CSC4140 Assignment 2

Computer Graphics

February 25, 2022

Learn to use VirtualBox and Mathematic Review

This assignment is 10% of the total mark.

Student ID: 120090266

Student Name: Feng Yutong

This assignment represents my own work in accordance with University regulations.

Signature: Feng Yutong

1 Introduction of HW2

The task of this assignment is to get a model to screen. Given three points, the program will transform them to screen coordinates and draw the corresponding wireframe triangle on the screen. Since the rasterizer is rendered frame by frame, you can use the A and D keys to rotate the triangle around the any axis $\boldsymbol{u}$. It is determined by two points, i.e., $\boldsymbol{u} = P1 - P2$. The translation and scale operation are expressed as Vector3f $\boldsymbol{T}$ and $\boldsymbol{S}$. The model transformation is defined by

$$Model = Model_S \cdot Model_R \cdot Model_T$$

The view transformation is defined by

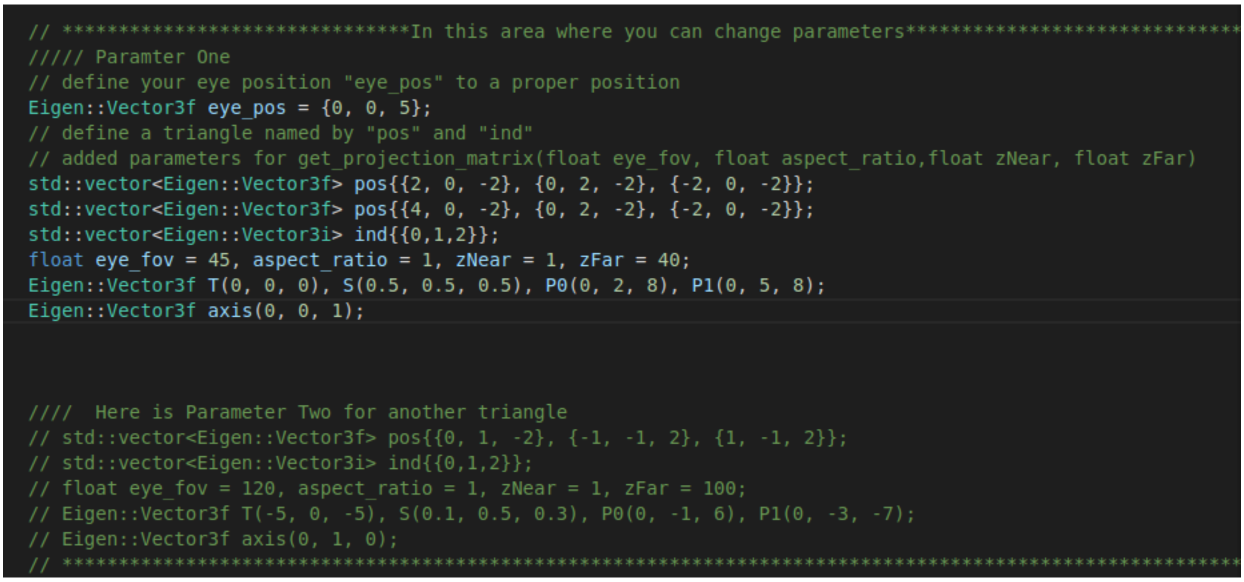
$$View = View_R \cdot View_T$$

where eye position is $(0, 0, 5)$.

During projection, the aspect ratio is *aspect_ratio*, eye fov is *eye_fov*, near plane is at *zNear* , far plane is at *zFar*. The whole process of MVP is defined by

