a1model-170050077_170050081

170050077 -- Nama N V S S Hari Krishna
170050081 -- Doddi Sailendra Bathi Babu

Note: Most of the boilerplate code structure like .glsl files, most of the .cpp, .hpp files are taken from tutorial 3(rotating a color cube)

Initializing buffers:

In `initBuffersGL()` created the required points for grid along with colors and stored them into points, colors array respectively. These points are created and stored in the array whilst maintaining the order of points(since glDrawArray picks the points one by one in order from the array) and assigned a light grey color to all the points. Similarly created the required points for the cursor cube and stored them into an array and assigned a green color to all the points.

Maintained an array of vaos and vbos with size 3 for the grid, cursor cube, and model. Now we bind the vaos and vbos using the points stored and the shader attributes(vPosition and vColor).

Rendering:

In `renderGL()` first, we call glClear to clear the content displayed in the window and then created the rotational and translation matrices and multiplied with the ortho matrix to give Model View matrix. Next, sequentially we bound each vao and called glDrawArrays with GL_LINES for the grid, GL_TRIANGLES for cursor cube and models

The new set of points and colors are copied to the buffers corresponding to the models if there is any change with respect to the previous points in the buffers. This is done by maintaining two vectors one for points and another for the colors of the models. The data from these vectors are copied to the arrays and then loaded into buffers. These two vectors are modified in the `key_callback()` function.

Key Callbacks:

In the key_callback() function in gl_framework.cpp we deal with various keyboard inputs.

- Keys M and I switch the current mode of the program by changing a global variable.
- H key switches the current grid-style among the full grid or without the lines in the middle done by Buffer data corresponding to the grid model.
- Keys up/down/left/right/'['/']' change the values of variables corresponding to the rotation angle which are used in creating the rotation matrix in `renderGL().`
- For functions which work only in the modelling mode:
  - Keys x/y/z/X/Y/Z change the values of variables corresponding to the cursor-cube position which are used in creating the translation matrix in `renderGL().`

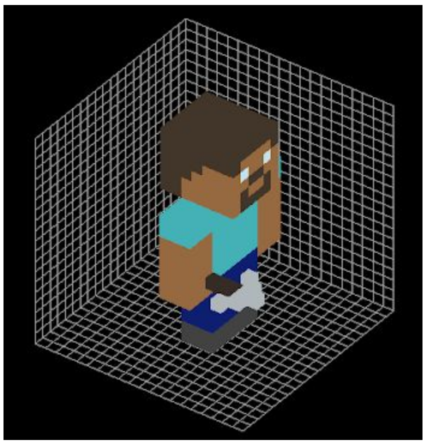
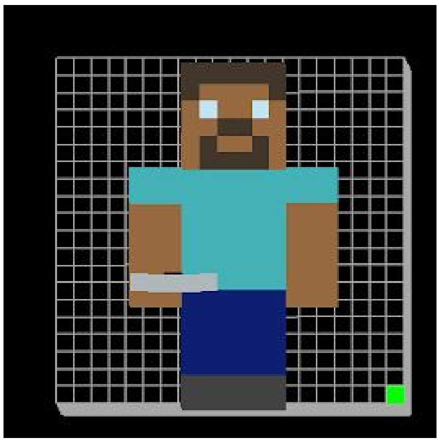
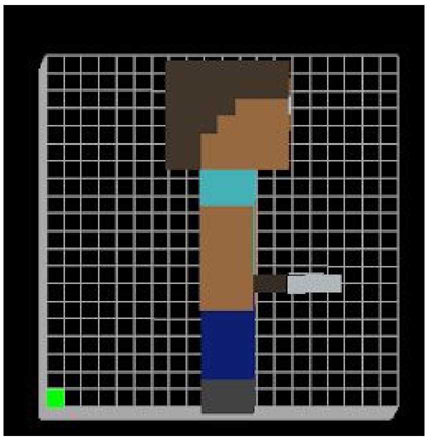
- P key stores the current position of the cursor cube and the current color in two vectors. These vectors store the points and colors of the model.
- C key sets the current drawing color to the value taken from the command line prompt.
- D key deletes the points and color entry corresponding to the current cursor position in the two vectors.
- R key empties the above two vectors thereby resetting the grid.
- S key saves the data in the above two vectors into a file in the data folder in the specified format. The file name is taken through command line prompt.
- K key loads the data from the file into the above two vectors. The file path is taken through the command line prompt.

Main function:

The main function is the same as that given in tutorial 3 except that we added a few lines for loading the shader files.

Models generated:

Model01:

Model02: