CS450/550 Fall 2020(INTRODUCTION TO COMPUTER GRAPHICS)

NAME : Si Thu Lin
EMAIL : linsi@oregonstate.edu
PROJECT NAME : Final Project
VIDEO : https://media.oregonstate.edu/media/1_7b4v1s7m

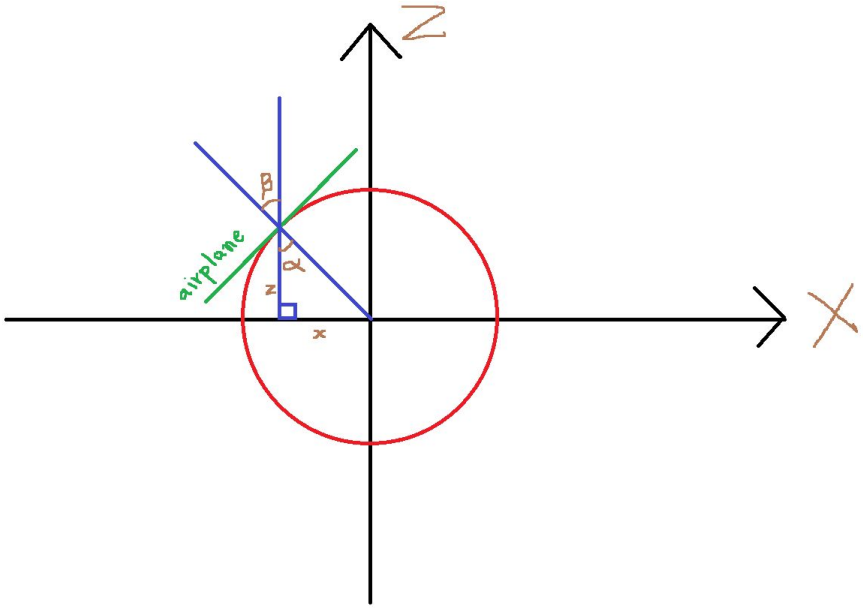
My final project proposal is
"I am planning to make an airplane or helicopter fly around and finally hit a target. When the airplane or the helicopter hits the target, the target will be broken into particles. To change the target into particles, I will use vertex and fragment shader. I am thinking to use the function used to draw the sphere in the previous projects in the way that I will use GL_POINTS instead of GL_QUADS. I will use some lighting in my final project. I might also use texturing for the target to be hit.".

The steps of the project

Professor Bailey replied to me to use an airplane instead of a helicopter since the helicopter has been used in the class. In this final project, I make the airplane to fly in the circular direction. When the airplane hits the sphere, the sphere will be broken into particles and the particles will fall. To do that, I use two sphere functions. One is a solid sphere which is formed with quads and can be seen as soon as my code is executed. The other is the sphere which is formed with points. In other words, I used GL_QUADS to draw the former sphere and GL_POINTS to draw the latter one. Initially, the solid sphere will be used. However, when the airplane hits the sphere, the sphere which is drawn with points will be used instead. To make the particles fall, I use Vertex and fragment Shaders. To make the falling of particles realistic, different velocity values are used for the particles. So, velocity values for each vertices are assigned randomly at the beginning of the program.

I use two equations from the CS475/575(INTRO TO PARALLEL PROGRAMMING) class. They are supposed to be used for the project 7(a) at that time. However, I choose 7(b) for the last project. So, I did not have the chance to use them. One of them is pp = p + v*DT + G*(.5*DT*DT) equation where p is the current vertex position, pp is the new vertex position, v is the velocity value, DT is the time and G is the gravity. The other is vp = v + G*DT where vp is the new velocity value, v is the current velocity value, G is the gravity and DT is the time. I calculate the new velocity and new vertex position for each vertex ,and send the corresponding new vertex position for each vertex to the vertex shader using the attribute variable.

I also make the airplane fly realistically. To do so, I make sure that the airplane turns its head to the direction in which it is flying.



