Eurographics Symposium on Point-Based Graphics (2004), pp. 1–6
M. Alexa, S. Rusinkiewicz, (Editors)

Shadows for Incomplete Point-based Isosurfaces

paper46

Abstract

This paper presents a method for computing shadows on incomplete point-based isosurfaces. Such surfaces are obtained in our setting by a view-dependent isosurface extraction method that allows interactive exploration of large scale datasets on commodity hardware. This approach raises the need for a novel and efficient technique to generate high-quality shadows. The latter dramatically enhances the realism of the visualized isosurface and provides essential cues to perceive depth and small features of the geometry.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation - Viewing Algorithms; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Color,shading,shadowing,and texture;

1. Introduction

Isosurface extraction is an important technique for visualizing three-dimensional scalar fields. By exposing contours of constant value, isosurfaces provide a mechanism for understanding the structure of scalar data. These contours isolate surfaces of interest, focusing the analysis on important features in the data such as material boundaries, while suppressing extraneous information.

Recently, a *Point-based IsoSurfaces Approach*, termed *PIsA*, was presented to handle isosurfaces from large datasets. A key feature of this approach is its ability to provide interactive extraction and rendering of these large isosurfaces on a single desktop computer. Interactivity, however, is just one important aspect for a successful investigation.

Global illumination in general, and shadows in particular, increase the level of realism, and provide essential depth cue and spatial relationships between objects [WFG92]. Figure 1 shows an example of a point-based isosurface where shadows provide vital cues to the relative position of the registration tubes with respect to bones. The shadows also enhance the relationship between various bones and a few small cavities.

In this paper, we present a technique that builds on *PIsA*, and extends it to allow shadow casting with respect to point light sources.

The contents of this paper are structured as follows. In section 2 we review earlier work on point-based techniques and view-dependent isosurface extraction. In section 3 we review view-dependent isosurface extraction methods in general, and the new point-based approach in particular. Section 4 presents our algorithm for casting shadows for these sparse and incomplete isosurfaces. Results are presented in section 5, followed by conclusions, and future work in section 6.

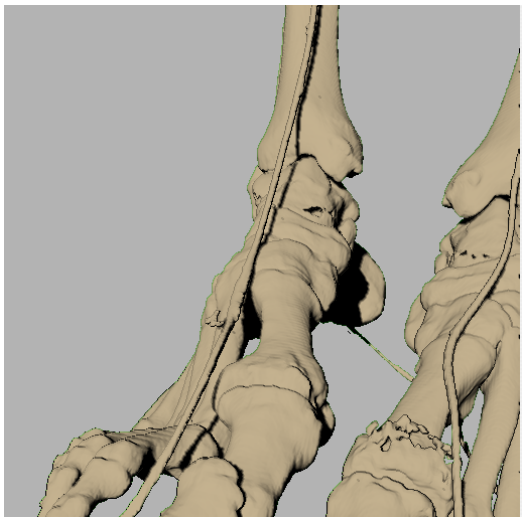

Figure 1: *Shadows can increase the level of realism and provide depth cues.*

2. Previous Work

2.1. View-dependent Isosurfaces

Isosurface extraction poses a unique challenge because its geometry is defined only with respect to a given isovalue, which the user can change interactively. Consequently, only a limited amount of information can be gathered in a preprocess for later use in an interactive session. Viewing an isosurface requires extraction and rendering phases, both of which can be hampered by the size of the isosurface, which can reach millions of polygons.

To address this issue, Livnat and Hansen [LH98] proposed an output sensitive approach based on view-dependent extraction of the isosurface. The approach is based on a front-to-back traversal of the data while, maintaining a virtual framebuffer of all extracted triangles. The virtual framebuffer is used during the traversal to cull sections of the dataset, which are hidden from the given view point by closer sections of the isosurface.

A ray casting approach was presented by Parker *et al.* [PSL*98, PPL*99]. This approach lends itself well to large shared memory machines because of the parallel nature of ray casting. An additional benefit of the ray casting approach is the ability to generate global illumination effects such as shadows.

Recently, Anonymous authors [Ano04] proposed an interactive point based isosurface extraction approach, termed *PIsA*, that extracts and represents a view-dependent isosurface as a collection of 3d points. The main thrust of this work is to enable interactive interrogation of large isosurfaces using a single desktop machine. While this approach demonstrated its ability to run faster then the ray casting approach, it is not as flexible. In particular, *PIsA* framework does not provide global illumination effects such as casting shadows. In this paper, we present a method for extending that work to provide shadow computation from a dynamic light source.

