Contour Rendering

Per H. Christensen
mental images

Introduction

A contour image represents the essence of a scene, a "simplified reality." Contours are used in non-photorealistic images for comics and cartoons to illustrate geometric shape, spatial relationships, color, texture and illumination. This article describes traditional and computer-generated contour rendering, gives examples of contour placement and styles, discusses some practical issues, and describes the contour shader interface of the rendering program mental ray.

Traditional Contour Images

The first use of contour lines to illustrate objects can be found in prehistoric cave paintings. The ancient Egyptians used contour drawings to decorate tombs: first a master painter drew the outline of the objects to be depicted, then a bas-relief sculptor carved into the rock along the contour lines, and finally a painter would fill in color between the carved contour lines. The Bayeux tapestry is another example of early use of contours, it illustrates William the Conqueror's conquest of England in 1066.

Contour lines are an important part of the styles of comics and cartoons. *Max and Moritz* [4] from the 1860s and *The Yellow Kid* [10] from the 1890s are widely considered to be the first "real" comics. Since then, comics have developed into a genuine art form. Storytelling in this medium requires a clear flow of the story, well-composed images that clearly present the situations and actions, selective use of colors and contour lines that illustrate the shape of objects (and possibly also their color, texture and illumination). Some of the most famous comic artists are Carl Barks [3], Hal Foster and Will Eisner [6] from the U.S., Georges 'Hergé' Remi [11], Edgar P. Jacobs, Jacques Martin and André Franquin from Belgium, and Albert Uderzo, Jean-Claude Mézières and Jean 'Moebius' Giraud from France – just to name a few. In addition, there is the entire field of Japanese comics [12].

Animation presents an additional challenge: continuity in time. Masterpieces include the classic Disney movies [13], Tex Avery cartoons [1] and Japanese "anime" [8]. To create a believable animation, the movements must have the right timing and follow some fundamental principles: squash and stretch, anticipation, follow-through, secondary action and exaggeration [13]. Furthermore, the contour placement and style have to be consistent from frame to frame. Traditionally, a lead animator makes contour line drawings ("ink line" drawings) of the most significant poses of an animated character, then an assistant makes contour drawings of the frames in between ("in-betweening") and finally color is painted in the spaces between the contour lines ("painting"). Since there are many frames per second of animation, the in-betweening and painting stages represent a huge workload.

Computer-Generated Contour Images

Computers offer advantages in modeling, animation and rendering of animations. For modeling, when an object has been modeled in 3D, it is easily viewed from all directions. For animation, it is often easier to get realistic movements with 3D models than with 2D projections, and the tedious in-betweening is done automatically. For rendering, the painting of each frame is done automatically. One might fear that artistic expression gets lost this way, but the challenging artistic tasks of developing the story, shapes and movements are still left to artists. The time-honored principles of traditional animation also apply to computer-generated animation!

One disadvantage of using computers is that modeling cartoon characters in 3D can be challenging. The traditional characters are usually conceived in 2D space, and some of them don't even have consistent 3D representations. For example, Mickey Mouse's ears are circular no matter what direction they are seen from. This is only true for a sphere. Nevertheless, his ears are supposed to be curved discs, not spheres – they are just always moving and turning perpendicular to the viewer. Aside from such problems, once the characters are modeled, the animation and rendering is much faster using computers than with traditional methods.

Computer-generated contours have been used in many movies, for example in the wildebeest stampede in Disney's *The Lion King*, in crowd scenes in *The Hunchback of Notre Dame*, in the Hun charge in *Mulan* [7], for a boat in Extrafilm's *Asterix in America*, in Ghibli Studios' *Princess Mononoke*, in Fox's *Anastasia* and in Dreamworks' *Prince of Egypt*. In the following two sections, the most important principles in computer-generated contour rendering are described.

Contour Placement

The first decision to be made regarding contours is where they should be placed. Some possibilities are illustrated in Figure 1.

