In this sketch, we present a rendering meta-primitive called the paintstroke and develop a fast, view-adaptive method for rendering it using our implementation of the A-Buffer. Since the paintstroke's shape is that of a generalized cylinder, it is not a general-purpose tool for building arbitrary objects. At the expense of this generality, however, comes a highly optimized rendering algorithm that greatly improves over traditional methods for drawing the equivalent curved surfaces.

Paintstrokes serve as useful building blocks for a variety of finely detailed natural phenomena such as fur, hair, fine branches, wicker, and pine needles. By incorporating various view-dependent image effects into the rendering algorithm, paintstrokes can be further used for modeling streams of water, icicles, and wisps of smoke. Self-shadowing is simulated on a global scale, offering an alternative to shadow mapping.

Implementation

The standard way to render a curved surface is to tessellate it into a fixed set of three-dimensional polygons. Our algorithm tessellates the paintstroke dynamically, directly making use of the image-space projection of the generalized cylinder to obtain a desirable arrangement of polygons in screen coordinates. Capitalizing on important modeling and rendering efficiencies, the algorithm arranges the polygons so as to maximize their projected screen size, thereby minimizing their number. The resulting savings in vertex transformations, rasterization overhead, and A-Buffer fragment blending more than repay the cost of the dynamic tessellation.

After transforming the control points from world to viewing coordinates, we generate Hermite interpolants for the path and radius components, and linear interpolants for the others. The paintstroke is then subdivided along its length into segments, with a granularity that adapts to its screen-projected size and curvature. Each segment is subsequently tessellated into polygons such that the side of the paintstroke closestto the viewer is always completely covered by one, two, or four polygons, depending on the user-adjustable quality level. Quality-zero paintstrokes use a single polygon per segment and represent the most efficient tiling pattern, although they do not render correctly when viewed head-on, and the distribution of normals along their breadth, derived by linear interpolation across a single polygon, is somewhat inaccurate. Nevertheless, they are an excellent choice for fur, as shown in the bottom right figure.

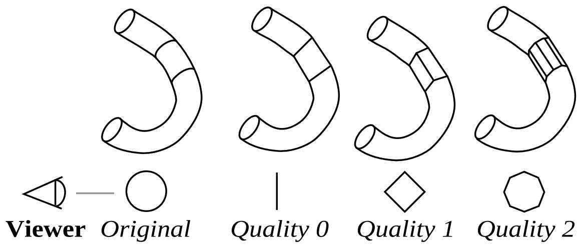


FIGURE 1 View-dependent tessellation meshes

Special Rendering Effects for Paintstrokes

The dynamic, view-dependent tessellation of paintstrokes allows for several useful rendering effects that would be difficult or impossible to achieve with a fixed polygonal model.

Lengthwise Opacity Variation

This feature simulates volume opacity, which varies according to the penetration of the light rays passing through a paintstroke segment. A simple formula involving the segment's path and the viewer's direction estimates the opacity, with a maximum value achieved when the viewing angle is along the path, and a minimum value when it is orthogonal to it.

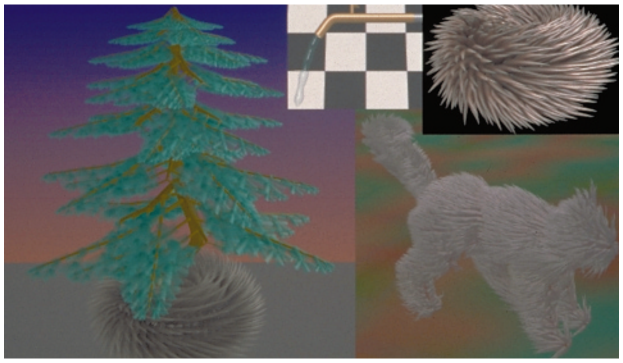
Breadthwise Opacity Variation

The opacity of the paintstroke can also vary across its breadth, allowing the user to specify distinct opacities for the centre and for the edges. Our view-dependent tesselation scheme makes this effect particularly easy to implement. It can be used to produce fuzzy paintstrokes or to simulate the Fresnel effect for transparent fluids, as shown in the top middle figure.

Global Lighting Algorithm

This algorithm works well when a large number of control points are fairly uniformly distributed over a convex volume. Each control point of a paintstroke contains a user-specified global normal and depth value. The former indicates the normal of the global shape to which the control point belongs, and the latter the relative depth from the surface. The estimated amount of light penetration at each control point is computed using the depth, global normal, and average light direction. The control point's reflectance is then scaled accordingly. The left and top right figures are good examples of this effect's efficacy.

Thanks to Michiel van de Panne for the original idea and motivation for this research, and for the many useful discussions about it.



Animal (bottom right) modeled by Nick Torkos using approximately 12,000 quality-zero paintstrokes and as many polygons, rendered at 640x480 resolution in six minutes on a 250MHz R4400. Torus (top right) modeled using 1,250 hybrid quality (level zero / one) paintstrokes, rendered at 256x192 in 15 seconds on a 200MHz 604e. Water (top middle) modeled with four paintstrokes (three for tap, one for stream), rendered at 256x192 in four seconds on a 200MHz 604e.

Ivan Neulander
University of Toronto
ivan@dgp.utoronto.ca
http: / / www.dgp.utoronto.ca / people / ivan / ivan.html