2.2. Point Based Methods

Points as display primitives have experienced a growing interest in the computer graphics community during the last years. Motivated by the increasing and typically huge size of geometric datasets to be processed, various methods have been designed that aim at extending to point sets the rich mathematical and computational framework traditionally developed for triangle meshes.

Point primitives recently found various applications for scientific visualization too[WM03, HE03]. For isosurface extraction, more specifically, Ji et al.[JSG03] used points in non-photorealistic rendering of isosurfaces for remote visualization of large data sets.

2.3. Shadows Computation

Shadows can add realism and provide essential spatial and depth cues. One of the most common approach to computing shadows is based on Willians [Wil78] *shadow maps*. A shadow map is a depth image computed from the viewpoint of a light source. To determine if a point in space is in shadow, it is transformed into the coordinate system of the light source and its distance from the light is compared to the corresponding value in the depth map. If the point is closer to the light it is lit, otherwise it is considered in shadow.

Traditional shadow maps, suffer from aliasing due to insufficient spatial sampling resolution. Fernando *et al.* [FFBG01] proposed an adaptive hierarchical scheme where a shadow map is dynamically refined in regions that contain shadow boundaries. Stamminger and Drettakis [SD02] use perspecive shadow maps which are generated in normalized device coordinate space. A jittered samples and pre-filtering system was proposed by Lokovic and Veach [LV00] to address self shadows casts by fine hair. Other effors targeted special effect such soft shadows [AAM03] and translucent material [DS03].

3. The Point-based Isosurfaces Approach

Point-based view-dependent isosurfaces [Ano04] offer a new approach of representing very large and complex isosurfaces that provide efficient extraction and fast rendering. Below, we describe in detail the general framework of this technique and the key features we will reuse for casting shadows.

3.1. View-dependent Isosurface Extraction

A View-dependent isosurface extraction relies on a hierarchical front-to-back traversal of the data using value and visibility based pruning.

3.1.1. Value-Based Pruning

Value-based pruning relies on a precomputed hierarchical representation, such as an Octree [WG90] or GridTree (nested grids) [Ano04]. Each of the tree nodes store the minimum and maximum values of its children, which allows the extraction algorithm to ignore the corresponding sub-tree if the given isovalue is outside this range. The tree can then be traversed in either depth-first, bredth-first, or best-first order.

3.1.2. Visibility Based Pruning

Visibility pruning is achieved by traversing the GridTree in a front-to-back order with respect to the view point and traversing the children in a depth first order. When a leaf node is reached, the isosurface geometry is extracted and the geometry is scanned onto an offscreen visibility framebuffer. During the traversal, the algorithm performs a visibility query on each node that passes the value-based pruning test. The visibility query is done by projecting the bounding

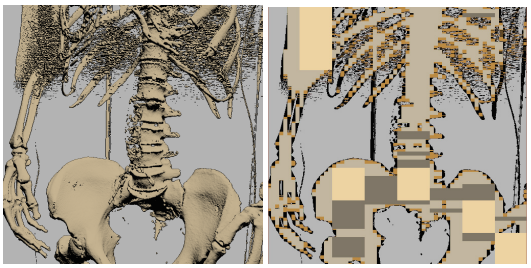

Figure 2: *A view-dependent isosurface and its corresponding final visibility map.*

box of the node onto the visibility framebuffer, and checking if it includes any pixels marked *visible*.If all of the pixels are marked *non-visible*, the entire node is hidden by previously extracted geometry and should be pruned.

To clarify the role of the visibility framebuffer in casting shadow, section 4, note that in each stage of the extraction process, the visibility framebuffer provides a status report about which pixels are still of interest, *i.e.*, in this case pixels that are marked as *visible*. Figure 2 shows an extracted isosurface along with its final visibility mask. For efficiency, *PIsA* uses a hierarchical visibility framebuffer. Different colors are used for illustration purposes only because the visibility framebuffer is implemented with only 1-bit per pixel.

3.2. Point-based Isosurface

The point-based isosurfaces approach stems from the observation that, for very large isosurfaces, the projection of triangles on the screen is very small and often even sub-pixel in size. As such, extracting triangles wastes resources during the extraction and rendering phases. We refer the reader to the *PIsA* [Ano04] paper where anonymous authors present a more complete discussion and reasoning for selecting points.

Within in the scope of this work, we note that the extracted points are based solely on the position of their projection on the screen, not on special attributes of the surface, such as curvatures or angle with respect to the observer or a light source. The point-based isosurface represents a correct reconstruction only with respect to its original viewpoint. From any other viewpoint, and in particular an offset light source, the point-based isosurface represents a sparse and incomplete reconstruction. Figure 3 shows an offset closeup view of an isosurface that demonstrates these characteristics.

