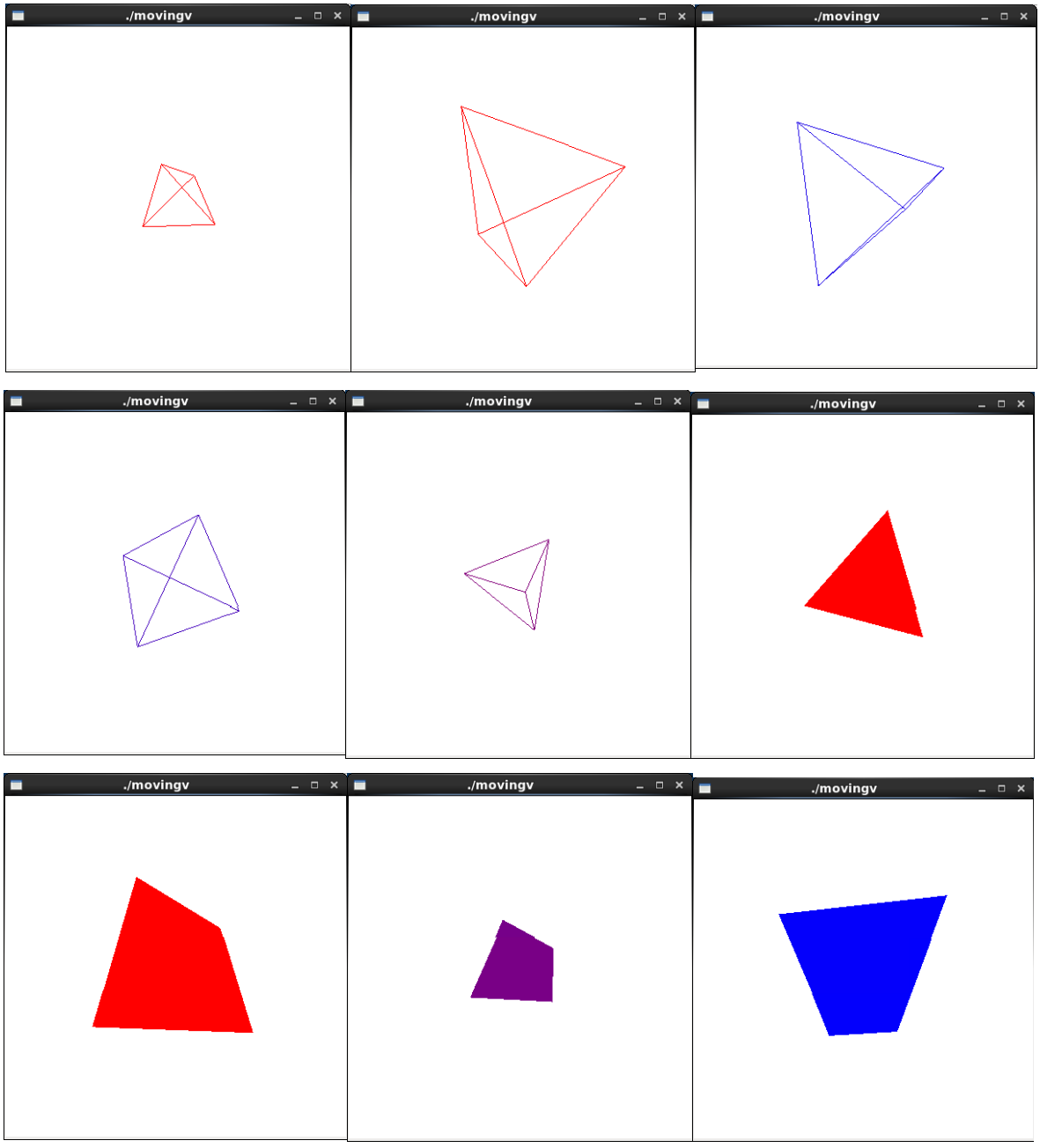
Yazhuo Liu
Homework 3

Write a shader program that renders a colored tetrahedron, which gradually shrinks to a point and expands back to its original shape. While it is shrinking and expanding, the color of the tetrahedron changes.

