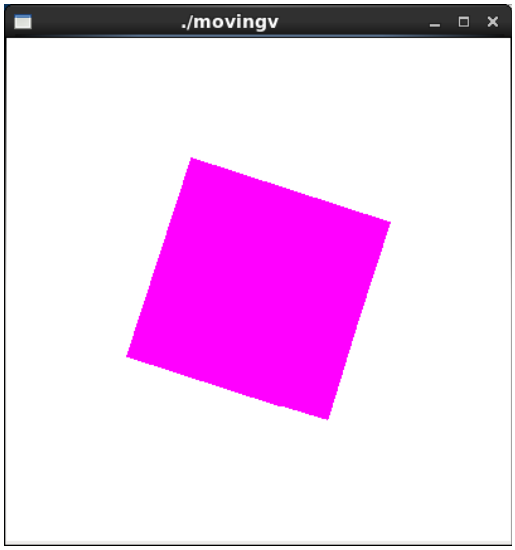
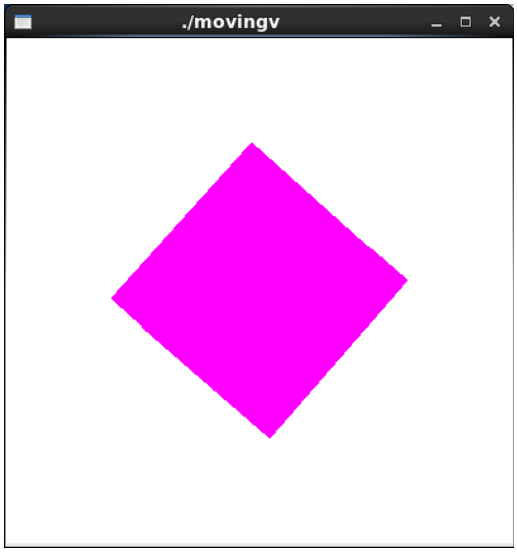
Yazhuo Liu
HW 2

***Write a shader program that animates the rotation of a square about the z-axis.***



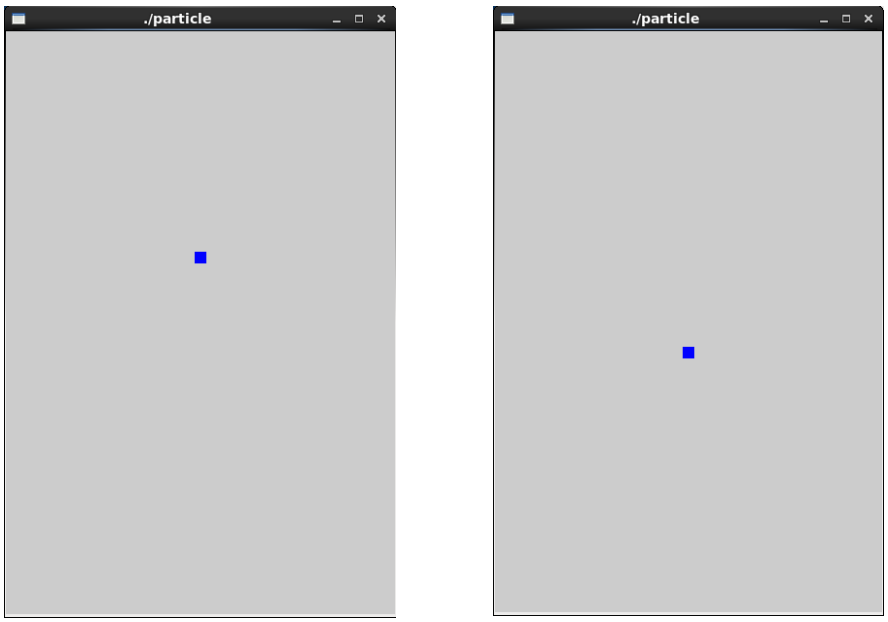




Code:

//square.cpp

```
...
void display(void)
{
  GLfloat vec[4];

  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glClearColor( 1.0, 1.0, 1.0, 0.0 );   //get white background color

  glBegin( GL_POLYGON );
    glVertex2f( -1, -1 );
    glVertex2f( 1, -1 );
    glVertex2f( 1, 1 );
    glVertex2f( -1, 1 );
  glEnd();

  glRotatef(1, 0, 0, 1);

  glutSwapBuffers();
  glFlush();
}
...
```

***Write a vertex shader program that will bounce a sphere whose initial velocity and position are provided by the application program.***