4. Casting Shadows

Point-based isosurfaces present a unique challenge for shadows computation because only partial geometry is explicitly given. Most shadow generation algorithms are based on an explicit representation of the given geometry, which allows

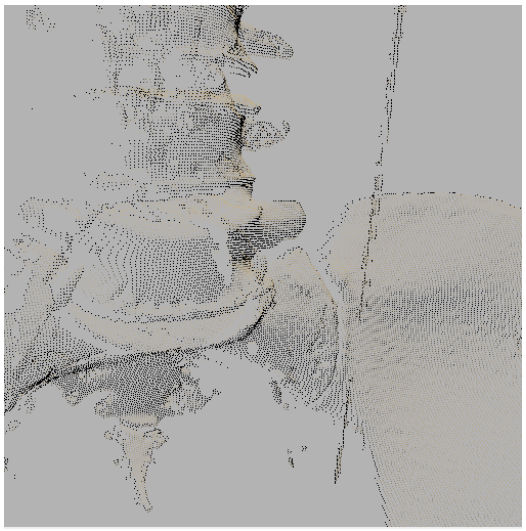

Figure 3: *A close view of the point-based isosurface shown in Figure 2*

for pre-processing, careful position of the light source, or attaching shadow maps to various objects in the scene. While not all geometry of the isosurface is generated explicitly, we do have an implicit representation of the whole isosurface using the original dataset and the given isovalue.

The ray casting method [PSL*98, PPL*99] can be used in the case of point-based isosurfaces by casting rays from the extracted point toward the light source, and traversing the data. The point-based approach, however, is faster [Ano04], on a single desktop computer, then ray casting in extracting then the original point based isosurface. In addition, the ray casting requires at least one ray per point. Alternatively, the point-based extraction approach is to determine the visibility of a sub-section of the screen at once via the projection of nodes bounding boxes.

4.1. The Algorithm

As described in section 2, there is a large body of literature addressing the sampling issues associated with shadow maps. Within the scope of this paper, we restrict our consideration to new issues induced by the notion of point-based isosurfaces, namely the incomplete and sparse reconstruction with respect to the light source, and high cost associated with isosurface extraction. We emphasize that our scene is comprise of volume data, and possibly a few widgets and landmarks. Thus shadows are mainly casted by other sections of the same isosurface. Note, however, that the isosurface can be extremely complex with tens of millions of polygons, and thousands of disconnected components of various sizes.

With these issues in mind, our shadow casting algorithm

is derived from a combination of conventional shadow maps and ray casting. In its most basic form, the algorithm is based on reusing *PIsA* for a second pass starting from the light source. With respect to the ray casting approach, we traverse the rays backward, from the light source to the isosurface points. Likewise, we do not cast light on all geometry, as done with shadow maps. Rather, occlusion information is computed only for points generated in the first pass.

4.2. Incomplete Light Map

For clarification we use the following terminology: Let G_e represent the geometry (points) of the isosurface extracted with respect to the eye viewpoint. Let G_l represent the isosurface geometry extracted with respect to the light. Let it also define the projection of a geometry, G, with respect to viewpoint v as $P^v(G)$.

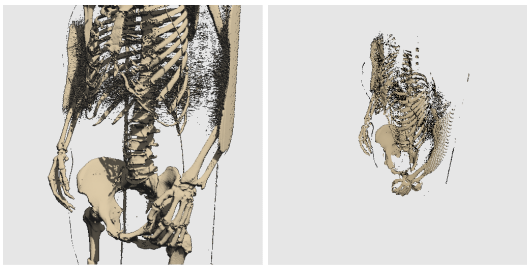

Figure 4: *Viewing an incomplete point-based isosurface from the eye* (left) *and light* (right)*viewpoints.*

Considering Figure 4, note that much of the view-dependent isosurface extracted from the light viewpoint, G_l, may not be seen from the eye nor does it contribute to the occlusion query it was meant to answer. Clearly, we need to cary out the occlusion (shadow) test *only* for the geometry G_e extracted with respect to the eye.

To remove this redundancy, we take advantage of the approach used by *PIsA* framework. Recall that in *PIsA*, the visibility framebuffer is first cleared so all pixels are marked *visible*. We extract the isosurface geometry, G_e, scan it onto the visibility framebuffer, The covered pixels are marked as *non-visible* to signal that no further extractions in that region are necessary. $(\{F(x) = \neg visible \,|\, x \in P^e(G_e)\})$.

The setup phase of the shadows casting pass, comprise of setting all pixels in the visibility framebuffer to *non-visible*. We then project G_e with respect to the light, but set the pixels to *visible*. More formally, we set $\{F(x) = visible \,|\, x \in P^l(G_e)\}$. In essence, we punch holes *only* where we need to find the closest part of the isosurface.

Note that when using the punching holes in the light visibility mask approach, the number of points the extraction from the light finds will be *at most* as large as the number of eye points. If some of the eye points fall on the same pixel the number of punched holes will be smaller, and the algorithm will run faster.

4.3. Faraway Light Source

If the distance between the light source and the isosurface is much larger then the distance between the eye and the isosurface, the isosurface will cover only a small section of the visibility framebuffer, and will be sampled at a lower resolution. As a result, many points will be projected onto the same pixel on the light framebuffer, *i.e.*, $\{P^l(x_i) = P^l(x_j) \,|\, \exists i, j \quad x_i, x_j \in G_e\}$. In this case, only one point will be lit, while the rest will be in shadow.

This problem is known as *self shadowing* in shadow maps literature. Adaptive Shadow Maps [FFBG01] and Perspective Shadow Maps [SD02] are two recent approaches that are suitable for our case.

In the scope of this work, we can make one additional improvement step before applying these solutions or a variant thereof. To solve the inconsistency mensioned above, we narrow the field of view from the light so that $P_1^l(G_e)$ will cover as much of the visibility framebuffer as possible, taking maximum advantage of the visibility framebuffer limited resolution. Here again, we face the incomplete reconstruction issue of these isosurfaces. As opposed to the traditional situation where the geometry is known, the extent of the extracted geometry G_e is unknown a priori.

One approach is to use the bounding box of the entire volume, as seen in Figure 5, but this is clearly an over estimation that leads to poor sampling. A second approach is to compute the bounding box of G_e in world space, and project that bounding box onto the light framebuffer. We choose to use a much tighter bounding box by computing it with respect to the light coordinate system. Though the light position is known, the model view and projection transformations should be defined based on the current G_e.

We start by locating the center of G_e, and using the light position to define the line of sight and the perpendicular plane. Next, we select any two perpendicular vectors on the plane, and, using the line of sight as the third axis, define the modelview transformation. Now all points in G_e can be converted to this new light coordinate system and compute their, now much tighter, bounding box. Finally, we use this bounding box to define a perspective projection. Note that we can use a perspective projection that is not symmetric with respect to the two axis on the new light plane. Figure 6 shows a full field of view of an isosurface from the viewpoint of the light, and the adjusted view.

The problem may compound even further if the light is far enough that $\{mid P^l(x) \,| = 1 \,| x = \text{sub volume}\}$. In this case, the *PIsA* algorithm creates a single point to represent the whole sub-volume. This can cause many points in G_e to be erroneously considered in shadow. The ability of *PIsA* to

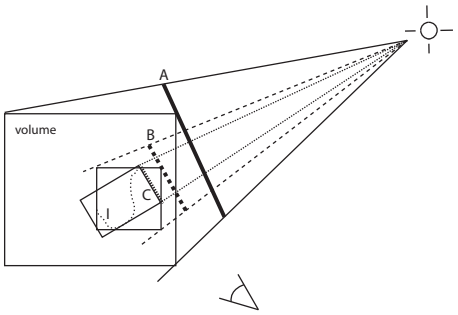

Figure 5: *Bounding box computations. A) using the volume bounding box. B using a world axis aligned bounding box of the extracted isosurface. C) using a bounding box aligned with the light axis. Note the relative projections on the vairous bounding boxes on the light framebuffer, which is situated at znear distances from te light source.*

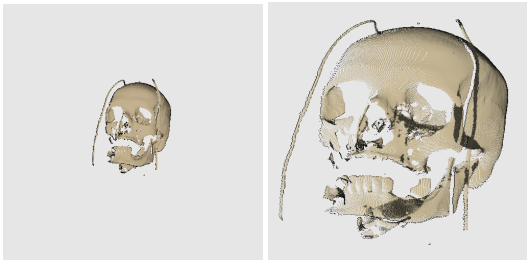

Figure 6: *View from the light: A fullview vs. deformed and zoomed view.*

create a single pixel for a far enough sub-volume is key to its interactive performance. In order preserve interactive performance, we allow the algorithm to extract such points for G_e, but turn it off for shadow casting.