In Figure 1a, only the silhouette of each object has contour lines. Despite the simplicity, we get a sense of the shape of the objects — we can already recognize the ubiquitous "Utah teapot" — and we can see that one teapot is in front of another. In Figure 1b, there are interior lines in addition to the silhouette lines. The contours mark places with a large change in depth (distance to viewer) or surface orientation. Figure 1c shows contours between different colors. For

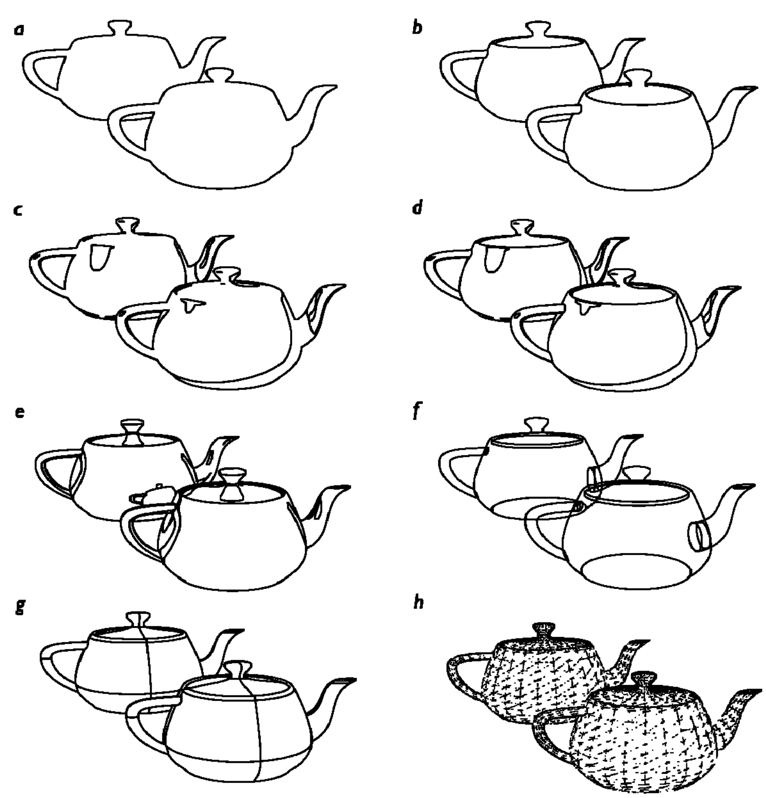

Figure 1: Contour placement (from top left): (a) silhouette lines; (b) at changes in depth and orientation; (c) between different colors; (d) combination of (b) and (c); (e) on reflections; (f) through transparency; (g) at patch borders; (h) at triangulation.

this purpose, we have given the teapot a bright color at highlight regions, a medium color in illuminated regions other than highlights and a dark color in shadow (colors not shown). Figure 1d is a combination of 1b and 1c. This is close to the traditional placement of contours in comics and cartoons, where contours are drawn at places with a large change in depth or surface orientation to clarify shape and spatial relationships, and at the border between two colors to emphasize the colors and ease painting.

An object seen in a mirror or reflected in a lake should often have the same contours as if it were seen directly. Figure 1e shows contours on reflections. Similarly, there can also be contours on surfaces seen through transparent or refractive surfaces – Figure 1f shows an example of this.

Figures 1g and 1h show some nontraditional uses of contours. They both illustrate the underlying geometric representation. Figure 1g has contours at patch boundaries, showing that each teapot is represented as 32 surface patches. Figure 1h has contours at the edges of triangles, showing the triangulation of the teapots.

Contour Style

When the positions of the contours are determined, the next question is "How should the contours look?." There are two main parameters that determine the look of a contour: thickness and color. In the simplest contour style, all contours have

