Assignment 01: Bezier Curves

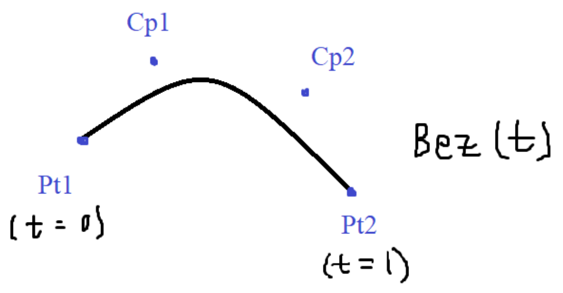
Documentation

Tanishq Jain, 2021294

Introduction

Piecewise Bezier curves provide good local control and C1 continuity at joins. In the given code, a sequence of left mouse clicks on the application canvas are used to draw a piecewise cubic curve connecting the points. Right mouse click ends adding of points and thereafter left mouse clicks can be used to select and move the control points.

Strategy to implement the interpolating piecewise Cubic Bezier curves and to enforce C1 continuity



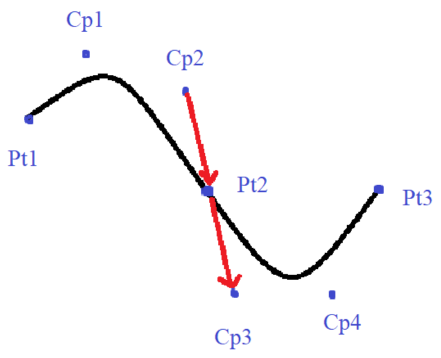
Suppose we are given points Pt1 and Pt2, and we have to implement a cubic Bezier curve interpolating these points. We can simply introduce additional control points Cp1 and Cp2 (randomly), and for parameter t,

$$Bez(t) = (1-t)^3\,(Pt1) + 3(1-t)^2 t\,(Cp1) + 3(1-t)t^2\,(Cp2) + t^3\,(Pt2)$$

Hence, for a sequence of N points Pt1, Pt2 ... PtN, we can interpolate them using piecewise cubic Bezier curves between every 2 simultaneous points.

But in order to make our interpolating curve smooth, we need to enforce C1 continuity within our piecewise cubic Bezier curves.

2 piecewise Bezier curves Bez1(t) and Bez2(t) follow C1 continuity if they are connected (C0 continuity) and their tangents (direction, magnitude) are identical at the joining point.



Now, suppose we have to interpolate given points Pt1, Pt2 and Pt3. We can implement Bez1(t) interpolating Pt1 and Pt2 as described above. While introducing control point Cp3 for implementing Bez2(t) interpolating Pt2 and Pt3, we need to enforce C1 continuity.

$$\frac{dBez(t)}{dt} = -3(1-t)^2(Pt1) + 3((1-t)^2 - 2t(1-t))(Cp1) + 3(2t(1-t) - t^2)(Cp2) + 3t^2(Pt2)$$

$$\frac{dBez1(t)}{dt}_{|\, t=1} = 3(Pt2 - Cp2) \ and \ \frac{dBez2(t)}{dt}_{|\, t=0} = 3(Cp3 - Pt2)$$

Hence, to get position of Cp3, we store the vector from Cp2 to Pt2 and add it to Pt2. There is no such constraint on choice of Cp4. In this way, we can implement both piecewise cubic Bezier curves and they follow C1 continuity. The method can be extended to interpolating a sequence of N points.

Implementation

pf – alias used for type pair<float, float>

pt1 and pt2 – current and next control point

cp1 and cp2 – control points to be introduced

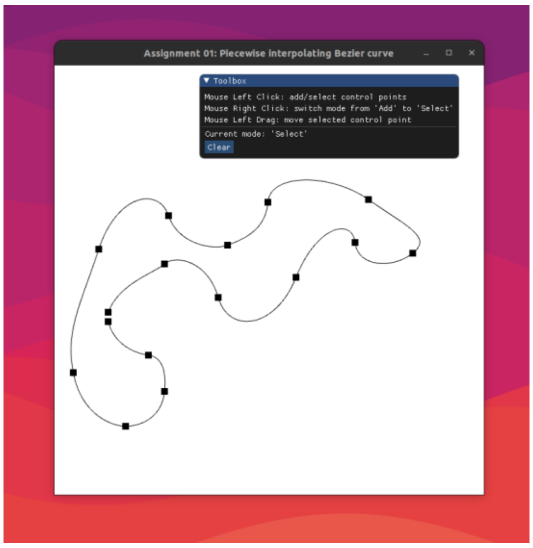
constraint – represents whether there is C1 constraint on position of cp1? (always TRUE after 1st point)

