Middle East Technical University | Department of Computer Engineering

CENG 477

Introduction to Computer Graphics

Fall '2022-2023
Assignment 3 - OpenGL with Programmable Shaders

Due date: January 21, 2023, Saturday, 23:55

1 Objectives

In this homework assignment, you will be creating a game using OpenGL called "Bunny Crush". In this game, the game board will be presented with a grid of bunnies, each with a different color. If there are three or more bunnies of the same color in a row or column, they will explode and new random color bunnies will slide in from the top to replace them. The player can click on bunnies to manually cause them to explode, and the game will keep track of the number of moves the player has made. The player's score will only be counted for bunnies that are exploded as part of a match. The goal of the game is to earn as many points as possible by matching as many sets of bunnies as possible.

Keywords: *OpenGL, programmable shaders, crush game*

2 Specifications

1. The executable file for this assignment will be named "hw3" with a window size of 640×600. This is the file that you will run to start the game. The program gets grid shape and the object file name as command line argument.

2. The game board will have a grid size given as command line argument. For example, "10 5" means that the game board will be a 10x5 grid of squares, each containing an object.

3. Your program will parse the object file and use this information to create the objects for the game. The object file contains information about the shape and appearance of the object, such as the vertices and faces that make up the object's 3D model. Your program will use this information to create a 3D model of the object that can be displayed in the game.

4. You will resize the objects considering the grid size to fit into the window properly.

5. The scene will be rendered using an orthographic projection, with the specified left, right, bottom, top, near, and far parameters of (-10, 10, -10, 10, -20, 20). Orthographic projection is a way of representing 3D objects on a 2D surface, where the size and shape of the objects are not affected by their distance from the viewer. The left, right, bottom, top, near, and far parameters specify the dimensions of the viewing volume, or the region of 3D space that is visible in the game.

6. At each frame of the game, the objects will rotate around the vector (0, 1, 0) at an angle of 0.5 degrees. A frame is a single image displayed by the game. The vector (0, 1, 0) specifies an axis of rotation, and the angle of 0.5 degrees specifies the amount of rotation that will occur around this axis at each frame. This will cause the objects to rotate slightly at each frame of the game.

7. If there are more than three objects of the same color in a row or column, they will explode and the objects above them will slide down to fill the empty space. For example, if there are four blue objects in a row, they will all explode and the objects above them will slide down to fill the empty spaces.

8. New random color objects will be added at the top of the grid and will slide down to their correct positions using an animation in which the y position of the bunny object is decreased by 0.05 at each frame. When a bunny explodes and leaves an empty space, a new bunny will be added at the top of the grid. The new bunny will start at the top of the grid and slide down to its correct position using an animation in which the y position of the bunny object is decreased by 0.05 at each frame.

9. If a bunny explodes, it will be scaled up by 0.01 in all directions at each frame until it reaches 1.5 times its original size, at which point it will be deleted. This will create an animation in which the exploding bunny grows in size until it reaches 1.5 times its original size, at which point it will be removed from the game.

10. The player can interact with the game using the mouse to manually explode bunnies and change the arrangement of the bunnies on the grid. For example, the player can click on a bunny to manually cause it to explode and match with other bunnies of the same color.

11. Mouse interaction and color matching will be disabled until all animations have completed. This means that the player will not be able to interact with the game using the mouse or the game won't match new objects until all animations, such as the slide animation for new bunnies and the explosion animation for matching bunnies, have finished.

12. Each object in the game will have its own point light source, located at the same x and y coordinates as the object and at a z coordinate of 1. The position of the light will move together with the object during the animation. A point light is a light source that emits light in all directions from a single point. By giving each object its own point light source, you can create a more realistic lighting effect in the game. In order to get same view for each of the objects, you will accept as eye position is at the same point with the light source.

13. The game will keep track of the number of moves made by the player, as well as the number of matched objects and the score earned. The number of moves is the number of times the player has manually exploded a bunny. The number of matched objects is the total number