$$Projection \cdot Rotation \cdot Model$$

You can run the program by entering *bash compile_run.sh* in the terminal and see the results. You can see the result of another set of parameters by changing annotation in main.cpp or enter your own parameters in the following area. You can also change line color by modifying *line_color*. The code provides parameters of triangle one and three in the result section.



```
// *******************************In this area where you can change parameters******************************
///// Paramter One
// define your eye position "eye_pos" to a proper position
Eigen::Vector3f eye_pos = {0, 0, 5};
// define a triangle named by "pos" and "ind"
// added parameters for get_projection_matrix(float eye_fov, float aspect_ratio,float zNear, float zFar)
std::vector<Eigen::Vector3f> pos{{2, 0, -2}, {0, 2, -2}, {-2, 0, -2}};
std::vector<Eigen::Vector3f> pos{{4, 0, -2}, {0, 2, -2}, {-2, 0, -2}};
std::vector<Eigen::Vector3i> ind{{0,1,2}};
float eye_fov = 45, aspect_ratio = 1, zNear = 1, zFar = 40;
Eigen::Vector3f T(0, 0, 0), S(0.5, 0.5, 0.5), P0(0, 2, 8), P1(0, 5, 8);
Eigen::Vector3f axis(0, 0, 1);



////  Here is Parameter Two for another triangle
// std::vector<Eigen::Vector3f> pos{{0, 1, -2}, {-1, -1, 2}, {1, -1, 2}};
// std::vector<Eigen::Vector3i> ind{{0,1,2}};
// float eye_fov = 120, aspect_ratio = 1, zNear = 1, zFar = 100;
// Eigen::Vector3f T(-5, 0, -5), S(0.1, 0.5, 0.3), P0(0, -1, 6), P1(0, -3, -7);
// Eigen::Vector3f axis(0, 1, 0);
// *********************************************************************************************************
```

Figure 1: Area for self-defined parameters

2 Results of HW2

2.1 Result1

The corresponding parameters are: triangle: (4, 0, -2), (0, 2, -2), (-2, 0, -2), eye_fov = 45, aspect_ratio = 1, zNear = -1, zFar = -40, $T = (0, 0, 0), S = (0.5, 0.5, 0.5), P_0 = (0, 1, 8), P_1 = (0, 1, 4)$, line color=(65, 155, 205).