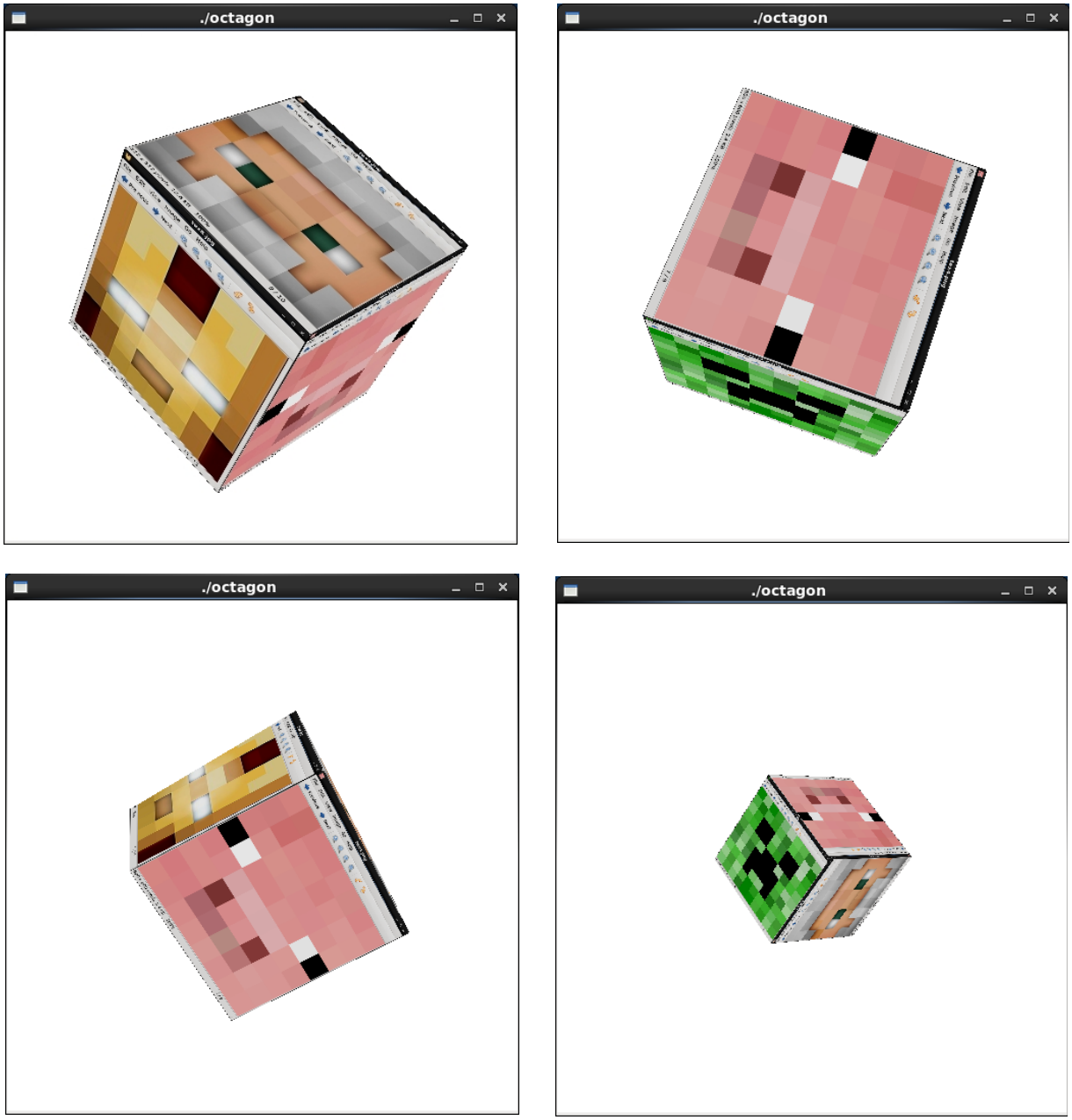


Code:

```
//tetrahedron.cpp

...
int installShaders(const GLchar *vertex, const GLchar *fragment)
{
...
    glUseProgram(programObject);

    GLchar names[][20] = { "CoolColor", "HotColor", "tempRange" };
    GLint loc[10];
    for (int i = 0; i < 3; i++) {
       loc[i] = glGetUniformLocation( programObject, names[i]);
       if (loc[i] == -1)
         printf("No such uniform named %s\n", names[i]);
    }

    glUniform3f(loc[0], 0.0, 0.0, 1);
    glUniform3f(loc[1], 1.0, 0.0, 0.0);
    glUniform1f(loc[2], 1.0);

    return 1;
}

int init(void)
{
...
   timeParam = glGetUniformLocation ( programObject, "time" );
...
}

...
static void Idle(void)
{
   glUniform1f( timeParam, glutGet ( GLUT_ELAPSED_TIME ) );
   glutPostRedisplay();
}

...
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

   float angle = glGetAttribLocation(programObject, "angle");
   int loc = glGetAttribLocation(programObject, "VertexTemp" );
```

```
   glRotatef(angle, 1, 1, 1);

   //glutWireTetrahedron();
   glutSolidTetrahedron();

   glutSwapBuffers();
   glFlush();
}

...

//tetrahedron.vert

uniform float time;     //value provided by application program
uniform float tempRange;

attribute float angle;
attribute float VertexTemp;

varying float temperature;

void main(void)
{
  float s = 0;

  angle += 2.0 * sin ( 0.0005 * time );

  s = s + 2.0 * sin ( 0.0005 * time );

  temperature = (VertexTemp / tempRange) + sin ( 0.0005 * time );

  gl_Position = gl_ModelViewProjectionMatrix * (vec4(s, s, s, 1.0)  * gl_Vertex );
}

//tetrahedron.frag

uniform vec3 CoolColor;
uniform vec3 HotColor;

varying float temperature;

void main(void)
{