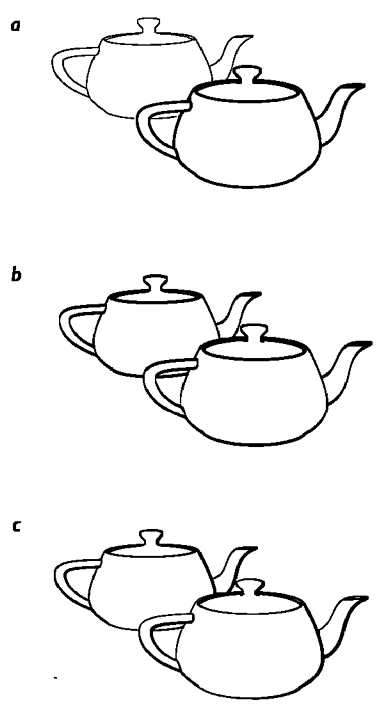

Figure 2: Countour thickness. (a) depth cue; (b) ink pen; (c) illumination-dependent.

constant thickness and color; this was the case for all the teapots in Figure 1.

Figure 2 shows some examples of more interesting contour thickness styles. A technique commonly used in traditional contour rendering is depth cue: the contour is thick if the object is near the viewer, and thin if the object is far away. Figure 2a shows an example of this. The contours in comics are sometimes drawn with an ink pen, in which case the contour is thick when the stroke is in one direction and thin in perpendicular directions. This gives a dynamic, artistic look. (This is rarely done in cartoons since it is difficult to maintain stroke directions from frame to frame.) Such an ink pen look is shown in Figure 2b. Figure 2c shows yet another style. Here, the contours are thin where the surface faces directly towards the light source, and thick where it faces directly opposite, with a continuous variation in between. With this style, the contour image gives an impression of illumination and shadows.

In other styles, the position of the contour can differ from the "correct" position. For example, the contour can be given a slight wiggle to simulate the tiny imperfections in traditional animations. A squiggly contour can indicate a rough, bumpy surface, or each contour line can be offset to simulate hand-drawn sketches.

Different objects in a scene can have different contour styles. For example, some objects can have thick contours for emphasis, or some objects can have sketchy contours to show that their shape is not final.

The contour color is also an important parameter. Figure 3 shows four different color contour styles. The top teapot in figure 3a has a contour with constant thickness and constant blue color. The contours on the teapot in front of it are colored by the underlying material: white around the highlight, yellow around other illuminated areas and brown around shadows. The color of the contours on the top teapot in Figure 3b is twice the material color, thus this contour indirectly depends on the illumination, color and texture of the material. The contour on the teapot in front of it is thick and red near the viewer and thin and blue further away; in addition, contours seen through this transparent material are only half as wide as they would otherwise be. This variation of thickness and color provides a very clear depth cue.

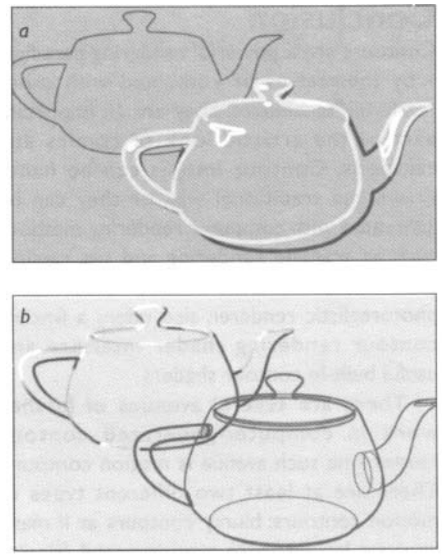

Figure 3: Color contours.

The contour color can also vary across the contour. For example, a glowing color can be obtained by varying the color and alpha from transparent on the edges of the contour to a solid color in the middle of the contour, or the color can change across the contour to generate a rainbow brush contour.

Practical Issues

Some of the practical issues that are involved in computer-generated contour rendering are described in this section.

Some surfaces should not have contours between them – for example, if they are part of the same object, but have been modeled separately. Therefore, it is sometimes necessary to specify pairs of surfaces that should not have a contour between them.

When different objects in the same scene have different contour styles, an unavoidable issue comes up: which of the two contour types should be used at the image-space intersection between two objects? We have found that using the contour type of the frontmost object works well. However, no object is in front at places where two objects abut, so some other strategy must be used there. Examples of such strategies are: using a priority list (indicating which contour types have priority over which other types), using the average color and thickness or using the contour type for the surface where the normal faces most towards the viewer.

