ARLAB

ME/CprE/ComS 557

Computer Graphics and Geometric Modeling

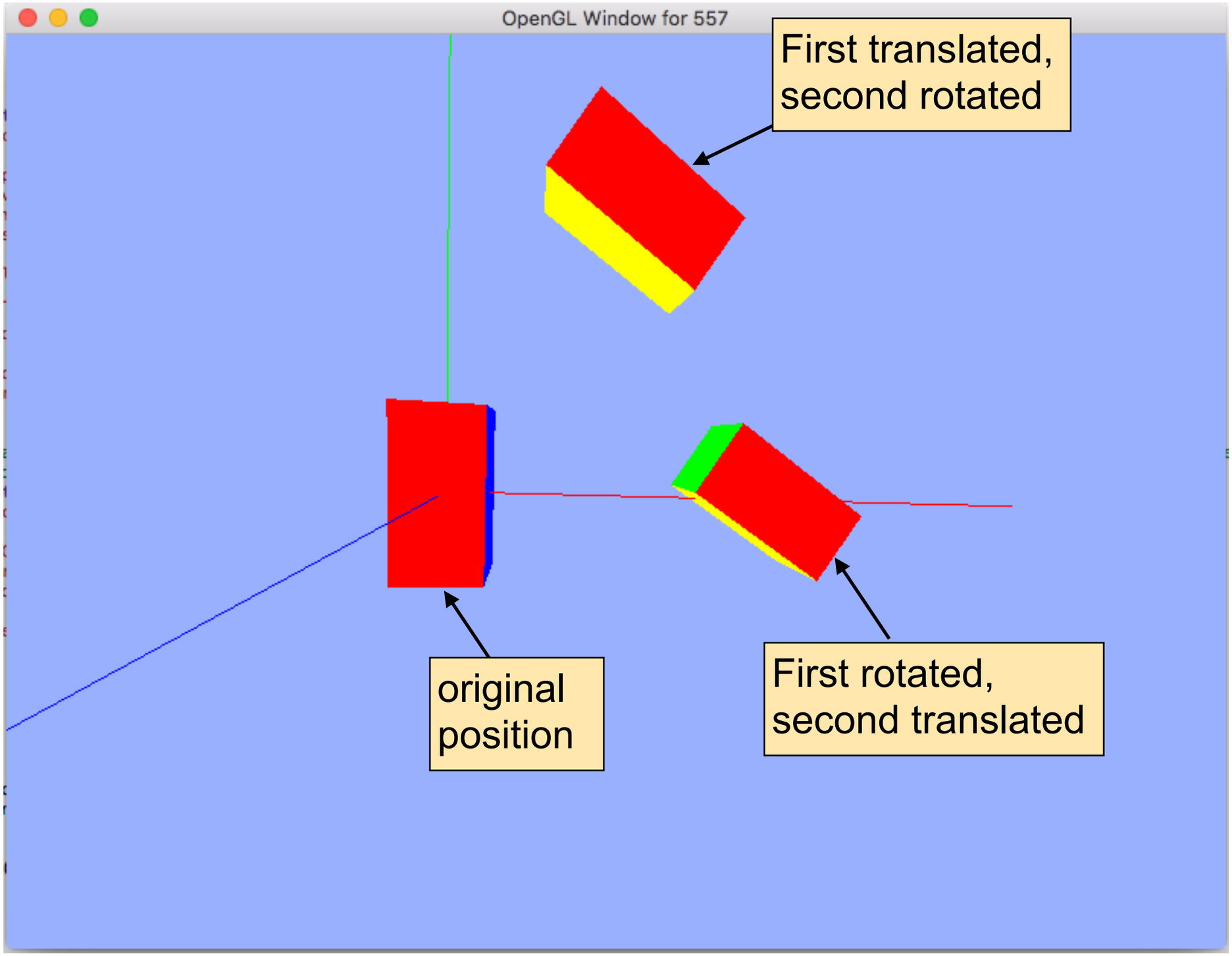
Transformation Example

September 22, 2015

Rafael Radkowski

IOWA STATE UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Transformation Example