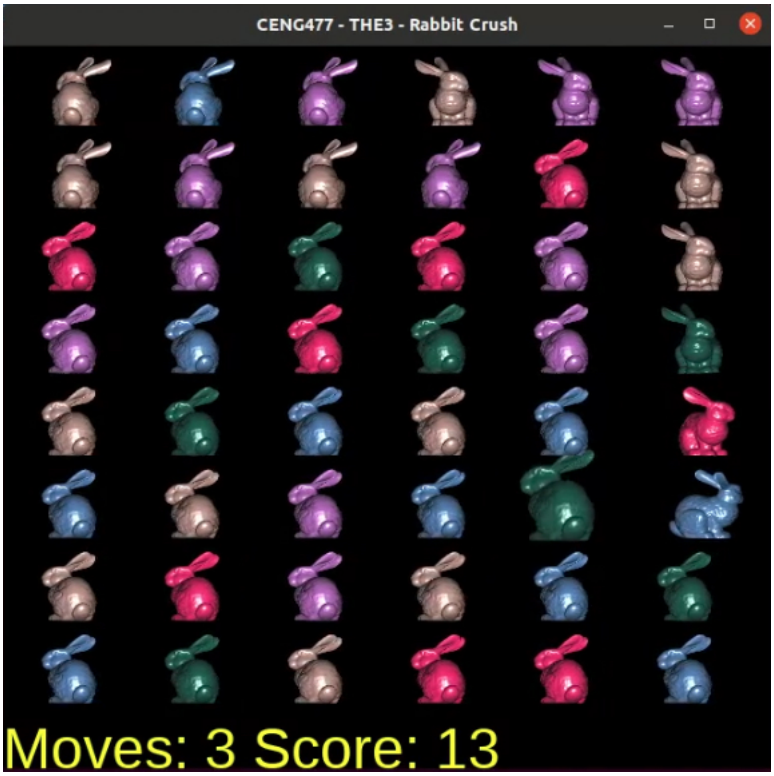
of objects that have exploded as part of a match. The score is the total number of points earned by the player.

14. These values will be displayed at the bottom of the window. The number of moves, the number of matched objects, and the score will all be displayed at the bottom of the game window, so that the player can see their progress as they play.

15. If the user presses the 'R' key, the game will restart with a new random arrangement of objects. If the user presses the 'ESC' key, the game will close. This will allow the player to easily restart the game or exit the game if they wish.

3 Sample input/output

The sample gameplay video is given at OdtuClass. You can find the screenshot below.



4 Regulations

1. **Programming Language:** C++

2. **Late Submission:** You can submit your codes up to 3 days late. Each late day will be deducted from the total 7 credits for the semester. However, if you fail to submit even after 3 days, you will get 0 regardless of how many late credits you may have left. If you submit late and still get zero, you cannot claim back your late days.

3. **Cheating: We have zero tolerance policy for cheating**. People involved in cheating will be punished according to the university regulations and will get 0. You can discuss algorithmic choices, but sharing code between each other or using third party code is strictly forbidden. To prevent cheating in this homework, we also compare your codes with online sources and previous years' student solutions. In case a match is found, this will also be considered as cheating. Even if you take a "part" of the code from somewhere/somebody else - this is also cheating. Please be aware that there are "very advanced tools" that detect if two codes are similar. So please do not think you can get away with by changing a code obtained from another source.

4. **Newsgroup:** You must follow the discourse (odtuclass.metu.edu.tr) for discussions and possible updates on a daily basis.

5. **Submission:** Submission will be done via OdtuClass. You can team-up with another student. You will provide a make file to build your homework. Be sure, before submitting, it is running on **Inek** machines. It will not produce any outputs since an interactive OpenGL display window will be used. Create a .zip file that contains your **source files and Makefile**. If your directory structure is wrong or makefile does not work (or you do not have one) and therefore we will have to manually try to compile your code - there will be an automatic **penalty of 10 points** for each. The .zip file should not include any subdirectories. **The .zip file name will be:**

   - If a student works with a partner student:

     `<partner_1_student_id>_<partner_2_student_id>_opengl.zip`

   - If a student works alone:

     `<student_id>_opengl.zip`

   - For example:

     ```
     e1234567_e2345678_opengl.zip
     e1234567_opengl.zip
     ```

   Make sure that when below command is executed, your executable file is ready to use:

   ```
   >_ unzip e1234567_opengl.zip -d e1234567_opengl
   >_ cd e1234567_opengl
   >_ make hw3
   >_ ./hw3 <grid_width> <grid_height> <object_file>
   ```

   Therefore you HAVE TO provide a Makefile in your submissions.

6. **Evaluation:** Your codes will be evaluated manually. You are responsible to implement everything mentioned in the text. You can add new features or change the view such as color. We will evaluate your results visually. Therefore, if you have subtle differences (numerically

different but visually imperceivable) when running the program, that will not be a problem. Allowed libraries other than standart libraries for the assignment are only glew, glfw, glm, . Using any other library is strictly forbidden. Homeworks implemented with GLUT will be ignored and not be graded. Read the specifications carefully. Anything that is conflicting with specifications will make you lose point/s.