投影矩阵算出来 projection(3, 2) = 1;，但画出来是个倒的，好像说完全相反也正常，释怀了

任意轴旋转直接套公式就可以了；

#include "Triangle.hpp"

#include "rasterizer.hpp"

#include <eigen3/Eigen/Eigen>

#include <iostream>

#include <opencv2/opencv.hpp>

constexpr double MY\_PI = 3.1415926;

Eigen::Matrix4f get\_view\_matrix(Eigen::Vector3f eye\_pos)

{

Eigen::Matrix4f view = Eigen::Matrix4f::Identity();

Eigen::Matrix4f translate;

translate << 1, 0, 0, -eye\_pos[0], 0, 1, 0, -eye\_pos[1], 0, 0, 1,

-eye\_pos[2], 0, 0, 0, 1;

view = translate \* view;

return view;

}

Eigen::Matrix4f get\_model\_matrix(float rotation\_angle)

{

// TODO: Implement this function

// Create the model matrix for rotating the triangle around the Z axis.

// Then return it.

Eigen::Matrix4f model = Eigen::Matrix4f::Identity();

float r = rotation\_angle / 180.0 \* MY\_PI;

Eigen::Matrix4f translate;

translate << cos(r), -sin(r), 0, 0,

sin(r), cos(r), 0, 0,

0, 0, 1, 0,

0, 0, 0, 1;

model = translate \* model;

return model;

}

Eigen::Matrix4f get\_projection\_matrix(float eye\_fov, float aspect\_ratio, float zNear, float zFar)

{

Eigen::Matrix4f projection = Eigen::Matrix4f::Zero();

float half\_eye\_fovY = eye\_fov / 2.0f / 180.0f \* MY\_PI;

float top = zNear \* tan(half\_eye\_fovY);

float bottom = -top;

float right = aspect\_ratio \* top;

float left = -right;

projection(0, 0) = 2\*zNear / (right - left);

projection(1, 1) = 2\*zNear / (top - bottom);

projection(2, 2) = (zFar + zNear) / (zFar - zNear);

projection(2, 3) = -2 \* zFar \* zNear / (zFar - zNear);

projection(3, 2) = -1;

return projection;

}

Eigen::Matrix4f get\_rotation(Vector3f axis, float angle) {

Eigen::Matrix4f rotation = Eigen::Matrix4f::Identity();

Eigen::Vector3f axis\_normalized = axis.normalized();

float cos\_theta = cos(angle);

float sin\_theta = sin(angle);

float one\_minus\_cos\_theta = 1 - cos\_theta;

rotation(0, 0) = cos\_theta + axis\_normalized[0] \* axis\_normalized[0] \* one\_minus\_cos\_theta;

rotation(0, 1) = axis\_normalized[0] \* axis\_normalized[1] \* one\_minus\_cos\_theta - axis\_normalized[2] \* sin\_theta;

rotation(0, 2) = axis\_normalized[0] \* axis\_normalized[2] \* one\_minus\_cos\_theta + axis\_normalized[1] \* sin\_theta;

rotation(1, 0) = axis\_normalized[1] \* axis\_normalized[0] \* one\_minus\_cos\_theta + axis\_normalized[2] \* sin\_theta;

rotation(1, 1) = cos\_theta + axis\_normalized[1] \* axis\_normalized[1] \* one\_minus\_cos\_theta;

rotation(1, 2) = axis\_normalized[1] \* axis\_normalized[2] \* one\_minus\_cos\_theta - axis\_normalized[0] \* sin\_theta;

rotation(2, 0) = axis\_normalized[2] \* axis\_normalized[0] \* one\_minus\_cos\_theta - axis\_normalized[1] \* sin\_theta;

rotation(2, 1) = axis\_normalized[2] \* axis\_normalized[1] \* one\_minus\_cos\_theta + axis\_normalized[0] \* sin\_theta;

rotation(2, 2) = cos\_theta + axis\_normalized[2] \* axis\_normalized[2] \* one\_minus\_cos\_theta;

return rotation;

}

int main(int argc, const char\*\* argv)

{

float angle = 0;

bool command\_line = false;

std::string filename = "output.png";

if (argc >= 3) {

command\_line = true;

angle = std::stof(argv[2]); // -r by default

if (argc == 4) {

filename = std::string(argv[3]);

}

else

return 0;

}

rst::rasterizer r(700, 700);

Eigen::Vector3f eye\_pos = {0, 0, 5};

std::vector<Eigen::Vector3f> pos{{2, 0, -2}, {0, 2, -2}, {-2, 0, -2}};

std::vector<Eigen::Vector3i> ind{{0, 1, 2}};

auto ind\_id = r.load\_indices(ind);

int key = 0;

int frame\_count = 0;

while (key != 27) {

auto pos\_id = r.load\_positions(pos);

r.clear(rst::Buffers::Color | rst::Buffers::Depth);

r.set\_model(get\_model\_matrix(angle));

r.set\_view(get\_view\_matrix(eye\_pos));

r.set\_projection(get\_projection\_matrix(45, 1, 0.1, 50));

r.draw(pos\_id, ind\_id, rst::Primitive::Triangle);

cv::Mat image(700, 700, CV\_32FC3, r.frame\_buffer().data());

image.convertTo(image, CV\_8UC3, 1.0f);

cv::imshow("image", image);

key = cv::waitKey(10);

std::cout << "frame count: " << frame\_count++ << '\n';

if (key == 'q') {

angle += 10;

}

else if (key == 'e') {

angle -= 10;

}

else if (key == 'w') {

eye\_pos[1] -= 0.1;

}

else if (key =='s') {

eye\_pos[1] += 0.1;

}

else if (key == 'a') {

eye\_pos[0] += 0.1;

}

else if (key == 'd') {

eye\_pos[0] -= 0.1;

}

}

return 0;

}