A shading pipeline for 2D animation techniques

Hedlena Bezerra[1], Luiz Velho[2], Bruno Feijó [1]
[1] PUC-Rio - Pontifícia Universidade Católica do Rio de Janeiro
[2] IMPA - Instituto Nacional de Matemática Pura e Aplicada

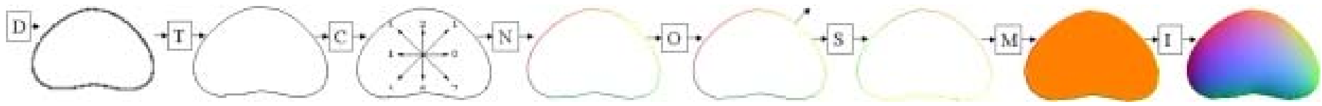

Figure 1: Pipeline - [D] Digitization; [T] Skeletonization; [C] Curve Extraction; [N] Normals; [O] Orientation; [S] Smoothing; [M] Mask; [I] Interpolation

1. Introduction

Traditional animation is often leaded by 2D cartoons where each frame is hand draw. Throughout the history of cartoon-making, it is possible to find a variety of techniques that still demand a huge effort from the animator, and sometimes, skills that require years of practice.

In computer graphics, the artist controls the rendering process by changing lights, materials, textures, and shades. The primary components typically used to illuminate a point on a surface are its position and surface normal. With hand-made drawings, the surface normal is unknown, and the position information lacks depth.

This work describes a pipeline to process 2D images as a preparation for shading effects. Our pipeline is especially suitable to cel-based digitized cartoons. Processing is image-based and is intended as a way to minimize the amount of user intervention.

2. Pipeline description

Figure 1 depicts pipeline main stages. First stage, [D], is called Digitization where frames are scanned from cels into a digital format.

In second stage, [T], the Skeletonization, we are interested in morphological information about the object curves. A thinning algorithm erodes, therefore reducing objects without changing their topology.

The Curve Extraction stage, [C], implements contour coding. An arbitrary curve is composed from a sequence of unitary vectors with a limited set of possible directions. A chain code method using an 8-connected grid models its layout and direction. A 2D normal vector can be easily derived at any point in that representation, by taking the perpendicular direction to the curve orientation (phase [N]).

A point in the curve may have two normal vectors that share the same direction, but have opposite orientations. The following stage, [O], guesses for each segment, which is the best vector orientation by inspecting its dominant curvature, and by choosing the one that points outwards the center of the curve.

The Smoothing stage [S] low-pass filters the normal fields produced in [N] as follows: for each point, a normal vector in a point is calculated by a vector addition of its previous and next neighbor's normal vectors.

Once normal vectors are estimated wherever possible, an interpolation stage, [I], employs sparse interpolation to approximate them over the remaining image according to [Johnson]. The [M] stage constructs a mask to indicate which region of the image this interpolation must be applied on. These areas are identified via a flood fill algorithm.

3. 3D shading effects

The pipeline described above makes possible to apply 3D shading effects to 2D cel animation. Most rendering techniques can be adapted to use 2D positions and approximate normals to illuminate a drawing.

The architecture is suitable not only for photo-realistic but also for non-photo-realistic techniques for rendering animated scenes. These techniques can also be applied to many kinds of animation such as, for example, frame-by-frame animation. The complete system provides a flexible framework for designing and generating new styles of 2D/3D animations.

4. References

[Johnson] Johnson, Scott F. "Lumo: Illumination for Cel Animation". Proceedings of the 2nd International Symposium on Non-photorealistic Animation and Rendering. Annecy, France.