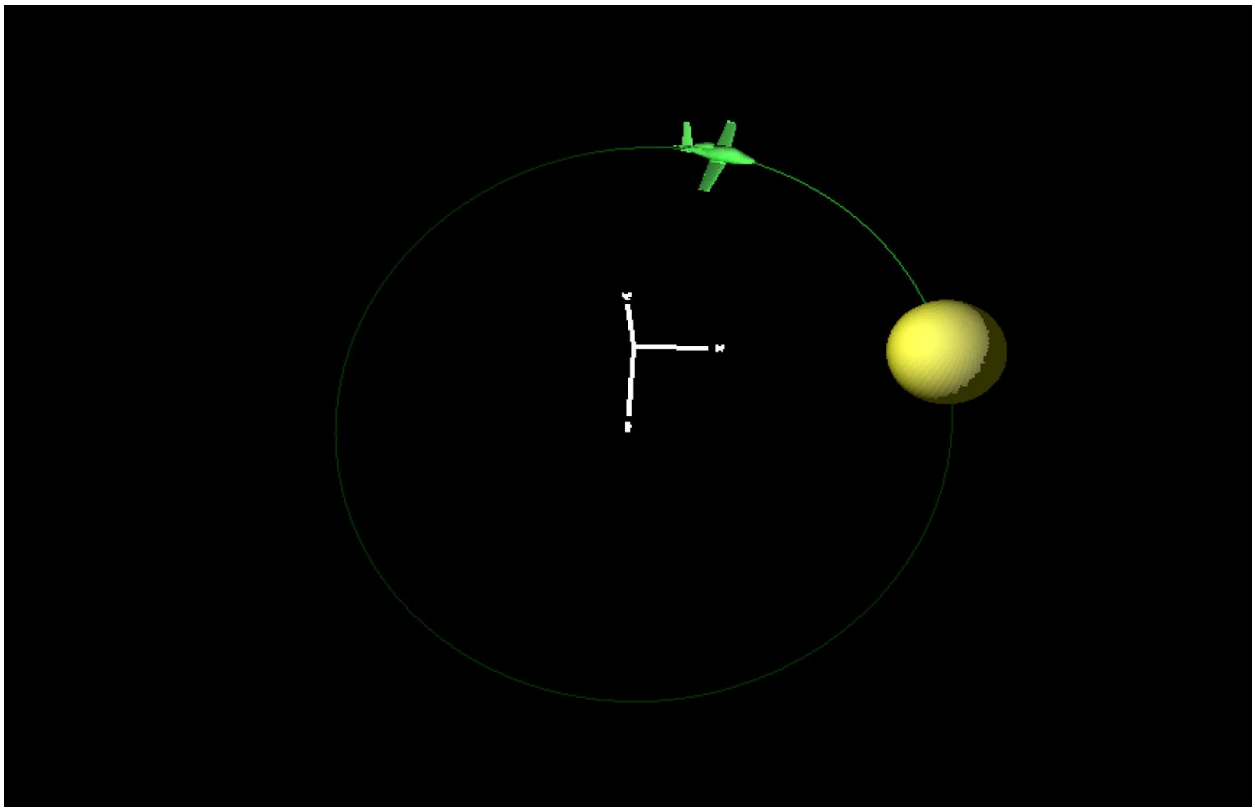
In this figure, the green line represents the airplane and the red circle the route the airplane uses to fly. To rotate the airplane correctly, I need beta value. From the alpha value, I can get beta value. To get the alpha value, I use tangent(alpha)=opposite side/ adjacent side equation which is also equal to alpha=tangent inverse(opposite side/ adjacent side). I can get the sides value from the position of the airplane. There is atan() in C++ to get the tangent inverse value. When I get the alpha value, I need to convert the value to degree since the result value is in radian. I multiply the result alpha value with (180./PI) to the degree. With a little modification to the result degree, I can rotate the airplane correctly.

After the particles fall, a new sphere will appear again to be hit by the airplane again. The new sphere will not appear awkwardly. A new sphere will appear as a small one and it will be bigger gradually until the size is big enough. In this final project, lighting is used as well. Shader also uses lighting.

