中国海洋大学

OCEAN UNIVERSITY OF CHINA

海纳百川 取则行远

计算机图形学

Lab4 实验报告

实验题目:	Bezier 曲线
姓　　名:	翟一航
年　　级:	2023 级
专　　业:	软件工程
实验时间:	2025 年 12 月 1 日

目录

1 实验要求

在本次实验中，你需要实现 de Casteljau 算法来绘制由 4 个控制点表示的 Bézier 曲线。你需要修改的函数在提供的 main.cpp 文件中：

bezier：该函数实现绘制 Bézier 曲线的功能。

recursive_bezier：该函数使用一个控制点序列和一个浮点数 t 作为输入，实现 de Casteljau 算法来返回 Bézier 曲线上对应点的坐标。

提高项：实现对 Bézier 曲线的反走样。对于一个曲线上的点，不只把它对应于一个像素，你需要根据到像素中心的距离来考虑与它相邻的像素的颜色。

2 实验中遇到的问题与解决方法

问题一： 直接使用实验 1-3 的头文件进行项目构建，出现“No index file found.”的报错。

解决方案：

这是因为本实验需要使用 OpenCV 包，但是在原头文件中并没有这个包，需要在 CmakeList.txt 文件中写入对该包的查找：

find_package(OpenCV REQUIRED)

问题二： 在运行程序时，出现无限递归的现象，最终导致栈溢出。

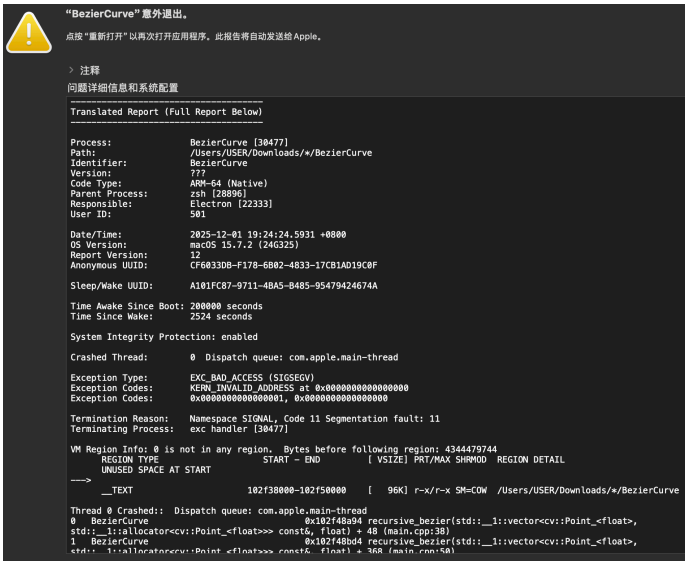

图 1: 栈溢出错误

解决方案：

检查后发现，问题发生在 recursive_bezier 函数中，在编写算法的终止条件错误，将终止条件写为：

control_points.empty()

这就导致没有正确处理控制点为 1 的情况，在控制点数量为 1 时，原本应该返回此时贝塞尔曲线在参数 t 处的精确坐标，但是如果条件写为检查控制点数组是否为空，就会导致在控制点数量为 1 时也会进入循环，出现无限递归的现象，应该将终止条件修改为：

control_points.size() == 1

问题三： 进行反走样优化后的曲线不连续

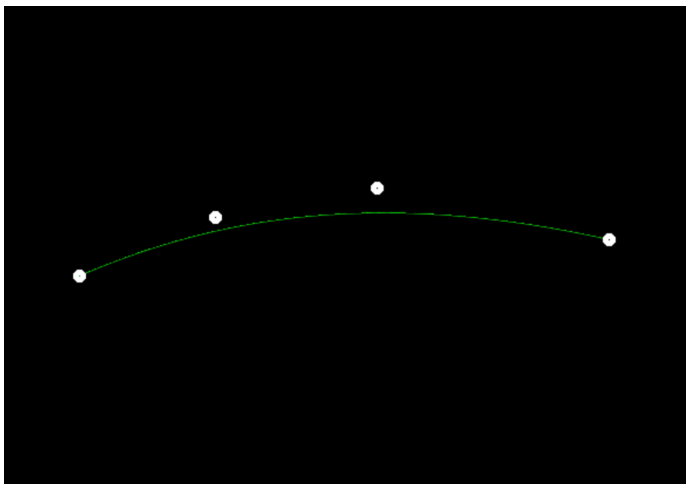

图 2: 反走样后不连续的曲线

解决方案：经查询后得知，这是因为浮点数计算精度误差导致本应为零的权重值变为微小的负值，被严格的 weight > 0 条件过滤掉，造成曲线边缘像素丢失。应该插入：

weight = std::max(0.0f, weight)

在比较前钳制权重值，避免浮点误差导致的负值被过滤掉。

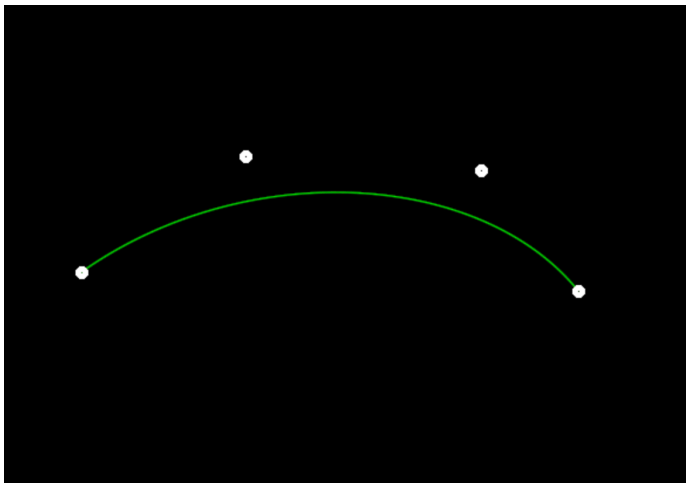

图 3: 正确反走样处理后的贝塞尔曲线

3 心得体会

通过本次实验，我对贝塞尔曲线的数学原理与计算机实现有了更加深入的理解，最终实现了完整的反走样效果。通过此次实验，我对于老师上课讲解的贝塞尔曲线的两种求法有了更深的掌握与理解，除此之外，还学会了反走样在提升贝塞尔曲线质量中的应用。总的来说，此次实验中我的收获巨大，通过完成实验基础内容与解决实验中遇到的问题，我深刻体会到理论知识与工程实践之间的紧密联系。