

Tools Programming

CMP405

Lewis Franklin

1501145

Table of Contents

[**Summary of Features**: 3](#_Toc7539733)

[**User Controls:** 3](#_Toc7539734)

[**Features:** 4](#_Toc7539735)

[Camera 4](#_Toc7539736)

[Movement 4](#_Toc7539737)

[Focusing: 4](#_Toc7539738)

[Arc ball: 4](#_Toc7539739)

[Objects: 5](#_Toc7539740)

[Selection: 5](#_Toc7539741)

[Creation: 5](#_Toc7539742)

[Transforming: 6](#_Toc7539743)

[Texture selection: 6](#_Toc7539744)

[Terrain: 6](#_Toc7539745)

[Conclusion: 7](#_Toc7539746)

[References: 8](#_Toc7539747)

# **Summary of Features**:

This tool includes the ability to control the camera with either the keyboard or the mouse all camera rotations and movement can be done with only the keyboard or with the help of the mouse. The program also includes the ability to focus on a selected object on a key press and orbit around that object with another key press. The terrain can be modified by importing a raw file height map with the use of a file selection window, there is also the ability to make the terrain wireframe to be able to see through it. There is the ability to manipulate the objects in the scene, the variables that can be changed are the position x,y,z, the rotation and scale. Other options include changing the texture of the object, copying and pasting of the object and loading in of a completely new object. The ability to delete objects is also included. The user can easily identify which object is being selected and currently manipulated as the object is turned to wireframe upon selection.

# **User Controls:**

The user can control the program just like any other pre made game engine or editor. The basic camera controls are; W to move the camera forward, S to move it backwards, A to move it to the left, D to move to the Right. The camera can also rotate using Q and E rotating to the left and right respectively. If the user presses the space bar the mouse cursor will disappear to display to the user that the camera is now being controlled with the mouse. Once the camera is being controlled by the mouse the user can now rotate up, down, left and right just like a camera in any game or game engine. If the user is not using the mouse for camera controls they can select objects in the scene with the left mouse button. The user can also quick copy objects with the control and c keys pressed at the same time, they can easily paste that object by pressing the control and v keys. The user can also delete the selected object by pressing the delete key. If the user does not wish to modify the objects they can chose to either just focus on the selected object orbit the object or use an arc ball camera mode to rotate around it. To any of these the object must be first selected then press the F key to focus the camera on the object. The user can then either press the O key to orbit the subject or press space which again removes the cursor from the screen and uses the mouse to rotate around the object. On the main tool bar beside the save objects button is a button for wireframe mode for the terrain.

# **Features:**

## Camera

One of the main areas where features were the most important was the camera. The camera is the most important as this is how a user interacts with the scene and moves around to see where everything is that they may be changing. With this in mind the first thing to focus on was the movement of the camera.

### Movement

This feature is one of the most simple but powerful part of the tool. The movement of the camera is determined with either just keyboard controls or the addition of the mouse. This allows the user to easily move around the scene in a simple and intuitive manner.

The method of doing this in code includes capturing the amount the mouse has moved from one frame to the next. In the program the mouse is locked to the centre of the program and the position relative to that is calculated and used every frame. These values are then passed to a camera class that uses Euler maths to calculate a pitch and yaw. Code for the x, y and z movement of the camera includes a forward vector and a right vector these are calculated using the yaw and pitch along with a cross product of the forward vector respectively.

### Focusing:

The features of the camera includes a focusing to an object mode. This is done by selecting an object and pressing the F key. What this mode does is centre the selected object to the middle of the screen allowing the user to get a closer look at what they have selected instead of using the free camera to get closer to the object.

In code this is done by setting the look at matrix in the camera class as a static matrix and setting that to the position of the selected object. The position of the camera is simply the position of the object with an offset setting the camera up in the air.

**Orbit:**

The orbit function does exactly that, whilst holding the O key it simply begins an orbital rotation of the selected object. After releasing the O key the camera jumps back to where it began.

This like the focus feature in code simply keeps a static look at matrix but slowly increases a rotation matrix on the z and x axis whilst still maintaining an offset of the camera in the air above the object.

