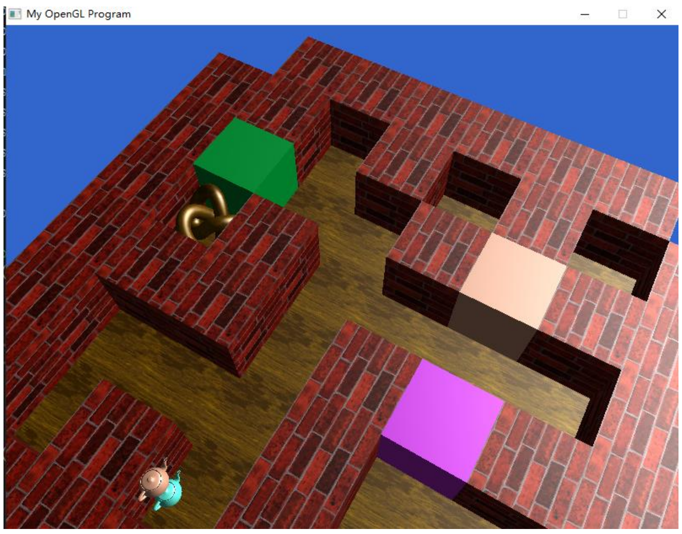
In project 4, I implemented all the required features and some extra features:
(5) Texture map the walls and floors (use different textures for different models)
(5) Jumping (must be smooth & realistic for full credit)
(5) Procedurally generate levels (there should always be a solution)
(5) Make a video of yourself playing, showcasing your features
(5) Two artistic/well-composed screenshots from your game
(5) Load existing models in the OBJ 3d format: http://enwp.org/Wavefront_.obj_file It's okay to support only some of the model features (e.g., only triangles)

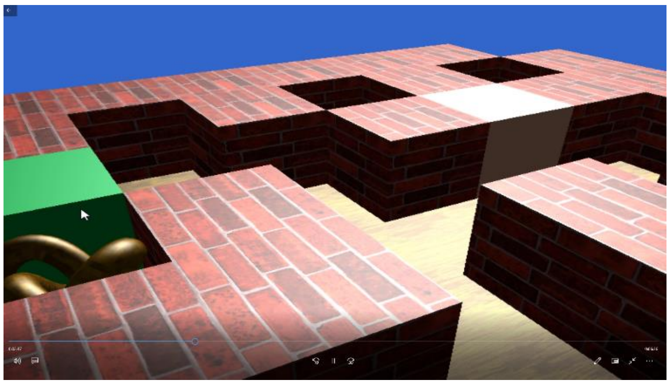
More demonstrations could be found in the video I uploaded.
Texture map: I loaded a few textures, but initially the extra texture doesn't work. Later on, I found that there are some codes needed to modify in texturered-fragment and textured-vertex file.

Jumping: I used free fall equation: y = 1/2*a*t*t+v0*t+x0, so it will look realistic. But the player keeps looking forward will make the jumping meaningless. So I set the player will look down a little bit when the player jumped. Then the player can see the map nearby.

Generate levels: More details were explained at the end of the video

Artistic pictures: These two was taken in the air, giving an overview of the whole scene





Loading obj: I tried working on this feature. But it doesn't work finally. I checked out the file format of the obj file(vertex, uv coordinate, normal, face). I transfer all the obj data into the txt format we used all the time. But unfortunately, it doesn't work.

Other issues I met:
The collision seems hard to implement. I tried to adjust the camera position, but sometimes the camera will pass through the walls, or see the inside of walls. I thought it could be solved by separate player position and camera position.

Parsing the map file took a while, but a lot of code could be copied from the previous project.

For the user interface, I initially tried to work on integrated all the inputs (camera direction changed when the mouse slide and the keyboard changed the camera direction). This method is used by common video games. But I can't make the mouse SDL API works, and there didn't exist a solution on the internet. So, I come back to an easy solution.