5. Results

Our setup includes a Dual Processor Power Mac 2GHz G5 with 2GB memory and ATI 9800 Pro graphics card. We chose the *Visible Woman* dataset for our benchmarks because it is a relatively large dataset (512x512x1734 shorts = 867MB) that exhibits noisy complex isosurfaces. As seen in Figure 6 and Figure 4 the skeleton provides challenging test cases for view dependent isosurface extraction algorithms in general and shadow casting in particular.

All the images in this paper where produced using a 512x512 screen resolution for both the point-based isosurface extraction and for the shadowing pass. Table 1 summarize the performace of the extraction and the cost of adding shadows. It is important to note that the isosurface are extracted and rendered *as is*, *e.g.*we do not remove the noise in the torso case. This is in part because the isosurfacce is meant to show the raw MRI data and not impose any unnecessary (and possibly wrong) interpretations.

Test case	Frames per sec	Cull (msec)	Render (msec)	Points
Skul	4.8	184	24	119,398
w/ shadows	2.2	398	50	119,398
Torso	1.1	872	23	113192
w/ shadows	0.6	1639	46	113,192

Table 1: *Performance*

6. Conclusions

We presented a novel method of casting shadows for point-based isosurfaces. Our approach uses the same mechanism used to create the isosurfaces and thus providing a unified framework. We demonstrated the viability of using shadows in an interactive system as well as the visual benefits of using shadows in these highly complex and sometimes noisy scenes.

Acknowledgments

This work was supported in part by grants from the DOE ASCI, the DOE AVTC, the NIH NCRR, and by the National Science Foundation. The authors would like to thank Rachel McNeil and Charles Hansen for their contribution.

References

[AAM03] ASSARSSON U., AKENINE-MÖLLER T.: A geometry-based soft shadow volume algorithm using graphics hardware. *ACM Trans. Graph. 22*, 3 (2003), 511–520. 2

[Ano04] ANONYMOUS: Interactive point based isosurface extraction. In *Proceedings of IEEE Visualization 2004* (2004). (submitted). 2, 3

[DS03] DACHSBACHER C., STAMMINGER M.: Translucent shadow maps. In *Proceedings of the 14th Eurographics workshop on Rendering* (2003), Eurographics Association, pp. 197–201. 2

[FFBG01] FERNANDO R., FERNANDEZ S., BALA K., GREENBERG D. P.: Adaptive shadow maps. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), ACM Press, pp. 387–390. 2, 4

[HE03] HOPF M., ERTL T.: Hierarchical splatting of scattered data. In *Proceedings of IEEE Visualization 2003* (October 2003), pp. 433–440. 2

[JSG03] JI G., SHEN H.-W., , GAO J.: Interactive exploration of remote isosurfaces with point-based non-photorealistic rendering. In *Joint Eurographics - TCVG Symposium on Visualization* (2003). 2

[LH98] LIVNAT Y., HANSEN C.: View dependent isosurface extraction. In *Visualization '98* (October 1998), ACM Press, pp. 175–180. 2

[LV00] LOKOVIC T., VEACH E.: Deep shadow maps. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000), ACM Press/Addison-Wesley Publishing Co., pp. 385–392. 2

[PPL*99] PARKER S., PARKER M., LIVNAT Y., SLOAN P.-P., HANSEN C., SHIRLEY P.: Interactive ray tracing for volume visualization. *IEEE Transactions on Visualization and Computer Graphics* (1999). 2, 3

[PSL*98] PARKER S., SHIRLEY P., LIVNAT Y., HANSEN C., SLOAN P.-P.: Interactive ray tracing for isosurface rendering. In *Visualization 98* (October 1998), IEEE Computer Society Press, pp. 233–238. 2, 3

[SD02] STAMMINGER M., DRETTAKIS G.: Perspective shadow maps. In *Proceedings of ACM SIGGRAPH 2002* (July 2002), Hughes J., (Ed.), ACM Press/ ACM SIGGRAPH. 2, 4

[WFG92] WANGER L. C., FERWERDA J. A., GREENBERG D. P.: Perceiving spatial relationships in computer-generated images. *IEEE Comput. Graph. Appl. 12*, 3 (1992), 44–51, 54–58. 1

[WG90] WILHELMS J., GELDER. A. V.: Octrees for faster isosurface generation. *Computer Graphics 24*, 5 (November 1990), 57–62. 2

[Wil78] WILLIAMS L.: Casting curved shadows on curved surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques* (1978), ACM Press, pp. 270–274. 2

[WM03] WELSH T., MUELLER K.: A frequency-sensitive point hierarchy for images and volumes. In *Proceedings of IEEE Visualization 2003* (October 2003), pp. 425–432. 2