Interactive Computer Graphics: Project Progress Report Non-photorealistic Shading- Gooch Shading, Cel Shading, Stippling Shading

Rohit Singh (u1210167)

22, April 2019

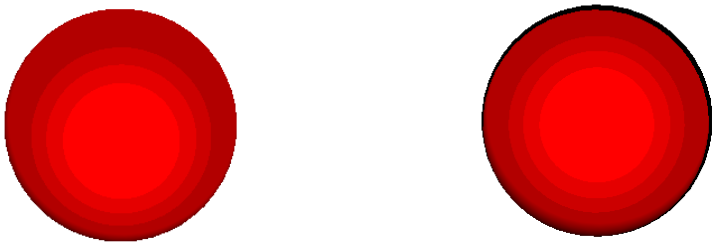
1 Overview

Non-photorealistic rendering is a very broad term in computer graphics. It covers all rendering techniques and visual styles that are obviously and deliberately different from the photorealstic appearance of physical objects. For my project I aim to implement a number of most widely used NPR shading modesls- Cel/Toon shading, Gooch Shading and Stippling Shading. When implementing one or more shaders for any specific visual style, one should first determine which features of the style have to be implemented. This is mainly a task of precise analysis of the visual style and a certain degree of the understanding of the shader language and its capablities.

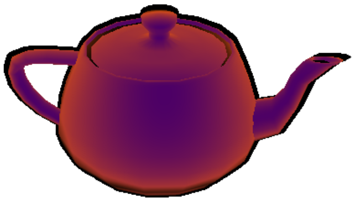
2 Project Implementation Description:

I have successfully implemented the three proposed shading models- Gooch Shading, Cel Shading, Stippling Shading. Since all three models require outline highlighting, I spent quite sometime exploring ways to do this. I tried out thresholding, sobel filter and stencil buffer. The final implementation uses thresholding to get internal outlines and stencil buffer for the out side shape. This is an improvement over the approach I had shared as part of my progress report which uses only thresholding and so object's extrernal outlines are weak. Here are screenshots comparing the last and the current techniques.

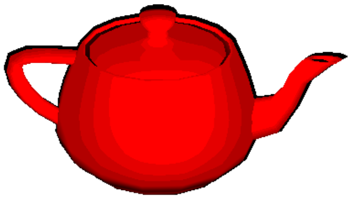
Using only Thresholding —— —— Using Thresholding and Stencil Testing



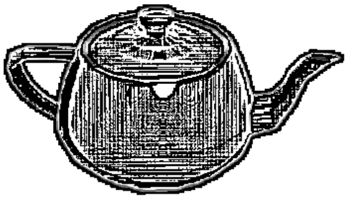
2.0.1 Gooch Shading



2.0.2 Toon Shading



2.0.3 Stippling Shading



3 Conclusion:

These NPR shading algorithms I used in this project implemented the core functions of each model. And the final results have obviously different artistic expressions and vary distinctly in their saliant features. Considering the implementation process, most of my effort went towards parameters adjustment to get an acceptable output results. I realised that creating a beautiful border lines is a challenging venture. At present, my project is able to express the main functions, but it also needs to improve to meet the practical usage.

4 Developmnet and Runtime Environment

Operating System: Windows 10
System Type: 64-bit OS, x64-based processor
IDE: Visual Studio Community 2017 Version 15.5.7
Compiler: Microsoft C / C++
Additional libraries: glfw-3.2.1, GLEW-1.11.0, GLM, stb_image, glad
Note: All relevant includes can be found in the submitted **Includes** folder and their source is listed as a comment at the top of each file.

5 Instructions to run project:

Step1: Unzip the submitted **Final_Project_NPR_Quad_u1210167.zip** file.
Step2: Open solution file **Final_Project_NPR_Quad_u1210167.sln** in Microsoft Visual Studio 2017.
Step3: Set up additional dependencies for glfw, glew and glfw libraries in Visual Studio 2017 for the project.(I have included the .libs for glew and glfw in the project folder.)
Step4: Run the solution.

6 References:

- https://www.cs.utah.edu/ shirley/papers/gooch98.pdf
- http://www.chrille.maps4vips.info/Archiv/FIN/NZ/COMPSCI715/Project/Tutorial.html
- https://learnopengl.com/

*********Thanks for Reading*********