Contours can be on top of each other in image space. These contours need to be composited in the rendering process (or in a postprocess). Several strategies can be applied: the frontmost contour can be composited over the bottom one — using either alpha-blending or the brightest color in each color band — or the thickest contour can be chosen.

Conclusion

Contours are a powerful rendering paradigm – by themselves or combined with other rendering techniques. They are an important part of the artistic look of comics and cartoons. Contour images can be hand-drawn the traditional way, or they can be generated with computer rendering methods such as scanline rendering and ray tracing. mental ray, which is mostly known as a photorealistic renderer, also offers a flexible contour rendering shader interface and useful built-in contour shaders.

There are several avenues of further work in computer-generated contour images; one such avenue is motion contours. There are at least two different types of motion contours: blurry contours as if many contour images were superimposed (similar to motion blur in photorealistic renderings) and sharp "speed lines" in the direction of motion.

Acknowledgments

Thanks to my colleagues at mental images for ideas and suggestions during the development of the mental ray contour shader interface and contour shaders. Hoang-My Christensen and Rolf Herken provided valuable comments to improve the structure and readability of this article. The razorback model in Figure 4 is from Viewpoint Datalabs.

References

1. Adamson, Joe. *Tex Avery: king of cartoons*, Da Capo Press, 1985.
2. Arias, Michael. "Toon shaders," in release notes for SOFTIMAGE|3D versions 3.7 and 3.8, SOFTIMAGE, 1996.
3. Barrier, Michael. *Carl Barks and the art of the comic*, Mark Lillien, 1982.
4. Busch, Wilhelm. *Max und Moritz*, 1865.
5. Driemeyer, Thomas. *Rendering with mental ray*, Volume 1 of 9.
6. Eisner, Will. *Comics and sequential art*, Poorhouse Press, 1985.
7. Guaglione, Eric. "The art of Disney's Mulan," SIGGRAPH 98 course notes #39, ACM, 1998.
8. Levi, Antonia. *Samurai from outer space: understanding Japanese animation*, Open Court Publishing Company, 1996.
9. mental images. *The mental ray series of handbooks*, volumes 1-3, Springer-Verlag, 1999. (To appear.)
10. Outcault, Robert F. *The Yellow Kid*, 1896.
11. Peeters, Benoît. *Tintin and the world of Hergé: an illustrated history*, Joy St. Books, 1992.
12. Schodt, Frederik L. and Osama Tezuka. *Manga! Manga! The world of Japanese comics*, Kodansha, 1986.
13. Thomas, Frank and Ollie Johnston. *The illusion of life: Disney animation*, Hyperion, 1981.

Per H. Christensen is working at mental images in Berlin, Germany. His work there focuses mainly on contour rendering, fast motion blur and global illumination. Before joining mental images, he received a Ph.D. in computer science from the University of Washington in Seattle, WA. The topic of his research was hierarchical finite-element methods for simulation of global illumination. In a previous life, he received a master's degree in electrical engineering from the Technical University of Denmark.

Per H. Christensen
mental images
Fasanenstrasse 81
D-10623 Berlin, Germany
Tel: +49-30-315-997-0
Fax: +49-30-315-997-33
Email: **per@mental.com**
http://www.gk.dtu.dk/~per/
http://www.mental.com

The mental ray Contour Shader Interface

To our knowledge, there are currently only two commercial toolsets for computer-generated contours on 3D models: SOFTIMAGE's Toon shaders by Michael Arias [2], and mental ray's contour shaders. Both are based on mental ray [5], a commercially available renderer developed by mental images. This sidebar describes mental ray's contour shader interface.

The position of contours is determined by a user-defined function. This function gets information (position, surface normal, color, etc.) about two neighboring points in the image, and decides whether there should be a contour between these two points or not. If it decides that there should be a contour, this decision is used for two purposes: if the two points are as close as they can possibly be with the specified sample density, a contour will be created between them. If they are not that close, mental ray's adaptive sampling is alerted to make sure that there are dense samples in between to find a more precise position of the contour.