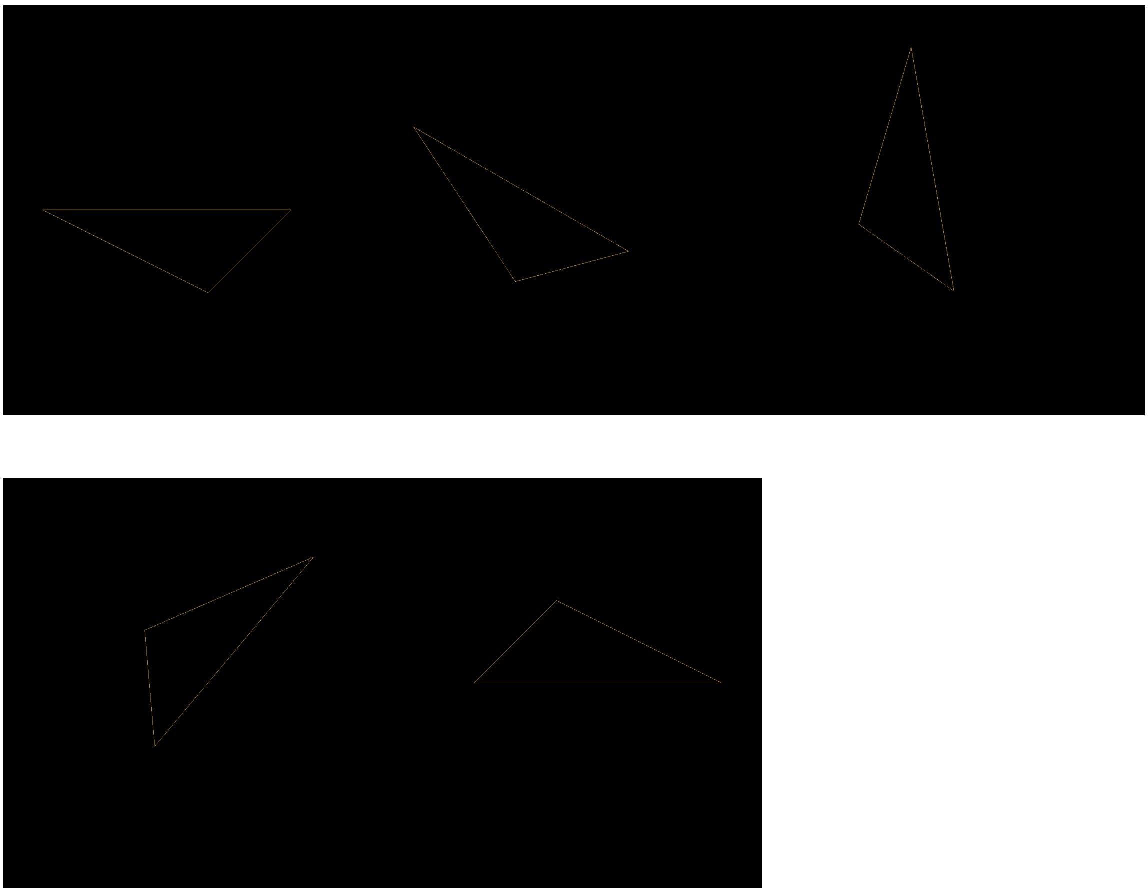


Figure 2: Result1 — 5 stage for triangle one when keeping pressing A keys

2.2 Result2

The corresponding parameters are: triangle: (4, 0, -2), (0, 2, -2), (-2, 0, -2), eye_fov = 45, aspect_ratio = 1, zNear = -1, zFar = -40, $T = (0, 0, 0), S = (0.25, 0.25, 0.25), P_0 = (0, 1, 8), P_1 = (0, 3, 8)$, line color=(255, 255, 255)