### Arc ball:

Arc Ball mode on the camera allows the user to move around the selected object with a degree of precision. The user can control the camera with the mouse whilst focused on the object. This improves the user experience when trying accurately place an object in the scene, a user can place an object and look around the object easily to see if it is colliding with anything else in a scene.

In the program this is coded by calculating the mouse movement, like in the free camera, and doing calculations for a rotation on the x and z axis like in the orbit camera. This function basically combines the two together to get a more precise orbital camera that moves the camera on more than just one plane and in a direction the user desires.

## Objects:

The objects in the scene have many useful features attached to them from making it easy for the user to see what they have selected to intuitive mechanics for the user to replicate and altered object.

### Selection:

A feature of the tool is the improved user experience when selecting an object. When the user selects an object the object becomes wire framed to display easily which one has been selected. The decision to make it wire framed comes down to being able to see if any object is colliding with it without having to focus and either orbit or arc ball the camera or using the free camera to go around all sides. Though this is not helpful when trying to change the texture and be able to see it easily the user can click off of the object to get the object to become full again.

The code for this is simple, the program gets the selected ID of the object and when recalling the display list makes an exception for the selected ID object to draw in wireframe mode by changing a boolean value.

### Creation:

In keeping with the current object being see through this is helpful with creating a new instance of that same object. This is done by a simple copy and paste which as suggested copies the selected object and pastes in it exactly the same place. Another way of making new objects is the use of a button in the edit drop down list on the top tool bar. This option called new model allows the user to add a new cmo file to the scene. Cmo file is the default model file that was included in the base application. When a new model is loaded in the values for position, rotation and scale are all set to zeros apart from scale which is set to 1, this puts the model right in the middle of the scene which is hopefully easy for the user to see. If a model is made that is not wanted by the user they can simply delete the object by selecting it and pressing the delete key on the keyboard.

In the code for the copy and paste is simple with only a few lines first creating a pointer to the selected object than adding that object again to the vector of scene objects. For the new model loading a dialogue window is created on the selection of the new model button which prompts the user for a file path to the object to be added. This then adds one to the vector of objects and sets the file path, position, rotation and scale. The user can then save this using the save objects button which saves that given file path to the database. Unfortunately if the file path is changed the object will no longer show up in the program. The delete code is simply if the player presses the required key that selected object is deleted from the scene object vector.

### Transforming:

Another feature of the object selection is the possibility to transform the position, rotation and scale of the objects. This is done by the user first selecting an object they wish to transform then going to the edit drop down menu and selecting transform. By doing this a dialogue window pops up with edit boxes that contain the current values set for the object, the user can then change these if they wish and the object will update in real time.

The code for this is simply passing in the current selected object into a transform window class, this class then holds the object for editing. The edit boxes then output the current values to the user, when the user changed these values a windows update data function is called which then writes the inputted values into the variables of the held object. This newly updated object is then passed back out to tool main where it replaces the current object in the object vector. The function Build display list is then called to visually update the object in the scene.

### Texture selection:

In the same Transform window the user can select a button that can change the texture of the selected object. This button prompts the user with a file path selection which is filtered to a dds texture file format. The user can then select the desired texture and the object texture will then change to that in scene.

This is done by simply using the windows standard file path selection (Microsoft, 2017) and changing the scene object texture path to the newly selected file path. A string conversion is also needed to change from CString to string (StackOverFlow, 2008) once that is passed through a call to build display list is made and the object is updated.

## Terrain:

Height map able

Wireframe able

# Conclusion:

Provide reflection and critical analysis of your work. What went right what went wrong and why? What would you do differently next time?

# References:

Microsoft (2017) *File loading.* Available at: <https://docs.microsoft.com/en-us/cpp/mfc/reference/cfiledialog-class?view=vs-2019#cfiledialog> (accessed: 25th April 2019)

StackOverFlow (2008) *CString to String conversion. Available at:* <https://stackoverflow.com/questions/258050/how-to-convert-cstring-and-stdstring-stdwstring-to-each-other> (accessed: 25th April)