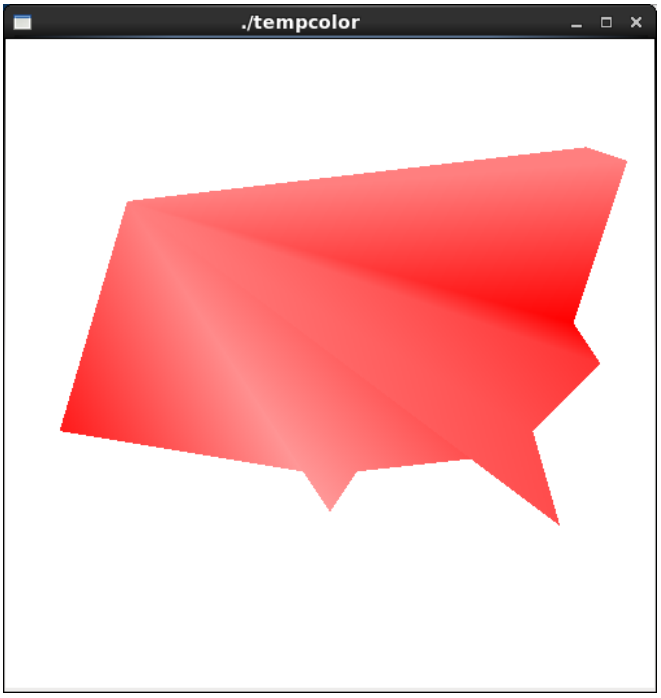

Code:

//particle.cpp

```
...
void display(void)
{
   GLfloat vec[4];

   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glClearColor( 0.8, 0.8, 0.8, 0.0 );              //get white background color

   float h = 12.0;
   glPointSize ( 12 );
   glUniform3f ( velParam, 0, 0, 0 );  //send vel to vertex shader

  glBegin ( GL_POINTS );
    glVertex2f ( 0.0f, h );
  glEnd();

   glutSwapBuffers();
   glFlush();

}
...
```

```
//particle.vert

...
void main(void)
{
  float s = 1000.0;                    //scale factor
  float g = -10.0;
  float t;
  float h, h0;
  float t0;
  float cor = 0.85;
  t =  time / s;                  //time in ms

  h0 = gl_Vertex.y;
  vec3 n = vec3( 0,1,0 );
  vec3 v1;
  vec3 v2;

  t0 = sqrt ( 2.0 * h0 / (-g) );

  v1.x = vel.x;
  v1.y = vel.y + g * t0;
  v1.z = vel.z;

  h = h0 + g/(2.0)*t*t;
  int count = 0;

  while (h <= 0.0) {
    v2 = cor * reflect (v1, n);
    t -= t0;
    if (t < 0.0)  t = 0.0;
    h = v2.y*t + g/(2.0)*t*t;
    h0 = cor * h0;
    if (h0 < 0.0)  h0 = 0.0;
    t0 *= 2;
    v1.y = v2.y + g*t0;
    if (count++ > 100) break;
  }

  gl_Position = gl_ModelViewProjectionMatrix * vec4 (0, h, 0, 1);
}
```

***Approximate the shape of America by a polygon. Find from the Internet or other sources the population density of America. Write a shader program to show briefly America's population distribution with red color indicating an area densely populated, green indicating moderately populated, and white indicating sparsely populated.***




Code:

//density.cpp

...
```
void display(void)
{
   GLfloat vec[4];
   int loc;

   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glClearColor( 1.0, 1.0, 1.0, 0.0 );    //get white background color

   loc = glGetAttribLocation(programObject, "VertexTemp" );

   glBegin( GL_POLYGON );
    glVertexAttrib1f(loc, 0.5);
     glVertex3f ( -1.5,1.2,0 );
    glVertexAttrib1f(loc, 0.9);
     glVertex3f ( -2,-0.5,0 );
    glVertexAttrib1f(loc, 0.4);
     glVertex3f ( -0.2,-0.8,0 );
    glVertexAttrib1f(loc, 0.4);
     glVertex3f ( 0,-1.1,0 );
    glVertexAttrib1f(loc, 0.5);
     glVertex3f ( 0.2,-0.8,0 );
    glVertexAttrib1f(loc, 0.7);
     glVertex3f ( 1.1,-0.7,0 );
    glVertexAttrib1f(loc, 0.7);
```

```
      glVertex3f ( 1.7,-1.2,0 );
     glVertexAttrib1f(loc, 0.7);
      glVertex3f ( 1.5,-0.5,0 );
     glVertexAttrib1f(loc, 0.8);
      glVertex3f ( 2,0,0 );
     glVertexAttrib1f(loc, 1.0);
      glVertex3f ( 1.8,0.3,0 );
     glVertexAttrib1f(loc, 0.5);
      glVertex3f ( 2.2,1.5,0 );
     glVertexAttrib1f(loc, 0.5);
      glVertex3f ( 1.9,1.6,0 );
     //glVertexAttrib1f(loc, 0.6);
      //glVertex3f ( 1.2,0.5,0 );
     //glVertexAttrib1f(loc, 0.6);
      //glVertex3f ( 1,0.9,0 );
    glEnd();

    glutSwapBuffers();
    glFlush();
}
...
```

Report:

I completed the last three problems, but couldn't do the first one.