introduceCPs() – sets values for cp1, cp2 in region near pt1, pt2

lastTangent – holds the tangent vector at joining point for previous curve segment

deltaX and deltaY – horizontal and vertical distance between pt1, pt2 (if value is 0, set to 0.5 so that control points are not collinear – ensures cubic Bezier curve)

getPoint() – computes co-ordinates of point bezT having parameter value t on Bezier curve with control points pt1, cp1, cp2, pt2 (in order)


Assignment 01: Piecewise interpolating Bezier curve

▼ Toolbox

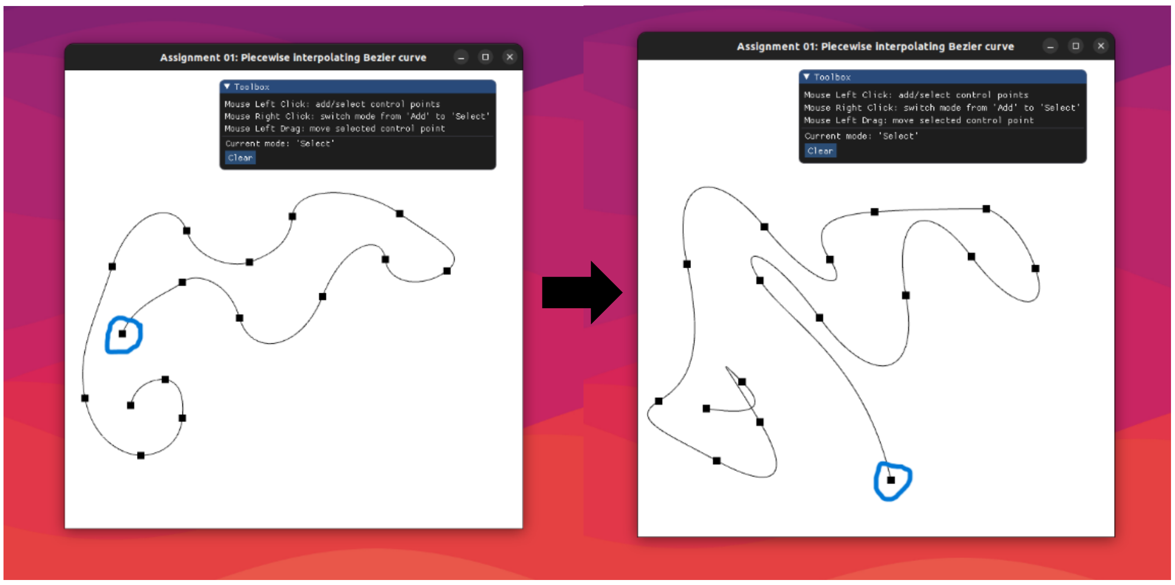
```
Mouse Left Click: add/select control points
Mouse Right Click: switch mode from 'Add' to 'Select'
Mouse Left Drag: move selected control point
Current mode: 'Select'
```

Clear

Whenever a control point is added (in 1st mode) or dragged (in 2nd mode), flag controlPointsUpdated is set to TRUE. It updates VAO/VBO for the piecewise cubic Bezier curve allowing us to draw the curve from points taken progressively.

Modifying the position of a control point updates all piecewise cubic Bezier curves following it in order to satisfy the C1 continuity constraints.

Figure below demonstrates the same by modifying the position of the 1st control point.


Assignment 01: Piecewise interpolating Bezier curve

▼ Toolbox

```
Mouse Left Click: add/select control points
Mouse Right Click: switch mode from 'Add' to 'Select'
Mouse Left Drag: move selected control point
Current mode: 'Select'
```

Clear