Code Example

ARLAB

```
////////////////////////////////////////////////////////////////////////////////
//// Create some models

// coordinate system
CoordSystem* cs = new CoordSystem(4.0);

// This example shows three boxes at different locations
// The first box is at its local origin, the location where the box was created.

// three identical boxes
// on its original location.
GLColoredBox* box_original = new GLColoredBox(0.5,1.0,0.5);

// This box will be rotated first and then translated
GLColoredBox* box_rotation_first = new GLColoredBox(0.5,1.0,0.5);

// This box will be translated first and than rotated.
GLColoredBox* box_translation_first = new GLColoredBox(0.5,1.0,0.5);

// This defines two matrices, one for translation and one for rotations
glm::mat4 model_matrix1 = glm::mat4(), model_matrix2 = glm::mat4();
glm::mat4 translation = glm::translate(glm::vec3(2.0f, 0.0f, 0.0f));
glm::mat4 rotation = glm::rotate( 1.0f, glm::vec3(0.0f, 0.0f, 1.0f));

// Here we translate first and rotate second
model_matrix2 =  rotation * translation;
box_translation_first->setModelMatrix(model_matrix2);

// Here we rotate first and translate second
model_matrix1 = translation * rotation;
box_rotation_first->setModelMatrix(model_matrix1);
```

The coordinate system

Code Example

ARLAB

```
////////////////////////////////////////////////////////////////////////////////
//// Create some models

// coordinate system
CoordSystem* cs = new CoordSystem(4.0);

// This example shows three boxes at different locations
// The first box is at its local origin, the location where the box was created.

// three identical boxes
// on its original location.
GLColoredBox* box_original = new GLColoredBox(0.5,1.0,0.5);

// This box will be rotated first and then translated
GLColoredBox* box_rotation_first = new GLColoredBox(0.5,1.0,0.5);

// This box will be translated first and than rotated.
GLColoredBox* box_translation_first = new GLColoredBox(0.5,1.0,0.5);

// This defines two matrices, one for translation and one for rotations
glm::mat4 model_matrix1 = glm::mat4(), model_matrix2 = glm::mat4();
glm::mat4 translation = glm::translate(glm::vec3(2.0f, 0.0f, 0.0f));
glm::mat4 rotation = glm::rotate( 1.0f, glm::vec3(0.0f, 0.0f, 1.0f));

// Here we translate first and rotate second
model_matrix2 =  rotation * translation;
box_translation_first->setModelMatrix(model_matrix2);

// Here we rotate first and translate second
model_matrix1 = translation * rotation;
box_rotation_first->setModelMatrix(model_matrix1);
```

The three boxes

GLColoredBox

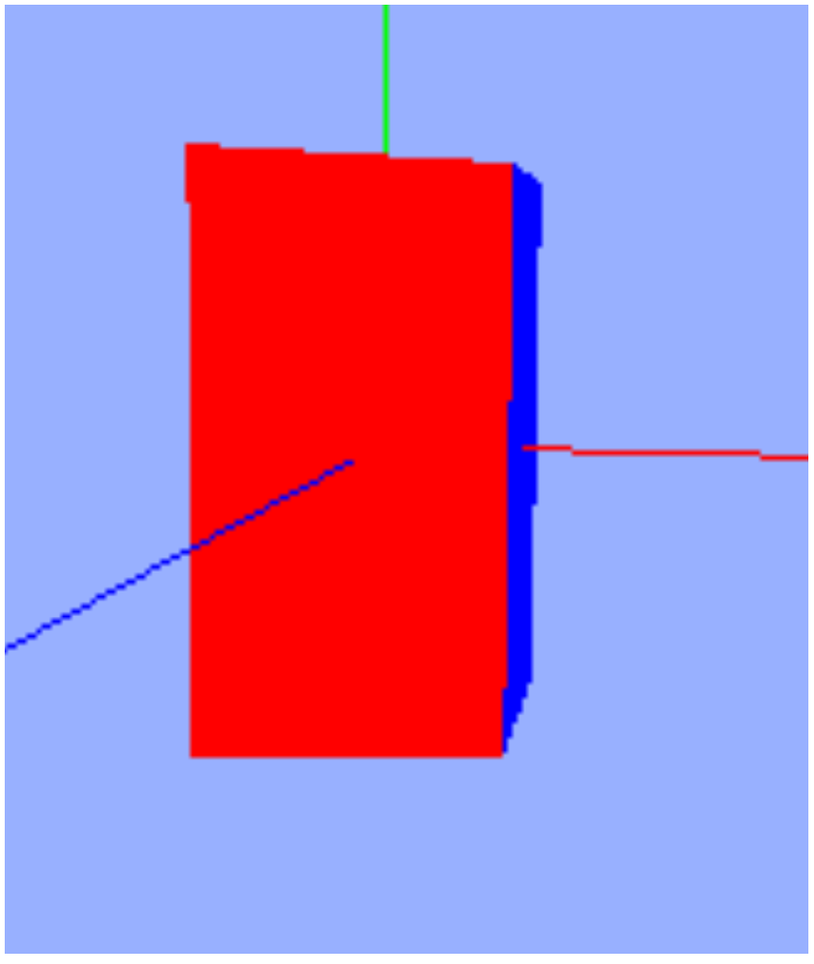
The class GLColoredBox renders a box with six different face colors.

