City University of Hong Kong
Department of Electronic Engineering

Computer Graphics for Engineers
EE4208

OpenGL Mini-Project

Report

By

Tenna Boeriis Rasmussen

40096350

Table of Content

Introduction

The aim of this project is to create a computer graphics animation using the techniques taught in the course "Computer Graphics for Engineers (EE4208)".

As an exchange student I decided to use a project theme to represent my home in Denmark. I am born and raised in Billund, Denmark. Billund is a small town with a population of only 6194. It might have been insignificant if it didn't happen to be home of Denmark's, probably, best known company worldwide; LEGO.

Since the LEGO was founded in Billund on 10 August 1932 by Ole Kirk Christiansen, the toy brick has influenced the lives of many children, both local and worldwide, and as a child of Billund my life included. I have therefore chosen LEGO as the theme of my mini project.

The resulting LEGO-themed computer graphics animation builds a LEGO house. With user-interaction through the keyboard it is possible to choose the height of the LEGO house to be build.

Animation and User Interaction

When the program starts the user is prompted for input by the console. To choose the height and start the building of the LEGO house the user must select a number between 1 and 6 (both included) on the keyboard. Once the height has been selected the building begins in the animation window. If a new height is selected the animation starts over to build a house with the new height.

The house is built of mainly 8-stud bricks, but also 4-stud and 24-stud bricks are used. It is worth noticing that for every other floor the bricks are shifted so they cover the "gaps" of the layer below. This improves realism as in a real-world scenario this would keep the building more stable. The roof consists of five 24-stud bricks and there will always be a roof on the building. This means that if a height of 1 is selected the house will purely be made up of a layer of the five larger bricks. When building a house it is convenient to have a door, which this house of course also has. That is, if the height is greater than one. Upon animation each of the bricks drops from above to land in the specific place that will continue the building of the house. A finished house with a height set to 5 is seen in figure 1.

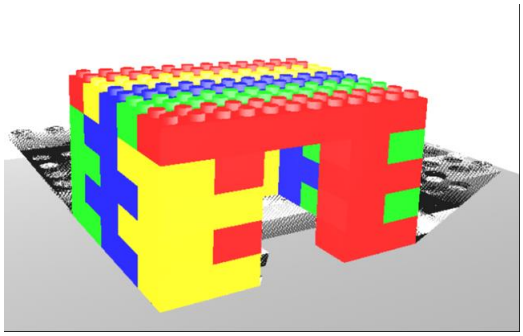

Figure 2 - Final state for the animation of the house with height 5.

LEGO Bricks

The LEGO bricks are built with a cube as the base of the brick and cylinders with disks as the studs on the bricks. The measurements of the 8-stud bricks are as seen on figure 3.

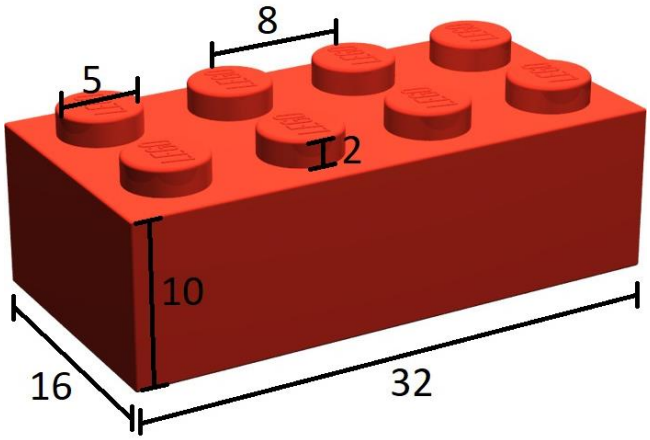

Figure 4 – LEGO brick modelling the measurements as used in the OpenGL project

To keep the code structured the functions to create the bricks are moved to their own set of header- and source-file called bricks.h and bricks.cpp. In table 1 the functions are described.

Function Prototype	Description
`void cube ( );`	Creates a standard 2 x 2 x 2 cube with the center at (0, 1, 0).
`void draw_8brick (double x, double y, double z, double theta, double rotX, double rotY, double rotZ);`	Creates an 8-stud brick with the position of (x,y,z) and a rotation of (theta, rotX, rotY, rotZ).
`void draw_4brick (double x, double y, double z, double theta, double rotX, double rotY, double rotZ);`	Creates an 4-stud brick with the position of (x,y,z) and a rotation of (theta, rotX, rotY, rotZ).
`void draw_24brick (double x, double y, double z, double theta, double rotX, double rotY, double rotZ);`	Creates an 24-stud brick with the position of (x,y,z) and a rotation of (theta, rotX, rotY, rotZ).
`void set_brick_material_properties (void);`	Sets the material properties of a brick.

Table 2 - Function descriptions for functions of brick.h

LEGO House

The LEGO house of the project is built using the bricks of bricks.h. The functions to create the LEGO house are defined in the header-file legoHouse.h and implemented in the source-file legoHouse.cpp. The files contains the build_lego_house()-function which animates the building of a house with the user-specified height. Two different methods of brick placement are used for the layers comprising the walls of the house. The two layers are interchanged to create the effect of shifted bricks as it is seen on figure 5.

