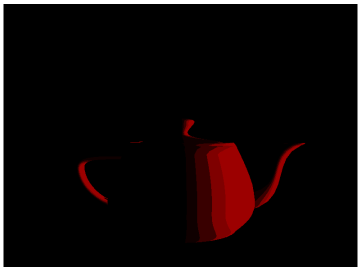
Computer Graphics Workshop 3 - Shaders and Lighting

Goals

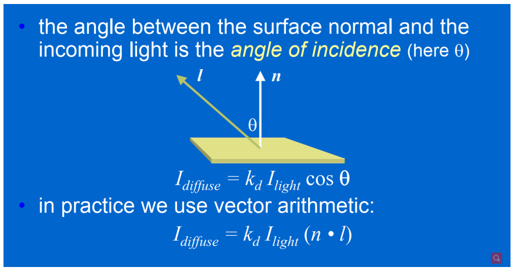
1. Understand using shaders
2. Implement a simple cartoon shader
3. Implement diffuse shading
4. Simulate a basic lighting change over the time of a day

Problem introduction

Based on previous workshops, you are able to read a height map from a file and display it as a surface in 3D space. As for this workshop you should have the starting point compiled and running on Linux and have read through the source code. Meanwhile, you should have learned related theory about lighting and diffuse shading. The following problems are ordered by difficulty, and should be done in this order. As a general rule, feel free to expand on the shaders and rendering methods to make the terrain look more interesting.



(a) Cartoon Teapot



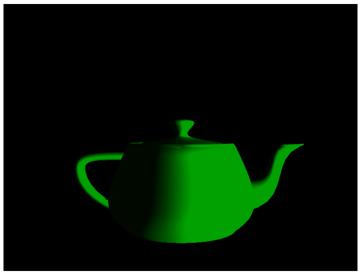
- the angle between the surface normal and the incoming light is the *angle of incidence* (here θ)

$$I_{diffuse} = k_d \, I_{light} \cos \theta$$

- in practice we use vector arithmetic:

$$I_{diffuse} = k_d \, I_{light} \, (n \bullet l)$$

(b) Diffuse lighting model



(c) Diffuse Teapot

Problem 1 (low difficulty)

1. Inspect the new way of loading in vertices in the `loadTeapot` function and run the program. Then, implement a cartoon-style shader for the teapot. Do this by applying some type of rounding to the end result. Use the passed in `light_intensity` variable as the light in the fragment shader. An example is shown in Figure 1a.

2. Make the "A" key toggle between cartoon and default shader.

Problem 2 (medium difficulty)

1. Calculate correct angle of incidence according to the formula in Figure 1b. Implement diffuse lighting for the teapot. If done correctly, you should see Figure 1c.

2. Make the "L" key toggle between diffuse and original shader. The cartoon and diffuse shader may not be enabled at the same time.

Problem 3 (high difficulty)

In the `update` function, add code that adjusts the `light_position` and `light_intensity` vectors responding to input variate, e.g., time.

- For medium credits you can implement a fixed position lighting whose intensity will change over "time in a day". (medium difficulty)

- For full credits you can simulate a full lighting change over "time of a day". You can consider such change from the position and intensity of light along time, e.g., the strongest light is at noon and the sun is beating down on the top of mountain. (high difficulty)

Tips: in order to deliver time variate to vertex shader, you could call: `glUniform1f(glGetUniformLocation(terrainshader.getProgram(), "x"), x)` API in the `render()` function.

Submission

The deadline is at the end of class, but if necessary, they may be submitted by 6pm of the day of the Workshop. Place in a ZIP file the following and submit on the LML Course Manager:

The top level of the zip file should contain a directory called **firstname.lastname.project** as described below:

In a directory called "firstname.lastname.project" (e.g. mary.smith.project)
(1) a file named "AnswerJournal.txt" which should list
- Your full name and student ID
- Mention which of the problems you solved.

(2) The source code, Makefile (the project must compile using "make") and
(3) Working executable of your solution