HW2-CAGD

阚皓玮

2020 年 10 月 9 日

1 使用说明

在 matlab 中运行 Hw2.m 脚本文件，出现交互图窗。工具栏中点击红色按钮添加点，蓝色按钮 De Casteljau 算法得到 Bézier 曲线，绿色按钮 Bernstein 基表示得到 Bézier 曲线。

2 实验内容

实现基于

1. De Casteljau 递归算法
2. Bernstein 基函数的代数方法

的 Bézier 曲线生成，并且实现通过控制多边形获得 Bézier 曲线的交互式编辑功能。

3 算法介绍

Input: 控制多边形的点 $\{b_0, b_1, ...b_n\}$

Output: Bézier 曲线 $x(t)$，其中 $t \in [0, 1]$

下面介绍两种获得 Bézier 曲线的算法:

De Casteljau 递归算法

对于 $t \in [0, 1]$，我们按照如下步骤计算 $x(t)$

1. 将控制多边形的边每一条边按照 t:(1-t) 的比例分割得到相应的点
2. 将新的点用按顺序用线连接
3. 再次将新得到的线按 1 中比例连接分割得到新的点，并将新的点按顺序连接
4. 按照 3 中方式重复以上步骤，直到只剩下一个点，即为 $x(t)$

基于 Bernstein 基函数的代数表示方法

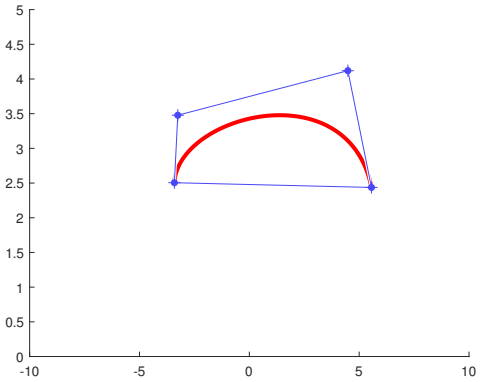
基于 Bernstein 基函数，我们可将 $x(t)$ 按如下方式表示

$$x(t) = \sum_{i=0}^{n} B_n^i(t) b_i$$

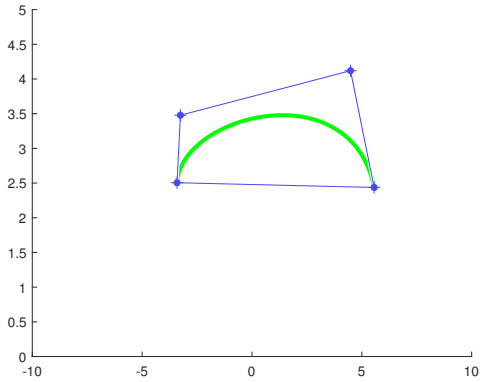
其中 $B_n^i(t) = \binom{n}{i} t^i (1-t)^{n-i}$

4 实验结果

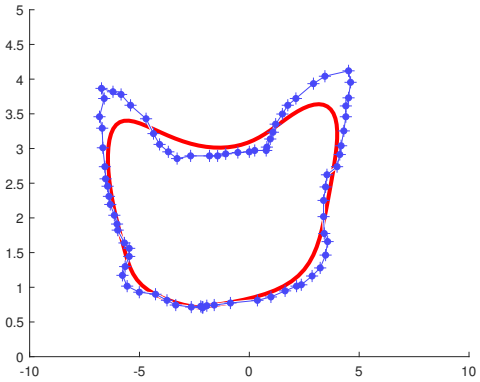
实验交互选点得到 Bézier 曲线结果如下：



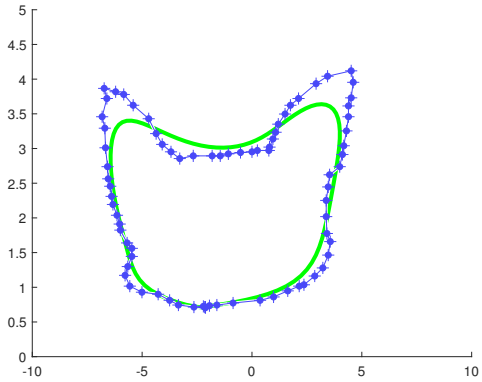
(a) De Casteljau



(b) Bernstein



(c) De Casteljau



(d) Bernstein

5 结果分析及比较

两种算法获得的曲线是相同的，理论上 Bernstein 基函数的表示方法更易于表示且复杂度更低，但是实际实验过程中 De Casteljau 算法的效率更高，这可能是由于 matlab 中计算 Bernstein 函数的速度较慢。另一个基于 Bernstein 基函数的表示方法的缺点是，当控制点的数量较大时，Bernstein 基函数中组合数的值较小，可能会超出计算机的精度范围，造成结果的不准确。因此在实际应用过程中，De Casteljau 算法可能会是更好的选择

A Code

完整代码可从https://github.com/mathendy/MS-USTC/tree/master/2020fall/CAGD/Hw2下载