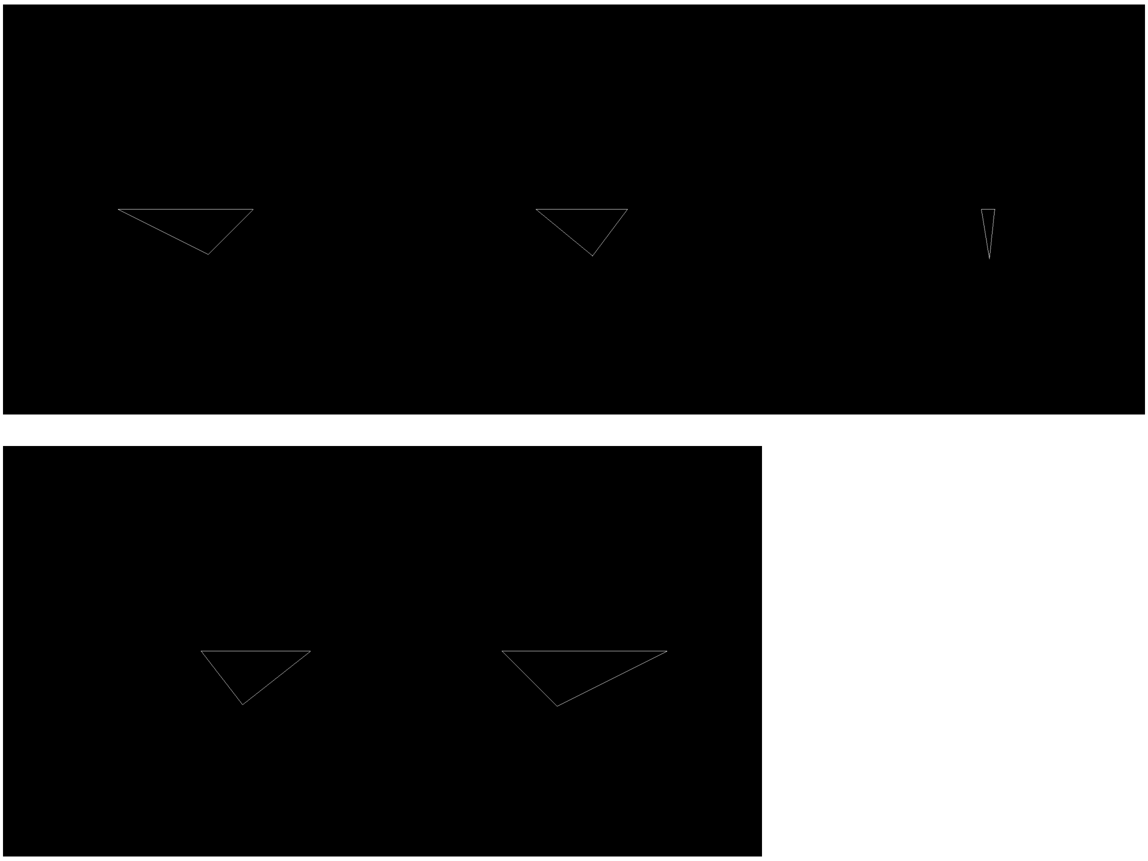
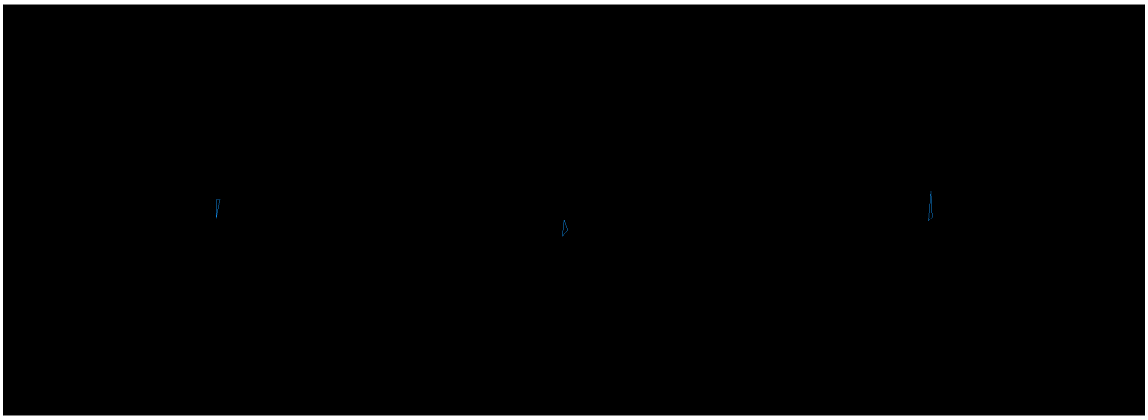


Figure 3: Result2 — 5 stage for triangle two when keeping pressing A keys

2.3 Result3

The corresponding parameters are: triangle: (0, 1, -2), (1, -1, 2), (1, -1, 2), eye_fov = 120, aspect_ratio = 1, zNear = -1, zFar = -100, $T = (0, 0, 0), S = (0.5, 0.5, 0.5), P_0 = (0, -1, 6), P_1 = (0, -3, -7)$, line color=(255, 153, 18).



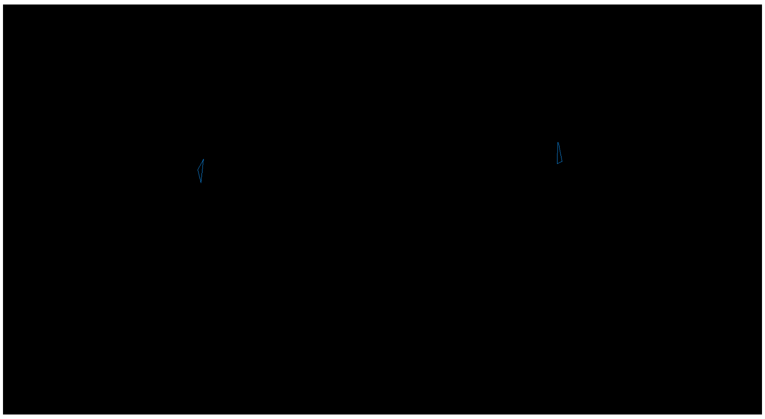


Figure 4: Result3 — 5 stage for triangle three when keeping pressing A keys