

# Homework 2: Modeling + Ambient & Diffuse Lighting

**Due date:** see LML Course Manager

## Goals

- Understand the fundamentals of creating a 3D model from Quads and Triangles
- Understand ambient and diffuse lighting and implement in OpenGL
- Understand OpenGL on Linux

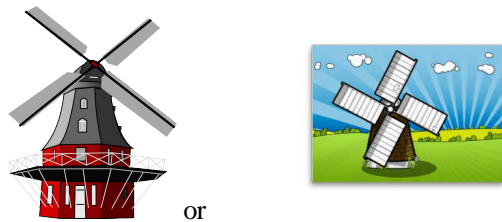
*No teams allowed for Homework 2*

All source code for Homework 2 should be written to compile and work on Linux.

Feel free to ask the teaching assistant for help.

### Problem 1 (low difficulty):

Using your solution to the first Workshop, create a *3D model of a windmill* on the ground to the right of the center of the ground using only `glBegin(GL_QUADS)` or `glBegin(GL_TRIANGLE_STRIP)`. Very roughly, it should look like one of these as viewed from the front:



or

To be specific, the goal of this problem is to ensure that you feel comfortable creating 3D models with polygons at a fundamental level. Thus, using functions like `glutSolidCube`, `glutSolidSphere`, etc. are not allowed for Problem 1.

### Problem 2 (medium difficulty):

(a) In your solution to Workshop 1, change the particles to spheres using the function `glutSolidSphere`. Note:

```
void glutSolidSphere(GLdouble radius, GLint slices, GLint stacks);
```

radius - The radius of the sphere.

slices - The number of subdivisions around the Z axis (similar to lines of longitude).

stacks - The number of subdivisions along the Z axis (similar to lines of latitude).

(b) Add to your solution from Workshop 1 the exact same lighting as in the lighting example code shown in class. This means that all the particles will now be spheres and will show ambient/diffuse/specular lighting as was done in the lighting example code using the 'a', 'd' and 'm' keys. Also, the windmill should be lit with the corresponding lighting of course.

### Problem 3 (high difficulty):

(a) Make the windmill blades rotate like a normal windmill. Toggle the blades spinning on/off using 'b'

(b) When one of the particles hits the windmill, make it explode. Note that a basic implementation will earn a sufficient grade, but the more elaborate the better (The more realistic, the better.). A basic implementation example for the windmill would be detecting if the particles are fairly close to it (e.g. hitbox). *It is necessary to implement these Keyboard*

*Assignments:*

```
case 's': // up (viewer is moving up - this used to be the 'a' key)
case 'x': // down (viewer is moving down - this used to be the 'z' key)
case 'j': // left (viewer is moving left)
case 'l': // right (viewer is moving right)
case 'i': // forwards (viewer is moving forwards)
case 'k': // backwards (viewer is moving backwards)
case 'f': // fire (fire the particle cannon)
case 'g': // toggle the gravity key on/off
case 'd': //toggle diffuse lighting
case 'a': //toggle ambient lighting
case 'm': //toggle material properties
case '4': //move the light up
case 'r': //move the light down
case 'e': //move the light left
case 't': //move the light right
case 'b': //toggle windmill blades spinning on or off
```

**GLUT Tip** - On some systems, if you hold a key down, it automatically does key repeat. On Linux, we had varying results. According to the documentation, it might be possible for Linux/Glut to have key repeat when you hold down a key. The function looks like this:

```
int glutSetKeyRepeat (int repeatMode );

repeatMode
    GLUT_KEY_REPEAT_OFF
    GLUT_KEY_REPEAT_ON
```

No guarantees - Your mileage may vary on this one!

## Submission Checklist

*Always check the LML Course Manager for the due date!*

*The program must compile and run on one of the machines in room 302 (or a LIACS student computer room)*

Place in a ZIP file the following and submit on the LML Course Manager. *The top level of the zip file should contain Journal.txt and a directory called project.firstname.lastname as described below :*

- (1) a file named "**Journal.txt**" which should list
  - The full names of the people who worked on the workshop with you.
  - The name of the machine you had it working on. e.g. 0009747, usually on the side of the machine.
  - Mention which of the problems you solved.

In a directory called "**project.firstname.lastname**" (eg. **project.johan.cruijff**)

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

*Good luck!*