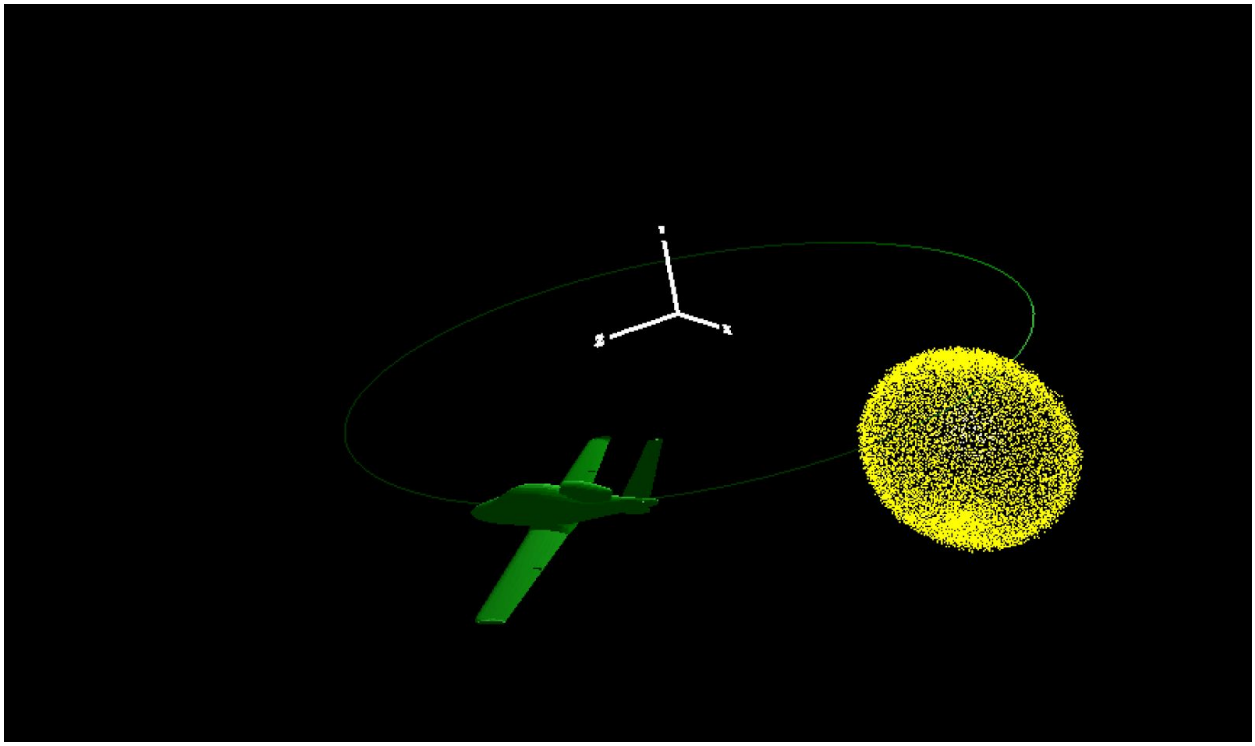
In my final project proposal, I said “I might also use texturing for the target to be hit.”. I did not guarantee that I would use the texture. Since I believe that I have put enough effort and time in my final project and the result is already beautiful, I have not included any texture in my final project.

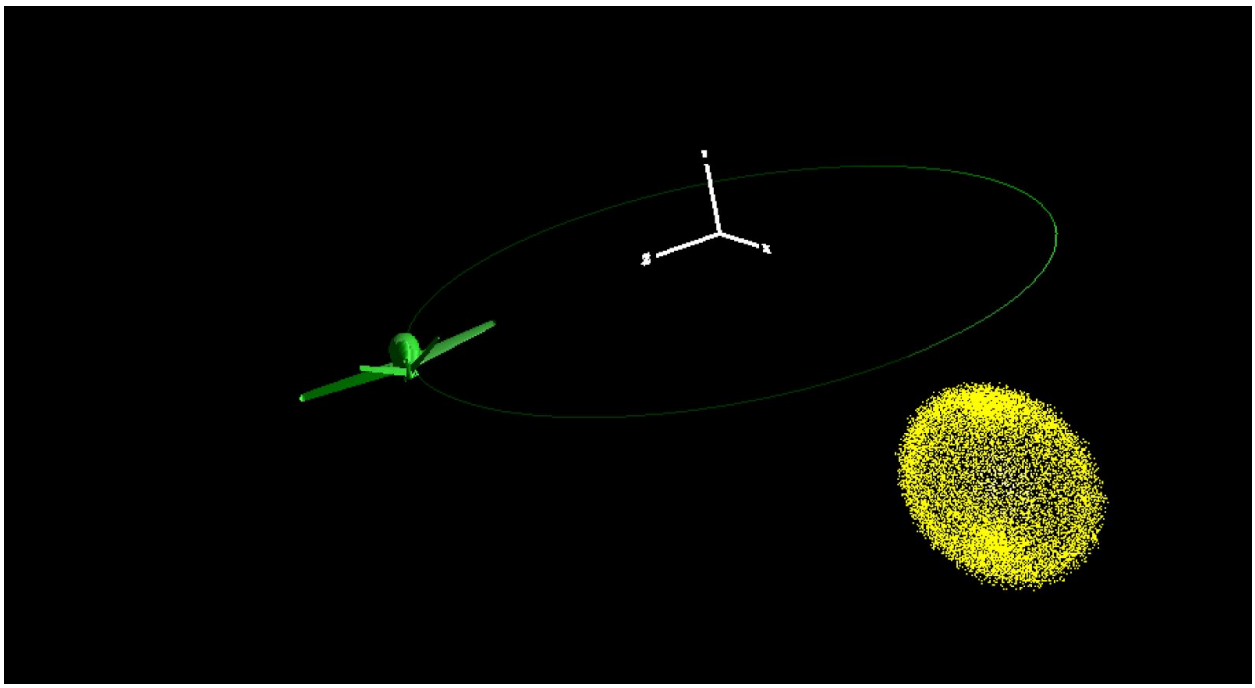
I am into applying physics and trigonometric laws in Graphic. In this final project, I had the chance to do that. Besides, I did not need to use attribute variables in the Shader project. In this project, I use attribute variables to send the new vertices positions to the Vertex Shader.