Constructor:

```
GLColoredBox(float size_x, float size_y, float size_z)
```

Parameters:

- size_x, size_y, size_x: the edge length along x, y, and z.



Functions:

- setModelMatrix(glm::mat4 matrix)
  set a model matrix to specify the position of this object
- glm::mat4 getModelMatrix(void)
  returns the current model matrix
- draw(void)
  draws the object

Code Example

ARLAB

```
////////////////////////////////////////////////////////////////////////////////
//// Create some models

// coordinate system
CoordSystem* cs = new CoordSystem(4.0);

// This example shows three boxes at different locations
// The first box is at its local origin, the location where the box was created.

// three identical boxes
// on its original location.
GLColoredBox* box_original = new GLColoredBox(0.5,1.0,0.5);

// This box will be rotated first and then translated
GLColoredBox* box_rotation_first = new GLColoredBox(0.5,1.0,0.5);

// This box will be translated first and than rotated.
GLColoredBox* box_translation_first = new GLColoredBox(0.5,1.0,0.5);

// This defines two matrices, one for translation and one for rotations
glm::mat4 model_matrix1 = glm::mat4(), model_matrix2 = glm::mat4();
glm::mat4 translation = glm::translate(glm::vec3(2.0f, 0.0f, 0.0f));
glm::mat4 rotation = glm::rotate( 1.0f, glm::vec3(0.0f, 0.0f, 1.0f));

// Here we translate first and rotate second
model_matrix2 =  rotation * translation;
box_translation_first->setModelMatrix(model_matrix2);

// Here we rotate first and translate second
model_matrix1 = translation * rotation;
box_rotation_first->setModelMatrix(model_matrix1);
```

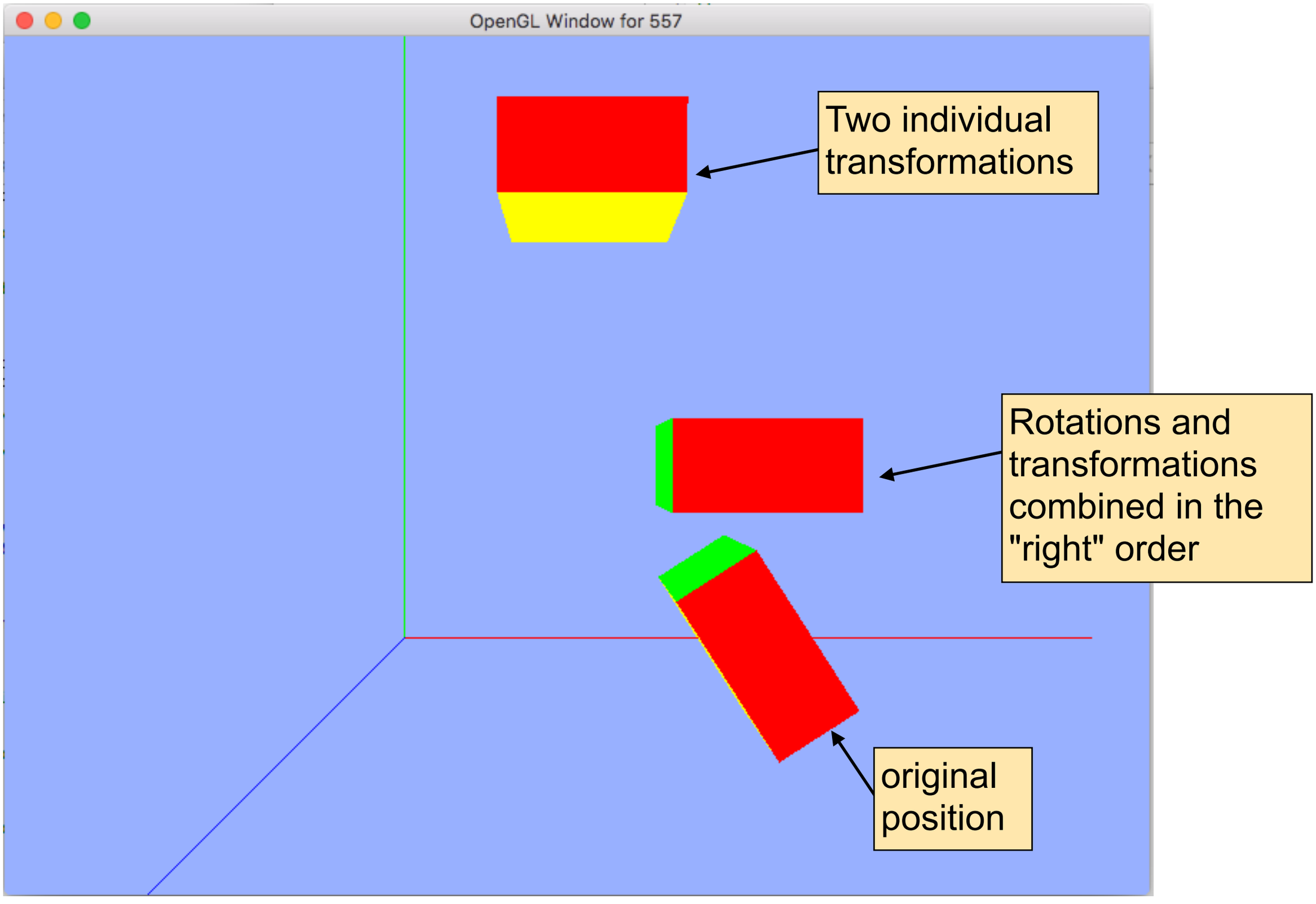
The transformations

```
// This is our render loop. As long as our window remains open (ESC is not pressed
while(!glfwWindowShouldClose(window))
{

    // Clear the entire buffer with our green color (sets the background to be gre
    glClearBufferfv(GL_COLOR , 0, clear_color);
    glClearBufferfv(GL_DEPTH , 0, clear_depth);


    ////////////////////////////////////////////////////////////////////////////////
    //// This renders the objects

    // Set the trackball locatiom
    SetTrackballLocation(trackball.getRotationMatrix());

    // draw the objects
    cs->draw();

    box_original->draw();

    box_rotation_first->draw();

    box_translation_first->draw();

    //// This renders the objects
    ////////////////////////////////////////////////////////////////////////////////


    // Swap the buffers so that what we drew will appear on the screen.
    glfwSwapBuffers(window);
    glfwPollEvents();

}
```