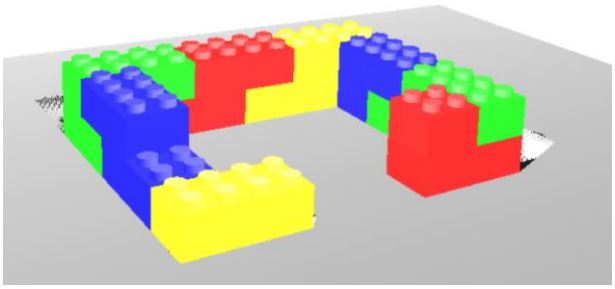

Figure 6 – The bricks lie in a shifted pattern.

Each layer of the wall and the roof has a fixed time set for the animation of the layer. The time is defined in the value build_time and set to 10000 ms. In each layer this time is divided into the number of bricks of the layer to define the animation time for each brick. The translation of the bricks use a cosine function to create acceleration. To make the acceleration sharper, but still without increasing the speed too much at the end, an exponent of 0.7 is used. A function to set different colors for the bricks is also created. In table 3 the functions of bricks.h are described.

Function Prototype	Description
`GLfloat* colorRotation(int num);`	Sets the color based on the input parameter.
`void buildOddFloor(GLdouble elapsedTime, int floorLevel);`	Animates and builds a layer of bricks for the wall using 9 bricks two of which are 4-stud bricks.
`void buildEvenFloor(GLdouble elapsedTime, int floorLevel);`	Animates and builds a layer of bricks for the wall using 8 bricks all of which are 8-stud bricks.
`void buildRoof(GLdouble elapsedTime, int floorLevel);`	Animates and builds a layer of bricks that make up the roof using 5 24-stud bricks.

`void build_lego_house(int floors, GLdouble elapsedTime, double t_prev);`	Builds the LEGO house with the user-defined height.

Table 4 - Function descriptions for functions of legoHouse.h

Shadows

Perspective projection is used to create the shadows on the ground. The object color is set to black. A coordinate system transformation is used to transform a light source s to the PRP and the z-x plane to the image plane. Then the brick is projected onto the image plane creating the shadows. A function for each of the brick types are created. They each take arguments to set the translation and rotation of the object, that casts the shadow. The shadow-functions are described in table 5.

Function Prototype	Description
`void generate_shadow(double x, double y, double z, double theta, double rotX, double rotY, double rotZ);`	Draws the shadow of the 8-stud brick.
`void generate_shadow_4brick(double x, double y, double z, double theta, double rotX, double rotY, double rotZ);`	Draws the shadow of the 4-stud brick.
`void generate_shadow_24brick(double x, double y, double z, double theta, double rotX, double rotY, double rotZ);`	Draws the shadow of the 24-stud brick.

Table 6 - Function descriptions for the functions of shadow.h

Floor

The floor of the scene is created by the only function of the files floor.h and floor.cpp. In the method the material properties are set to make the floor grey and less shiny than the bricks. The function is described in table 7.

Function Prototype	Description
`void draw_floor( void );`	Draws the grey floor of the scene.

Table 8 - Function description for the function of floor.h

Main function and initial setup

In Main.h the function drawscene sets values for lighting, depth buffering and colors of the animation window. Functions to update the animation and react to user input are also included in Main.h. Finally the main-function itself is also placed in Main.h. To animate the project the animate-function is set as the glutIdleFunc. The animate function ensures that that the elapsed time is updated. The elapsed time is the basis for controlling the timing of the animation. For the glutKeyboardFunc the selection-function is set. The selection function defines the height of the LEGO house based on user input. The selection-function also updated the variable t_prev, which marks the time of the user input. t_prev is used to define the start time of the animation and thus enables the animation to start over every time the user selects a new height of the house. In table 9 the prototypes and description of the functions in Main.h can been seen.

Function Prototype	Description
`void drawscene (void);`	Sets values for lighting, depth buffering, and the animation window. drawscene also uses the build_lego_house function from legoHouse.h to draw the actual house.

`void animate(void);`	Reads the elapsed time into the global variable t and calls glutPostRedisplay to ensure the scene is redrawn with the new updates.
`void selection (GLubyte key, GLint xMouse, GLint yMouse);`	Sets the height of the house based on the user-keayboard-input and sets t_prev used to mark the time for the beginning of the animation.
`void main (int argc, char** argv);`	Initiates the program.

Table 10 - Function description for the function of Main.h

Conclusion

By using the techniques of the course a LEGO-themed computer graphics animation with user interaction has been created. Hierarchal structures have been created to shape both the individual bricks, but also the whole LEGO house which is built from the bricks. The building of the house is animated to mimic the movements involved with building structures from LEGO bricks in real world scenarios. The shadows of the brick are projected according to a specific light source and the changing position of the bricks. Furthermore, the shading of the elements of the scene is made with gourad-shading to give the objects of the scene a smooth look, although with different properties for different materials to make the brick more shiny and the floor less. All in all the resulting project is a fun interactive animation, that utilizes the computer graphics techniques to bring out memories of childhood times.

Declaration of Authenticity

I declare that every line of code is written by myself and I have not committed any plagiarism.

Signed

_____

Tenna Boeriis Rasmussen