Non-Photorealistic Animation and Rendering
Creative Use of existing Software

Prof. Tilmann Kohlhaase
University for Applied Sciences Darmstadt

Abstract

The charm of traditional animation often results from the imperfection of frame by frame drawings. We developed a simple to use algorithm to simulate hand drawn lines inside of an existing software package. The direction, length and density of the lines renders the surface quality as curvature, reflection and occlusion in a handmade style. During animation the lines are updated to follow the mesh deformation. The solution integrates seamlessly with the existing production workflow.

Keywords: Non photorealistic Animation and Rendering, Handmade style, Line drawing, ink and paint illustration.

1 Introduction and Motivation

The look of computer animation tends to clean and perfect surfaces. One of the fascinations of hand drawn animation are the imperfections evoked through the passion of traditional animators, drawing frame by frame with pen and paper. The nature of animation itself has a build in level of abstraction that correlates with the look of this traditional approach.

On the other hand computer animation might be used to support and speed up the animation workflow by physical simulation, controlled tweening and to free the camera from the theatrical constraints. We show a solution for the simulation of a handmade render style well integrated inside the computer animation workflow. To proof the concept we realised a short animation movie for preschool children creating the target group could be distracted from the character imperfections evoked through the passion of traditional animators.

1.1 Observations

Context: An Observational Study

2 Technical Approach

Similar solutions in software packages, like toonshading, are working with simplified models and don’t evaluate direction and length of lines, that an artist would use do modulate the topography of the object.

There are good solutions for outline rendering but there is no support for rendering the surface structure and curvature by hatching and scribbling in a hand drawn way. Existing standalone solutions are not suitable in a tight production process. Based on related works and own tests we developed a model that draws lines in the direction of the strongest curvature on concave bendings and towards the lowest one on convex surface angles. During the animation the object surface is evaluated frame by frame to adapt the directions to the actual deformation. For the illustration of lines we tested various solutions like sprites or paint effects, but finally came up with the use of particle streaks simulating the fading of drawn lines perfectly.

3 Implementation and Future Work

We applied the standard particle system to the surface, emitting particles in regard of the light, direction and surface occlusion, resulting in denser hatching in dark areas. In the next step the algorithm evaluates the direction of the strongest or lowest curvature at the origin of each particle. To avoid problems with irregular uv textures the world coordinates of the surrounding mesh geometry was evaluated using a build in Maya script “nearestPointOnSurface”. Thus assigning each particle streak to its actual direction. The particles streak’s direction lies always in a plane tangential to the surface. From the camera direction this creates a denser hatching towards the outline of the characters. Settings for randomness allow for a very vivid hatch which we reduced during the project because we felt our target group could be distracted from the character’s facial expressions by too much flurry. To avoid too much randomness we finally constrained the particle origins to the surface. In addition we could influence colour, length, transparency and thickness of the lines as well as the behaviour dependent on the angle of the surface and the camera axis. Hidden particles on the back of the objects are not drawn.

To stay inside the production workflow we wrote a script that could be applied to any polygonal mesh, creating and connecting all the necessary nodes and runtime expressions. The workflow proved very stable during the whole production. During a simulation run the particles were cached to allow for fine tuning of the look of the rendered streaks, without recalculating the directions.

The performance suffered from the Maya script “nearestPointOnSurface” being not capable of multi threading.

Developing a solution to use multiple threads as well as integrating light measurement inside the script are some issues for future development. Over all this project showed us with a rewarding use of standard software outside of the box, developing an innovative and touching render style.

References

Tobias Isenberg a.o. 2006. Non-Photorealistic Rendering in Context: An Observational Study
Tilke Judd a.o. 2007. Apparent Ridges for Line Drawing
Forrester Cole a.o. 2008 Where Do People Draw Lines?

e-mail: tilmann.kohlhaase@h-da.de