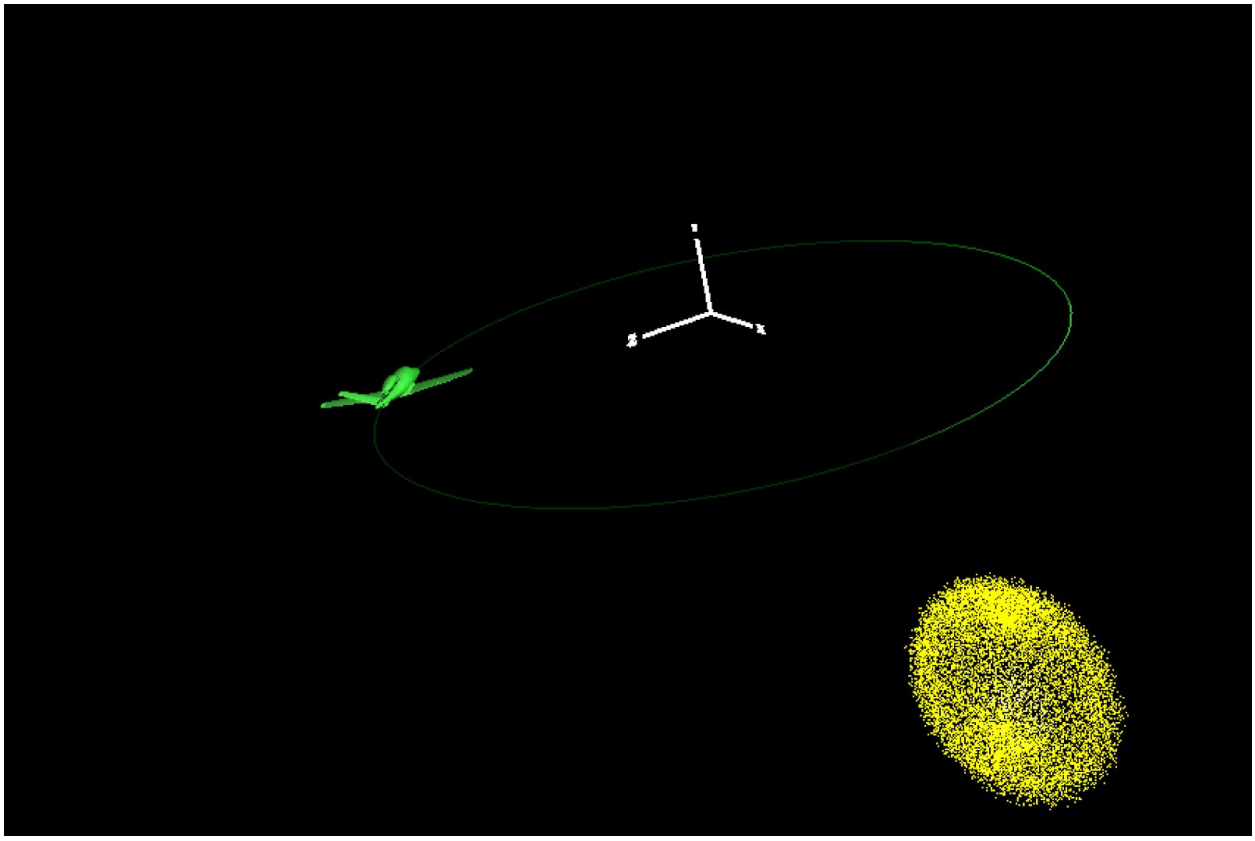
The images of the project



After the airplane have hit the sphere,






Y
Z
X

When the light is turned off,