Homework 8 – Bezier curve

注：本次作业的完整演示请见/doc/pic/*.gif

Basic:

1. 用户能通过左键点击添加Bezier曲线的控制点，右键点击则对当前添加的最后一个控制点进行消除

2. 工具根据鼠标绘制的控制点实时更新Bezier曲线。

Hint: 大家可查询捕捉mouse移动和点击的函数方法

1. 根据如下Bezier曲线的定义式生成曲线上的点，其中式中Vi是通过鼠标点击产生的控制点，u从0递增至1，递增的速度即为精度。实现求阶乘和求组合数的辅助函数。

$$P_i=\sum_{i=0}^{n}\binom{n}{i}(1-u)^{n-i}u^iV_i$$

```
int fac(int n) {
    if (n == 1 || n == 0)
        return 1;
    for (int i = n - 1;i > 1;i--)
        n = n * i;
    return n;
}
double combination(int n, int i) {
    return fac(n) / (fac(i)*fac(n - i));
}

vector<glm::vec3> BezierGen() {
    vector<glm::vec3> result;
    for (float t = 0; t <= 1; t += 0.001) {
        glm::vec3 temp = glm::vec3(0.0f, 0.0f, 0.0f);
        for (int i = 0; i < control_points_num; i++) {
            temp.x += combination(control_points_num-1, i) * pow((1-t),
              control_points_num-1-i) * pow(t, i) * control_points[i].x;
            temp.y += combination(control_points_num-1, i) * pow((1-t),
              control_points_num-1-i) * pow(t, i) * control_points[i].y;
        }
        result.push_back(temp);
    }
    return result;
```

```
}
```

2. 通过鼠标点击新建/删除控制点的坐标

```
void click_callback(GLFWwindow* window, int button, int action, int mods) {
    glm::vec3 clickPos = standardize(lastX, lastY);
    if (button == GLFW_MOUSE_BUTTON_RIGHT && action == GLFW_PRESS) {
        if (control_points_num >= 1) {
            control_points.pop_back();
            control_points_num--;
        }
    }
    if (button == GLFW_MOUSE_BUTTON_LEFT && action == GLFW_PRESS) {
        control_points.push_back(standardize(lastX, lastY));
        control_points_num++;
    }
}

void mouse_callback(GLFWwindow* window, double xpos, double ypos) {
    lastX = xpos;
    lastY = ypos;
}
```

3. 在渲染循环内绘制所有控制点

```
for (size_t i = 0; i < control_points.size(); i++) {
            float point[] = {control_points[i].x, control_points[i].y,
control_points[i].z};
            glBindVertexArray(VAO);
            glBufferData(GL_ARRAY_BUFFER, sizeof(point), point, GL_STATIC_DRAW);

            glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 3 * sizeof(float),
(void*)0);
            glEnableVertexAttribArray(0);

            glBindBuffer(GL_ARRAY_BUFFER, VBO);
            glBufferData(GL_ARRAY_BUFFER, sizeof(point), point, GL_STATIC_DRAW);
            glPointSize(5.0f);
            glDrawArrays(GL_POINTS, 0, 1);
        }
```

4. 当控制点多于1个时可以确定Bezier曲线，此时画出曲线上生成的点。

```
        if (control_points_num >= 2) {
            vector<glm::vec3> curve = BezierGen();
            for (size_t i = 0; i < curve.size(); i++) {
                float point[] = { curve[i].x, curve[i].y, curve[i].z};
```

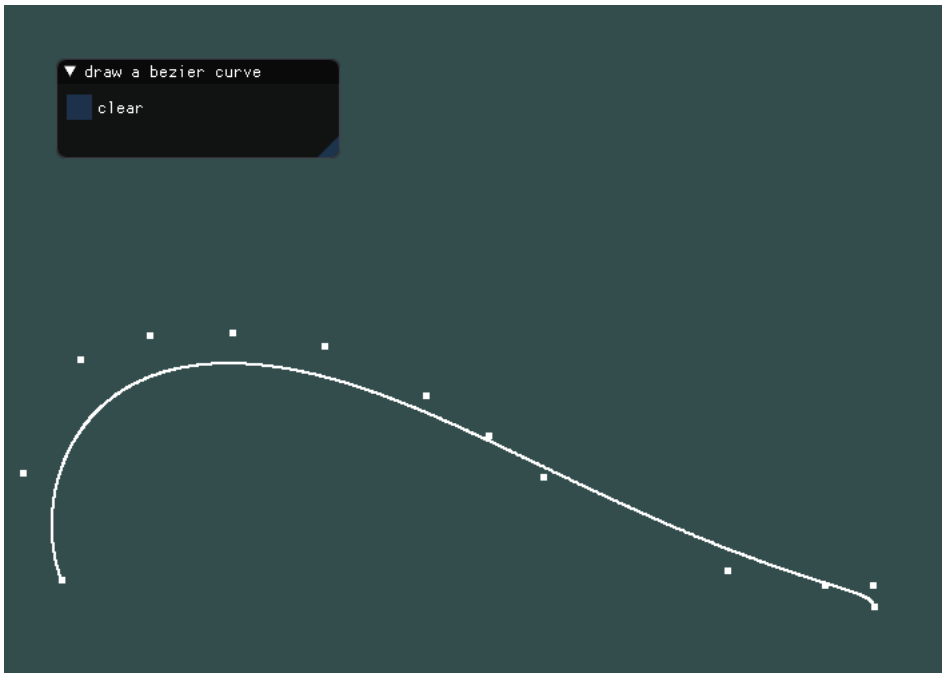
```
                glBindVertexArray(VAO);
                glBufferData(GL_ARRAY_BUFFER, sizeof(point), point,
GL_STATIC_DRAW);

                glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 3 * sizeof(float),
(void*)0);
                glEnableVertexAttribArray(0);

                glBindBuffer(GL_ARRAY_BUFFER, VBO);
                glBufferData(GL_ARRAY_BUFFER, sizeof(point), point,
GL_STATIC_DRAW);

                glPointSize(2.0f);
                glDrawArrays(GL_POINTS, 0, 1);
            }
        }
```

结果示例如图



Bonus：

1. 可以动态地呈现Bezier曲线的生成过程。