The style of contours is determined by user-defined contour shaders. A contour shader gets information about two neighboring points that should have a contour between them, and computes the contour thickness and color. Each object can have its own contour shader. If the two points are on different objects, the contour shader to be called is determined by the methods described in the section "Practical Issues" on page 59.

mental ray collects the discrete contour points generated by contour shaders and merges them into straight line segments using a "connect-the-dots" strategy. These contour line segments are passed to a contour output shader. The contour output shader can, for example, generate a PostScript file or rasterize the contour line segments and composite them onto the regular image. Examples can be seen in Figure 4.

With this approach, contours are rendered at the same time as the regular image, avoiding the burden of having two rendering passes and having to composite the results in postproduction. The contours can be rendered on parallel computers and networks of computers (just as the regular rendering) to speed up the rendering process. More information about mental ray, its contour shaders and contour shader interface can be found in [9]

– Per H. Christensen

a

b

c

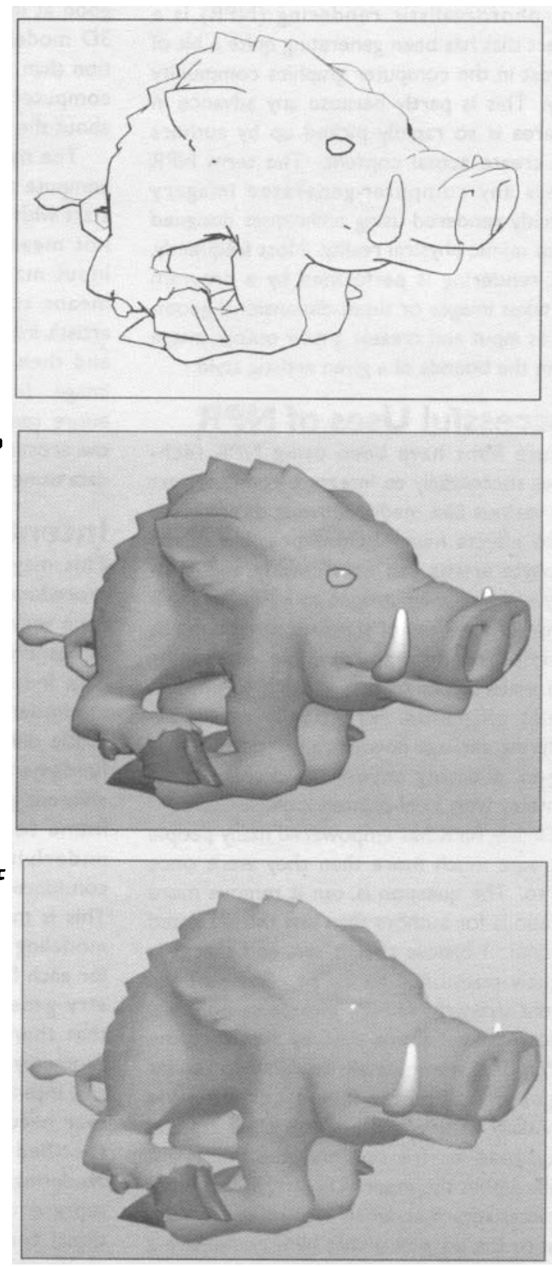

Figure 4: Contour output. (a) contour image; (b) regular image; (c) regular image with contours. ***See page 95 for color images.***

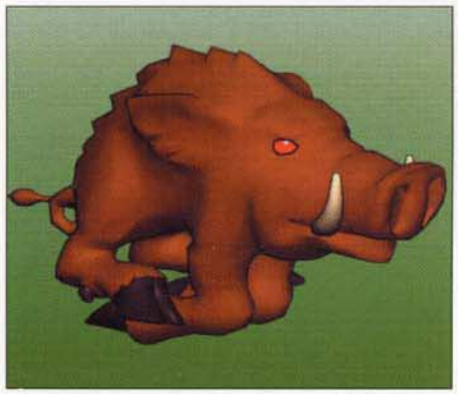

Figure 4: Per H. Christensen, The mental ray Contour Shader Interface. See page 61.