Set the current camera

This draws the object

Code Example 2



Code Example 2

The objects

```
////////////////////////////////////////////////////////////////////////////////
//// Create some models

// coordinate system
CoordSystem* cs = new CoordSystem(4.0);

// This example demonstrate how to invert the model matrix of an box that is not at its location.
// Rotate models only when they are at the orign. Otherwise, you rotate them with a lever.

// three identical boxes
// on its original location.
GLColoredBox* box_rotated_at_origin = new GLColoredBox(0.5,1.0,0.5);

// This box will be rotated first and then translated
GLColoredBox* box_rotated_at_curr_location = new GLColoredBox(0.5,1.0,0.5);

GLColoredBox* box = new GLColoredBox(0.5,1.0,0.5);
```

Code Example 2

```
// These matrices define the shifted location of all our objects. We suppose, they
// are not at their origin locaiton
glm::mat4 translation_first = glm::translate(glm::vec3(2.0f, 0.0f, 0.0f));
glm::mat4 rotation_first = glm::rotate( 0.57f, glm::vec3(0.0f, 0.0f, 1.0f));

glm::mat4 matrix_first = translation_first * rotation_first;

// Here we transform the objects for the first time.
box_rotated_at_origin->setModelMatrix(matrix_first);
box_rotated_at_curr_location->setModelMatrix(matrix_first);
box->setModelMatrix(matrix_first);
```

Code Example 2

```
// Let's assume, we want to further rotate and translate the models.
glm::mat4 translation_second = glm::translate(glm::vec3(0.0f, 1.0f, 0.0f));
glm::mat4 rotation_second = glm::rotate( 1.0f, glm::vec3(0.0f, 0.0f, 1.0f));

// This returns the current model matrix of each object
glm::mat4& current_location_0 = box_rotated_at_origin->getModelMatrix();
glm::mat4& current_location_1 = box_rotated_at_curr_location->getModelMatrix();

// here we take the current model matrix, and rotate first, next we translate.
glm::mat4 matrix_second_1 =  translation_second *  rotation_second * current_location_1;

// Here we rotate and add all objects in the order, rotation first, then translation.
glm::mat4 matrix_second_2 =  translation_second * translation_first * rotation_second * rotation_first;
```

```
// set the object to its new location
box_rotated_at_curr_location->setModelMatrix(matrix_second_1);
box_rotated_at_origin->setModelMatrix(matrix_second_2);
```

$$M = t_2 \; r_2 \; t_1 \; r_1$$

$$M = t_2 \; t_1 \; r_2 \; r_1$$