and games (I3D)* (2008).

[BO05] BOISSONNAT J.-D., OUDOT S.: Provably good sampling and meshing of surfaces. *Graphical Models 67*, 5 (September 2005), 405–451.

[CGA] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

[DAJ03] DE ARAUJO B., JORGE J.: Blobmaker: Free-form modeling with variational implicit surfaces. In *Comunicaçao ao 12 o Encontro Portugues de Computaçao Grafica* (October 2003), pp. 17–26.

[DKY*00] DOBASHI Y., KANEDA K., YAMASHITA H., OKITA T., NISHITA T.: A simple, efficient method for realistic animation of clouds. In *SIGGRAPH Proceedings* (July 2000), pp. 19–28.

[Gar85] GARDNER G. Y.: Visual simulation of clouds. In *SIGGRAPH Proceedings* (July 1985), vol. 19, pp. 297–303.

[HB97] HORST J. A., BEICHEL I.: A simple algorithm for efficient piecewise linear approximation of space curves. In *Image Processing Proceedings.* (1997), vol. 2, pp. 744–747.

[HBSL03] HARRIS M. J., BAXTER W. V., SCHEUERMANN T., LASTRA A.: Simulation of cloud dynamics on graphics hardware. In *Graphics Hardware* (July 2003), pp. 92–101.

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: a sketching interface for 3d freeform design. In *SIGGRAPH Proceedings* (1999), pp. 409–416.

[MTPS04] MCNAMARA A., TREUILLE A., POPOVIĆ Z., STAM J.: Fluid control using the adjoint method. vol. 23, ACM, pp. 449–456.

[Ney97] NEYRET F.: Qualitative simulation of connective cloud formation and evolution. In *Eurographics Workshop on Computer Animation and Simulation (SCA)* (1997).

[NISA07] NEALEN A., IGARASHI T., SORKINE O., ALEXA M.: Fibermesh: designing freeform surfaces with 3d curves. *ACM Trans. Graph. 26*, 3 (2007).

[Pra97] PRASAD L.: Morphological analysis of shapes, cnls newsletter 139, July 1997.

[SSEH03] SCHPOK J., SIMONS J., EBERT D. S., HANSEN C.: A real-time cloud modeling, rendering, and animation system. In *Symposium on Computer Animation* (2003), Eurographics Association, pp. 160–166.

[SWG05] SCHMIDT R., WYVILL B., GALIN E.: Interactive implicit modeling with hierarchical spatial caching. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 104–113.

[SWSJ05] SCHMIDT R., WYVILL B., SOUSA M., JORGE J.: Shapeshop: Sketch-based solid modeling with blob-trees. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling* (Aug. 2005), pp. 53–62.

[TB02] TREMBILSKI A., BROSSLER A.: Surface-based efficient cloud visualisation for animation applications. In *WSCG* (2002), pp. 453–460.

[TMPS03] TREUILLE A., MCNAMARA A., POPOVIĆ Z., STAM J.: Keyframe control of smoke simulations. *ACM Trans. Graph. 22*, 3 (2003), 716–723.

[WBCG] WITHER J., BOUDON F., CANI M.-P., GODIN C.: Seamless multi-scale sketch-based design of trees (pending publication).

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