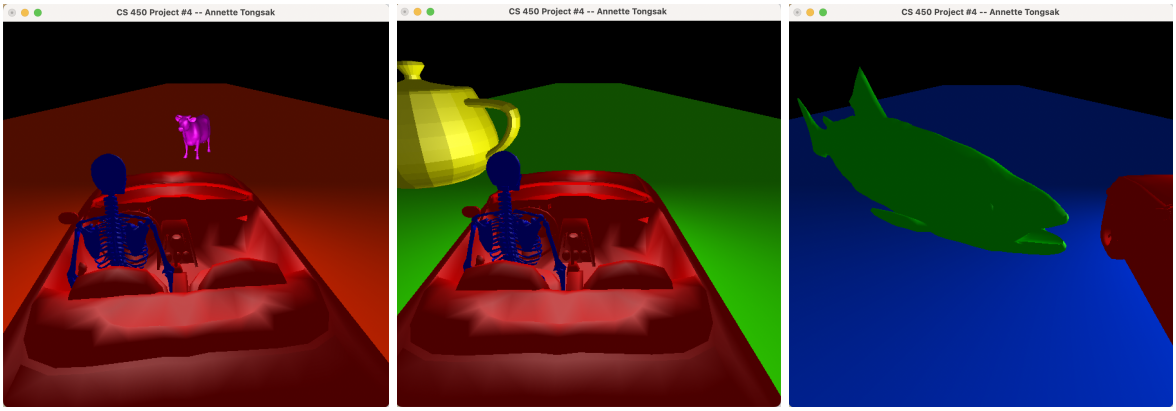
CS 450 Project #4
Keytime Animation

Annette Tongsak
tongsaan@oregonstate.edu



Video link

To create this display, I worked in this order:

1) Drew and animated the car
    a) I keytimed Xpos to set the speed of the car and used ThetaY to animate the car swerving around.
2) Animated normal behind view
    a) The lookAtX keytimes of the view are the same as that of the car's Xpos. Finding eyeX was tricky because the car is moving down the x-axis. However, I found the ideal eye position above the car while it was stationary and then edited eyeX's keytimes to move at the same speed as the car (+10).
3) Drew the long grid road
    a) I called Display List GridDL and scaled it to be 4 x 40—enough to make it seem like the road goes on forever for 10 seconds.
4) Drew and animated the point light
    a) I enabled lighting and applied the car animation to a point light so the car and other objects are illuminated in the scene as time goes on.
5) Drew and animated the cow
    a) I estimated where I wanted the cow to appear by translating it by 10 along the x-axis and then animated its Ypos1 to make it drop onto the road.
6) Drew and animated the teapot
    a) I estimated and translated the position of the teapot, then animated its translation up using Ypos2. I subsequently keytimed its rotation with ThetaY2 so it rotates while popping out of the grid.
7) Drew and animated the salmon
    a) I estimated the furthest intended position of the salmon, then keytimed its abrupt entrance into the scene with Ypos3.

8) Animated the color of the grid
    a) This took a while—I created Red, Blue, and Green quantities and keytimed each of them before applying them to the grid to create the color-changing effect.
9) Drew and animated a skeleton sitting in the car
    a) I downloaded the skeleton.obj file and edited the model in Blender before calling it into the scene. The same translation and rotation animations as the car were applied to the skeleton for it to look like it is driving.
10) Drew and animated the bunny
    a) I estimated the bunny's position before using Ypos and Zpos to translate it and then rotate it with ThetaZ. This makes the bunny look like it is doing a twirl in the air.
11) Animated frontal view
    a) I created eyeX2 to alter the normal x position of the eye so that it is further ahead on the x-axis. I also moved the eye's y position to be lower to get the angle that I wanted. Then, I added a keyboard functionality to switch between normal and frontal view.
12) Animated stuck view
    a) I created LookAtTheta to adjust the view's rotation over time. The keytime theta values are the additive inverse of the car's rotation values. I then added a keyboard functionality to switch between loose and stuck view.

