A 2D parameterized-curve discretization algorithm

Jianhua Xu

jhxu.cn@gmail.com

2022-05-11

Problem Statement

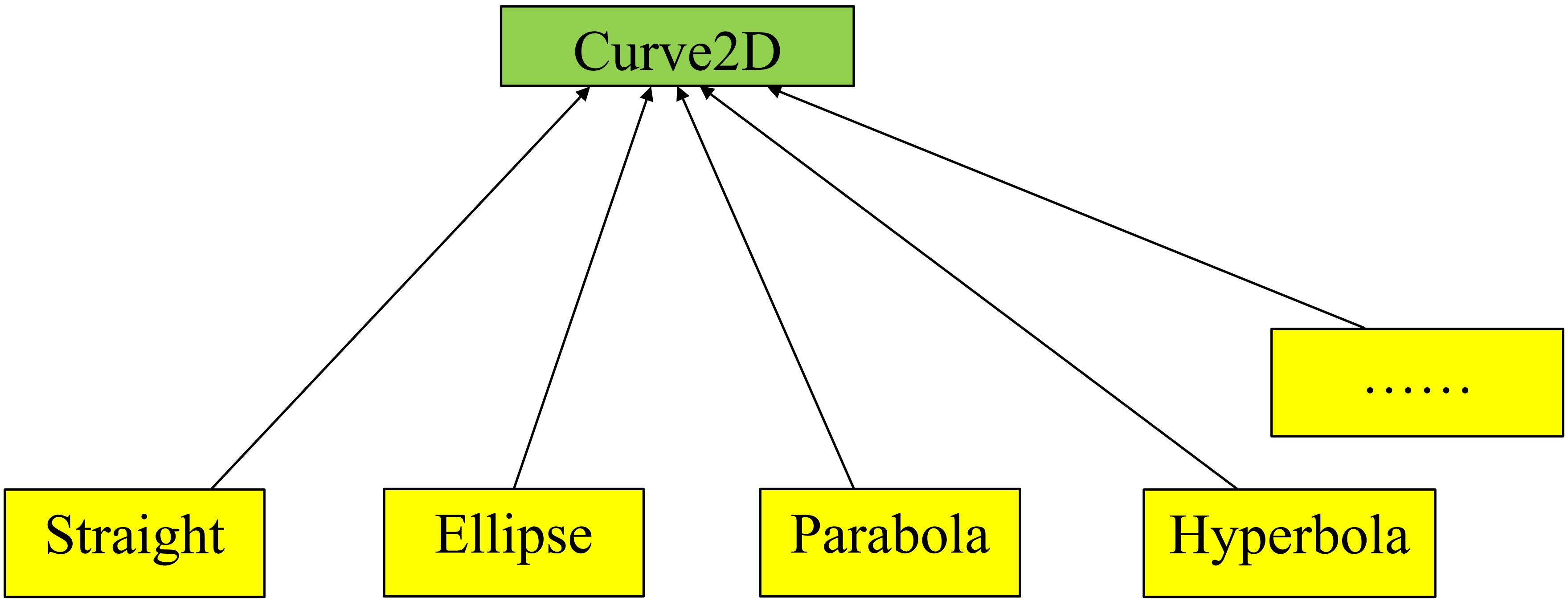
Constraints:

1. Must not exceed a discrete cartesian acceleration (length/time^2). Which is defined as an input threshold;
2. Must start and end with a velocity of 0;
3. The section of path is a parameterized curve. Must start at the curve parameter of 0 and end at the parameter of 1.

Goal:

Given an arbitrary 2D curve, output an array of parameters meet the constraints criteria. The array size should be as small as possible.

2D Curves



This diagram shows class relationships for different types of curves, more types can be added, depending on real applications.

Algorithm and Flow

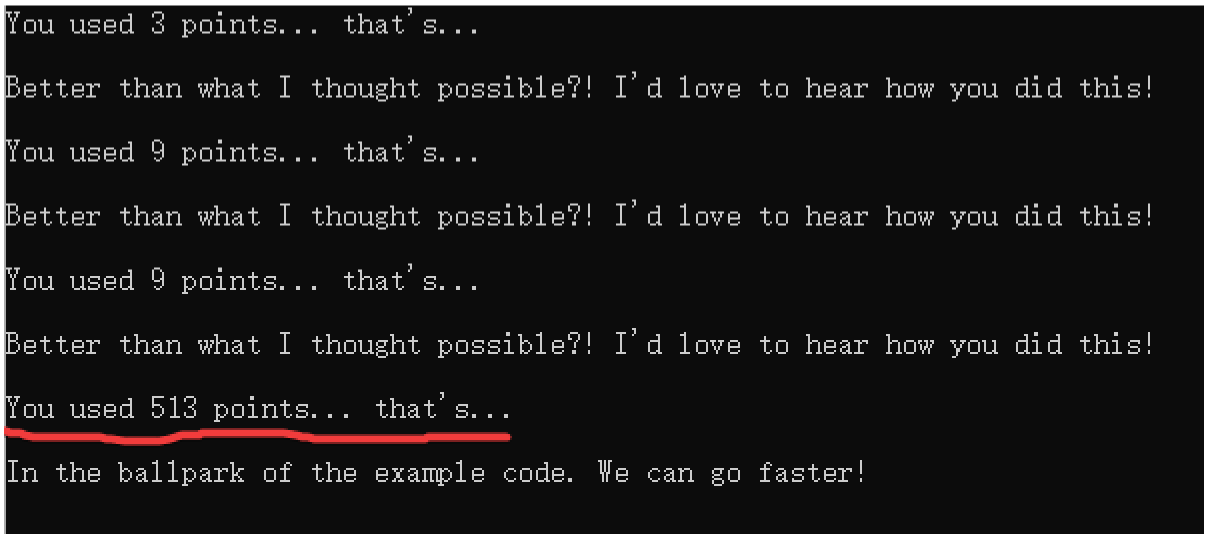
Divide and Conquer:

1. Initially t0 = 0, t1 = 1, mid_t = (t0+t1)/2, add t0 into output;
2. Calculate the acceleration of t0, mid_t, and t1, if current acc < acc_threshold, add mid_t and t1 into output, otherwise;
3. Recursively check [t0, mid_t), [mid_t, t1] until all acc < threshold, or recursion depth == 50 (prevent infinite recursion);
4. Get output parameters.

Note: based on the acc formula you provided, this algorithm works because each split will strictly decrease acc. It runs in linear time with the number N of parameters added, which is O(N), space complexity is same.

Test cases

1. Straight line: ptStart(0, 0), ptEnd(1, 1);
2. Ellipse: ptCenter(1.0, 1.0), dRadiusX = 3, dRadiusY = 2;
3. Circle: ptCenter(1.0, 1.0), dRadiusX = 5, dRadiusY = 5;
4. Ellipse: ptCenter(1.0, 1.0), dRadiusX = 10000, dRadiusY = 20000; – this is a big ellipse, I believe in real cases, such long curve will be split into smaller sections.



Future work

- **Different strategies can be experimented for point sampling:**
  - Pre-add some critical points, for example, the extreme points of a curve, then apply my algorithm piece-wisely;
  - Take curvature into account, add more points in the high-curvature areas;
- **More precisely control the points:**
  - Pre-define a size map (point density) along the curve, add points based on this size map.
- **Support more types of 2d curves:**
  - Currently only ellipse and straight are supported;
  - This method can be extended to 3D curves (x(t), y(t), z(t)).

Future work

- **Thank you!**