华东师范大学计算机科学与技术学院实验报告

实验课程：计算机图形学	**年级：**2018 级	**实验成绩：**
实验名称：二维几何变换算法	**姓名：**董辰尧	
实验编号：6	**学号：**10185102144	**实验日期：**2021-4-13
指导教师：王长波、李洋	**组号：**	**实验时间：**13:00-14:30

一、实验目的

利用操作系统 API 实现基本二维几何变换操作。

二、实验内容与实验步骤

实现基本 2 维矩阵操作（齐次坐标系）

实现平移变换，将 poly 中心移动回中心

实现缩放变换，x 和 y 方向各缩放为原来的 0.99

实现旋转变换，旋转 0.05 度

将 poly 移动回原先位置

三、实验环境

VS2017

四、实验过程与分析

首先需要下载助教写好的框架，并导入。

先补充 geometry.hpp 里面的矩阵相乘算法，具体实现如下

```
Matrix<Rows, _Cols> operator*(const Matrix<_Rows, _Cols> &rhs) {
    static_assert(cols() == rhs.rows(),
                  "Matrix multiplication dimensions do not match");
    const Matrix<Rows, Cols> &A = *this;
    const Matrix<_Rows, _Cols> &B = rhs;
    Matrix<Rows, _Cols> C;
    C.set_zero();
    // @TODO: Implement Matrix multiplication here C = A * B
    for (int i = 0; i < rows(); i++) {
        for (int j = 0; j < rhs.cols(); j++) {
            for (int k = 0; k < cols(); k++) {
                C(i, j) += A(i, k) * B(k, j);
            }
        }

    }
```

使用了 3 层循环，对于 C 的每一行每一列都要做运算。

接下来补充求平移、缩放、旋转的矩阵函数。关于矩阵内容，PPT 上都有。具体代码如下：

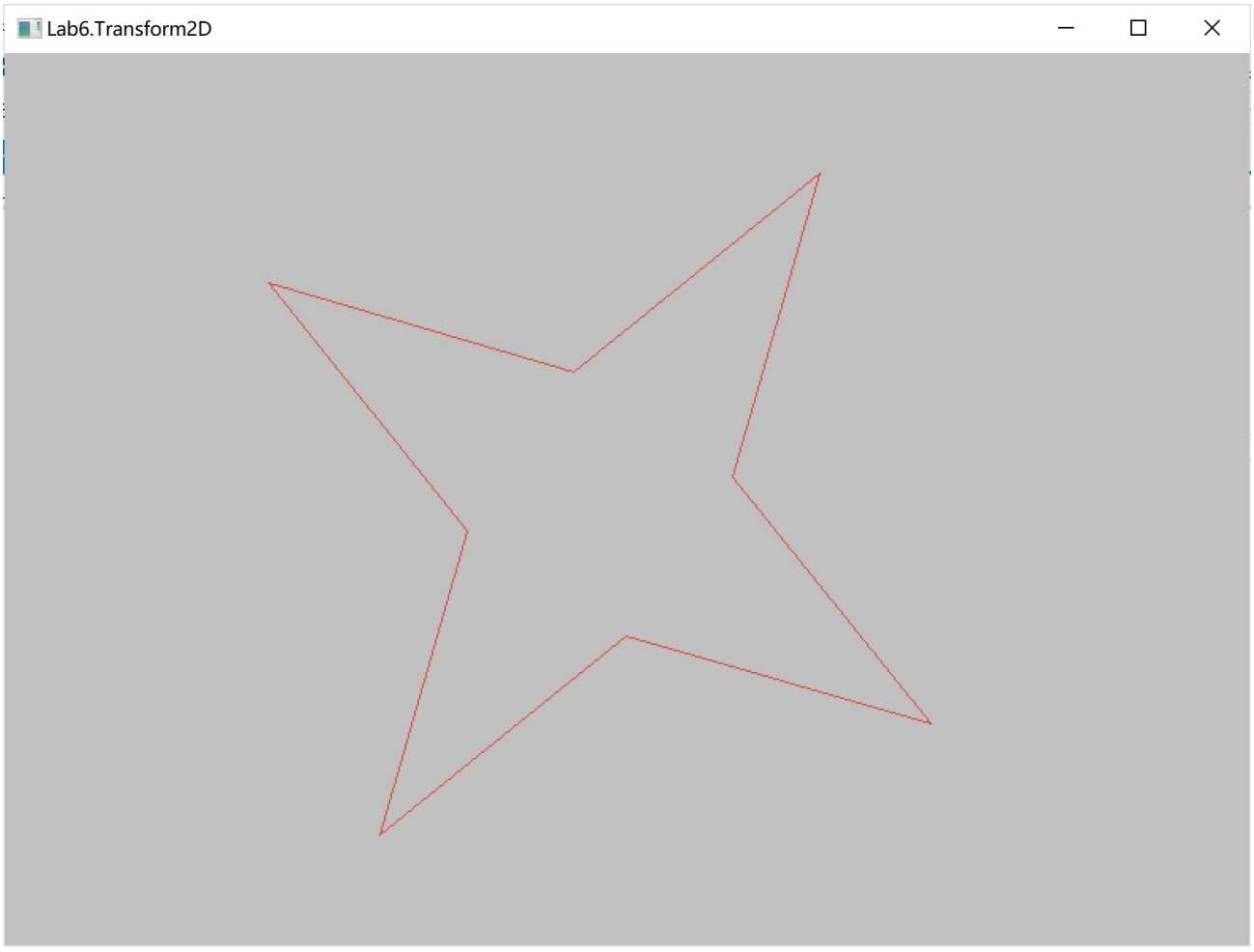
```
#include <cmath>
namespace GEO {

    // @TODO: implement transform basic function in 2D here
    Matrix3x3 rotate_matrix(float alpha) {
        // @Note: Rotate alpha(in radian measure) around the origin O
        Matrix<3, 3> A;
        A.one();

        A = { cos(alpha),sin(alpha),0,-sin(alpha),cos(alpha),0,0,0,1 };
        return A;
    }

    Matrix3x3 translate_matrix(float dx, float dy) {
        // @Note: Move (dx, dy)
        Matrix<3, 3> A;
        A = { 1,0,0,0,1,0,dx,dy,1 };
        return A;

    }

    Matrix3x3 scale_matrix(float sx, float sy) {
        // @Note: scale (sx, sy)
        Matrix<3, 3> A;
        A = { sx,0,0,0,sy,0,0,1 };
        return A;
    }
```

最后去 main 函数里面补充求图像转换矩阵的过程。代码如下：

```
272
273     void DrawWindow::TransformTestPolygon() {
274         // move the origin 0 to the center of star
275         // @TODO: transform polygon with transform_mat
276
277             transform_mat = GEO::translate_matrix(-400, -300);
278             tr                                    e_matrix(0.05);
279             tr                                    _matrix(0.99, 0.99);
280             transform_mat = transform_mat * GEO::translate_matrix(400, 300);
281         for (auto &p : test_poly) {
282             p = p * transform_mat;
283         }
284
285
286     }
```

GEO::Matrix3x3 DrawWindow::transform_mat

五、实验结果总结



本次实验遇到了很多障碍，先是花费了很长时间阅读代码，最后终于弄懂了之后发现自己不会使用这个矩阵。究其原因是我的 C++基础太薄弱。最后终于可以实

现任意的平移、旋转等等变换之后，不知道怎么进行一连串的变换，后来明白连续变换其实就是多个矩阵相乘的结果。最后完成了实验。