I attempted the extra credit, so there are 16 animated quantities in total. The 8 required quantities are in blue, while my choice quantities are highlighted in yellow.

- Light source
    - Xpos
    - ThetaY
- Viewing
    - eyeX
    - lookAtX
    - LookAtTheta
    - eyeX2
- Positioning
    - Ypos1
    - ThetaY2
    - Ypos2
    - Ypos3
    - Ypos
    - Zpos
    - ThetaZ
- Color
    - Red
    - Green
    - Blue

#	Quantity	Keytime Values - in InitGraphics()
1	Xpos	Xpos.Init( ); Xpos.AddTimeValue(0.0, 0); Xpos.AddTimeValue(2.0, 10); Xpos.AddTimeValue(4.0, 20); Xpos.AddTimeValue(6.0, 30); Xpos.AddTimeValue(8.0, 40); Xpos.AddTimeValue(10.0, 50);

2	eyeX	eyeX.Init();
		eyeX.AddTimeValue(0.0, -4);
		eyeX.AddTimeValue(2.0, 6);
		eyeX.AddTimeValue(4.0, 16);
		eyeX.AddTimeValue(6.0, 26);
		eyeX.AddTimeValue(8.0, 36);
		eyeX.AddTimeValue(10.0, 46);

3	lookAtX	lookAtX.Init();
		lookAtX.AddTimeValue(0.0, 0);
		lookAtX.AddTimeValue(2.0, 10);
		lookAtX.AddTimeValue(4.0, 20);
		lookAtX.AddTimeValue(6.0, 30);
		lookAtX.AddTimeValue(8.0, 40);
		lookAtX.AddTimeValue(10.0, 50);

4	ThetaY	ThetaY.Init(); ThetaY.AddTimeValue(0.0, 0); ThetaY.AddTimeValue(2.0, 30); ThetaY.AddTimeValue(3.0, 10); ThetaY.AddTimeValue(4.0, -10); ThetaY.AddTimeValue(4.5, -13); ThetaY.AddTimeValue(5.0, -10); ThetaY.AddTimeValue(6.0, 0); ThetaY.AddTimeValue(6.5, 0); ThetaY.AddTimeValue(7.0, 0); ThetaY.AddTimeValue(8.0, -15); ThetaY.AddTimeValue(8.5, -5); ThetaY.AddTimeValue(9.0, 0); ThetaY.AddTimeValue(10.0, 0);
5	LookAtTheta	LookAtTheta.Init(); LookAtTheta.AddTimeValue(0.0, 0); LookAtTheta.AddTimeValue(2.0, 30); LookAtTheta.AddTimeValue(3.0, 10); LookAtTheta.AddTimeValue(4.0, -10); LookAtTheta.AddTimeValue(4.5, -13); LookAtTheta.AddTimeValue(5.0, -10); LookAtTheta.AddTimeValue(6.0, 0); LookAtTheta.AddTimeValue(6.5, 0);

		LookAtTheta.AddTimeValue(7.0, 0); LookAtTheta.AddTimeValue(8.0, -15); LookAtTheta.AddTimeValue(8.5, -5); LookAtTheta.AddTimeValue(9.0, 0); LookAtTheta.AddTimeValue(10.0, 0);
6	Ypos1	Ypos1.Init(); Ypos1.AddTimeValue(0.0, 6); Ypos1.AddTimeValue(0.5, 1); Ypos1.AddTimeValue(4.0, 1); Ypos1.AddTimeValue(6.0, 1); Ypos1.AddTimeValue(8.0, 1); Ypos1.AddTimeValue(10.0, 1);
7	ThetaY2	ThetaY2.Init(); ThetaY2.AddTimeValue(0, 0); ThetaY2.AddTimeValue(2, 45); ThetaY2.AddTimeValue(4, 90); ThetaY2.AddTimeValue(6, 135); ThetaY2.AddTimeValue(8, 90); ThetaY2.AddTimeValue(10, 90);
8	Ypos2	Ypos2.Init(); Ypos2.AddTimeValue(0.0, -10); Ypos2.AddTimeValue(2.0, -10); Ypos2.AddTimeValue(2.5, -2); Ypos2.AddTimeValue(6, -2); Ypos2.AddTimeValue(8, -2); Ypos2.AddTimeValue(10, -2);
9	Ypos3	Ypos3.Init(); Ypos3.AddTimeValue(0, -40); Ypos3.AddTimeValue(7, -40); Ypos3.AddTimeValue(7.5, -1); Ypos3.AddTimeValue(8, -3); Ypos3.AddTimeValue(9.5, -4); Ypos3.AddTimeValue(10, -4);