   vec3 color = mix( CoolColor, HotColor, temperature );

   gl_FragColor = vec4( color, 1);
}
```

Write a shader program that uses texture techniques to paste 6 different images on the 6 faces of a cube . The cube rotates and changes size from time to time.



Code:

//cube.cpp

```
...
void init2DTexture()
{
  GLubyte *texImage_1 = makeTexImage( "tex1.png" );
  glGenTextures(1, &texName1);
  glBindTexture(GL_TEXTURE_2D, texName1);          //now we work on texName
```

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA, iWidth, iHeight, 0,
        GL_RGBA, GL_UNSIGNED_BYTE, texImage_1);

GLubyte *texImage_2 = makeTexImage( "tex2.png" );
glGenTextures(1, &texName2);
glBindTexture(GL_TEXTURE_2D, texName2);          //now we work on texName
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA, iWidth, iHeight, 0,
        GL_RGBA, GL_UNSIGNED_BYTE, texImage_2);

GLubyte *texImage_3 = makeTexImage( "tex3.png" );
glGenTextures(1, &texName3);
glBindTexture(GL_TEXTURE_2D, texName3);          //now we work on texName
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA, iWidth, iHeight, 0,
        GL_RGBA, GL_UNSIGNED_BYTE, texImage_3);

GLubyte *texImage_4 = makeTexImage( "tex4.png" );
glGenTextures(1, &texName4);
glBindTexture(GL_TEXTURE_2D, texName4);          //now we work on texName
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA, iWidth, iHeight, 0,
        GL_RGBA, GL_UNSIGNED_BYTE, texImage_4);

GLubyte *texImage_5 = makeTexImage( "tex5.png" );
glGenTextures(1, &texName5);
glBindTexture(GL_TEXTURE_2D, texName5);          //now we work on texName
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA, iWidth, iHeight, 0,
        GL_RGBA, GL_UNSIGNED_BYTE, texImage_5);

