Homework 9 - Bezier Curve

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

Bonus:

1. 可以动态地呈现 Bezier 曲线的生成过程。

鼠标控制

按照 TA 给的提示，这次作业首先需要考虑的就是怎么捕捉 mouse 移动和点击的事件。

查阅资料后 发现只需要使用 glfwSetMouseButtonCallback 和 glfwSetCursorPosCallback 设置鼠标点击回调事件和移动回调事件就可以获取到鼠标的点击事件和当前的位置。

```
void mouseMove_callback(GLFWwindow* window, double pos_x, double pos_y) {
    Mouse::getInstance()->pos_x = pos_x;
    Mouse::getInstance()->pos_y = pos_y;
}
void mouseClick_callback(GLFWwindow* window, int button, int action, int mods) {
    Mouse::getInstance()->isPress = action == GLFW_PRESS;
    Mouse::getInstance()->left = (button == GLFW_MOUSE_BUTTON_LEFT);
}
```

代码也是大同小异，简单几行。

绘制 Bezier 曲线

Bezier 曲线可以用数学公式表示：

$$Q(t)=\sum_{i=0}^{n}P_iB_{i,n}(t),\quad t\in[0,1]$$

其中 $B_{i,n}(t)$为下式

$$B_{i,n}(t)=\frac{n!}{i!(n-i)!}t^i(1-t)^{n-i},\quad i=0,\ 1\ldots n$$

首先，计算 Bernstein 基函数需要多次用到阶乘，这里将每次阶层的结果记录下来，我们使用 long long int 存储这个比较大的数字，但依旧只能计算到 20 的阶乘，所以我们最多也就只能画出 20 个点。

```
// 计算阶乘
long long int getFactorial(int i) {
  if (i > 104) return 1;
  for (; computedFac <= i; computedFac++) {
    factorialTmp[computedFac] = factorialTmp[computedFac - 1] * computedFac;
  }
  return factorialTmp[i];
}
```

```
// Bezier函数
float Bezier(int i, int n, float t) {
    if (n >= POINT_MAX || i >= POINT_MAX || n - i >= POINT_MAX) {
        overflow = true;
        return 0.0f;
    }
    long long int n_result = multiple_result[n];
    long long int i_result = multiple_result[i];
    long long int ni_result = multiple_result[n - i];

    float alpha = n_result / i_result / ni_result;
    return (alpha * pow(t, i) * pow(1 - t, n - i));
}
```

画出曲线和辅助线

```
void draw(int count, bool helping) {


    glGenVertexArrays(1, &VAO);
    glGenBuffers(1, &VBO);
    glBindBuffer(GL_ARRAY_BUFFER, VBO);
    glBufferData(GL_ARRAY_BUFFER, sizeof(float) * 6 * count, buff_point, GL_STATIC_DRAW);
    glPointSize(6);
    glBindVertexArray(VAO);

    glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 6 * sizeof(float), (void*)0);
    glEnableVertexAttribArray(0);
    glVertexAttribPointer(1, 3, GL_FLOAT, GL_FALSE, 6 * sizeof(float), (void*)(3 * sizeof(float)));
    glEnableVertexAttribArray(1);
    glBindBuffer(GL_ARRAY_BUFFER, 0);
    glDrawArrays(GL_POINTS, 0, count);
    if (helping) glDrawArrays(GL_LINE_STRIP, 0, count);
    glBindVertexArray(0);
    glDeleteVertexArrays(1, &VAO);
    glDeleteVertexArrays(1, &VBO);
```

流程和前面的作业一样 ，这里的参数 count 是点的个数，鼠标点击后 draw 一次。

效果图