10	Red	Red.Init();
		Red.AddTimeValue(0, 1); // red
		Red.AddTimeValue(2, 1); // yellow
		Red.AddTimeValue(4, 0); // green
		Red.AddTimeValue(6, 0); // cyan
		Red.AddTimeValue(8, 0); // blue
		Red.AddTimeValue(9, 1); // magenta
		Red.AddTimeValue(10, 1); // red

11	Green	Green.Init(); Green.AddTimeValue(0, 0); Green.AddTimeValue(2, 1);

		Green.AddTimeValue(4, 1); Green.AddTimeValue(6, 1); Green.AddTimeValue(8, 0); Green.AddTimeValue(9, 0); Green.AddTimeValue(10, 0); // red
12	Blue	Blue.Init(); Blue.AddTimeValue(0, 0); Blue.AddTimeValue(2, 0); Blue.AddTimeValue(4, 0); Blue.AddTimeValue(6, 1); Blue.AddTimeValue(8, 1); Blue.AddTimeValue(9, 1); Blue.AddTimeValue(10, 0); // red
13	Ypos	Ypos.Init(); Ypos.AddTimeValue(0, -5); Ypos.AddTimeValue(5.49, -5); Ypos.AddTimeValue(5.5, 0); Ypos.AddTimeValue(6, 2.5); Ypos.AddTimeValue(6.5, 0); Ypos.AddTimeValue(8, 0); Ypos.AddTimeValue(9, 0); Ypos.AddTimeValue(10, 0);
14	Zpos	Zpos.Init(); Zpos.AddTimeValue(0, 10); Zpos.AddTimeValue(5.5, 5); Zpos.AddTimeValue(6.5, -5); Zpos.AddTimeValue(8, -10); Zpos.AddTimeValue(9, -10); Zpos.AddTimeValue(10, -10);
15	ThetaZ	ThetaZ.Init(); ThetaZ.AddTimeValue(0, 0); ThetaZ.AddTimeValue(5.5, 0); ThetaZ.AddTimeValue(6.5, -360); ThetaZ.AddTimeValue(8, -360); ThetaZ.AddTimeValue(9, 0); ThetaZ.AddTimeValue(10, 0);

16	eyeX2	eyeX2.Init(); // add 5
		eyeX2.AddTimeValue(0.0, 4);
		eyeX2.AddTimeValue(2.0, 14);
		eyeX2.AddTimeValue(4.0, 24);
		eyeX2.AddTimeValue(6.0, 34);
		eyeX2.AddTimeValue(8.0, 44);
		eyeX2.AddTimeValue(10.0, 54);

I am confident that my animation successfully does what it’s set up to do. This confidence is based on the seamless passing of objects as the car and view move and the observable motion of various elements in the scene. The view animation in particular is designed to maintain a consistent point of view in normal view, with both the eye and look-at positions located above the car. In addition, the animation of the camera when in stuck view is clearly following the rotation path of the car, as the car looks stationary relative to its surroundings. The other animations, such as the grid’s color changes and movements of the cow, teapot, and salmon, are obvious when watching the project.