GLubyte *texImage_6 = makeTexImage( "tex6.png" );
glGenTextures(1, &texName6);
```

```
  glBindTexture(GL_TEXTURE_2D, texName6);          //now we work on texName
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
  glTexImage2D( GL_TEXTURE_2D, 0, GL_RGBA, iWidth, iHeight, 0,
                GL_RGBA, GL_UNSIGNED_BYTE, texImage_6);

  delete texImage_1;
  delete texImage_2;
  delete texImage_3;
  delete texImage_4;
  delete texImage_5;
  delete texImage_6;
}

...
int init(void)
{
...
   timeParam = glGetUniformLocation ( programObject, "time" );
...
}

...
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

   float angle = glGetAttribLocation(programObject, "angle");

   glRotatef( angle, 1, 0, 1 );

   glActiveTexture(GL_TEXTURE0);
   glBindTexture(GL_TEXTURE_2D, texName1);

   glBegin ( GL_QUADS ); //back face
     glTexCoord2f (0,0);
     glVertex3f (-1,-1,-1);
     glTexCoord2f (1,0);
     glVertex3f (-1,1,-1);
     glTexCoord2f (1,1);
     glVertex3f (1,1,-1);
     glTexCoord2f (0,1);
     glVertex3f (1,-1,-1);
   glEnd();
```

```
  //glActiveTexture(GL_TEXTURE1);
  glBindTexture(GL_TEXTURE_2D, texName2);
  glBegin ( GL_QUADS ); //front face
    glTexCoord2f (0,0);
    glVertex3f (-1,-1,1);
    glTexCoord2f (1,0);
    glVertex3f (1,-1,1);
    glTexCoord2f (1,1);
    glVertex3f (1,1,1);
    glTexCoord2f (0,1);
    glVertex3f (-1,1,1);
  glEnd();

  glBindTexture(GL_TEXTURE_2D, texName3);
  glBegin ( GL_QUADS ); //right face
    glTexCoord2f (0,0);
    glVertex3f (1,-1,-1);
    glTexCoord2f (1,0);
    glVertex3f (1,1,-1);
    glTexCoord2f (1,1);
    glVertex3f (1,1,1);
    glTexCoord2f (0,1);
    glVertex3f (1,-1,1);
  glEnd();

  glBindTexture(GL_TEXTURE_2D, texName4);
  glBegin ( GL_QUADS ); // left face
    glTexCoord2f (0,0);
    glVertex3f (-1,-1,-1);
    glTexCoord2f (1,0);
    glVertex3f (-1,-1,1);
    glTexCoord2f (1,1);
    glVertex3f (-1,1,1);
    glTexCoord2f (0,1);
    glVertex3f (-1,1,-1);
  glEnd();

  glBindTexture(GL_TEXTURE_2D, texName5);
  glBegin ( GL_QUADS ); // top face
    glTexCoord2f (0,0);
    glVertex3f (-1,1,-1);
    glTexCoord2f (1,0);
    glVertex3f (-1,1,1);
    glTexCoord2f (1,1);
    glVertex3f (1,1,1);
    glTexCoord2f (0,1);
    glVertex3f (1,1,-1);
  glEnd();
```

```
  glBindTexture(GL_TEXTURE_2D, texName6);
  glBegin ( GL_QUADS ); // bottom face
    glTexCoord2f (0,0);
    glVertex3f (-1,-1,-1);
    glTexCoord2f (1,0);
    glVertex3f (1,-1,-1);
    glTexCoord2f (1,1);
    glVertex3f (1,-1,1);
    glTexCoord2f (0,1);
    glVertex3f (-1,-1,1);
  glEnd();

  glutSwapBuffers();
  glFlush();
}

...

//cube.vert

...
  angle += 2.0 * sin ( 0.0005 * time );
...
```

Report:

I had some troubles when I was doing the second part of problem 2, but after I searched similar problems online, I believe I have fixed the problems.