

Render the Possibilities

SIGGRAPH 2016



THE 43RD INTERNATIONAL
CONFERENCE AND EXHIBITION ON

 Computer Graphics
Interactive Techniques

24-28 JULY

ANAHEIM, CALIFORNIA

Render the Possibilities

SIGGRAPH 2016



THE 43RD INTERNATIONAL
CONFERENCE AND EXHIBITION ON



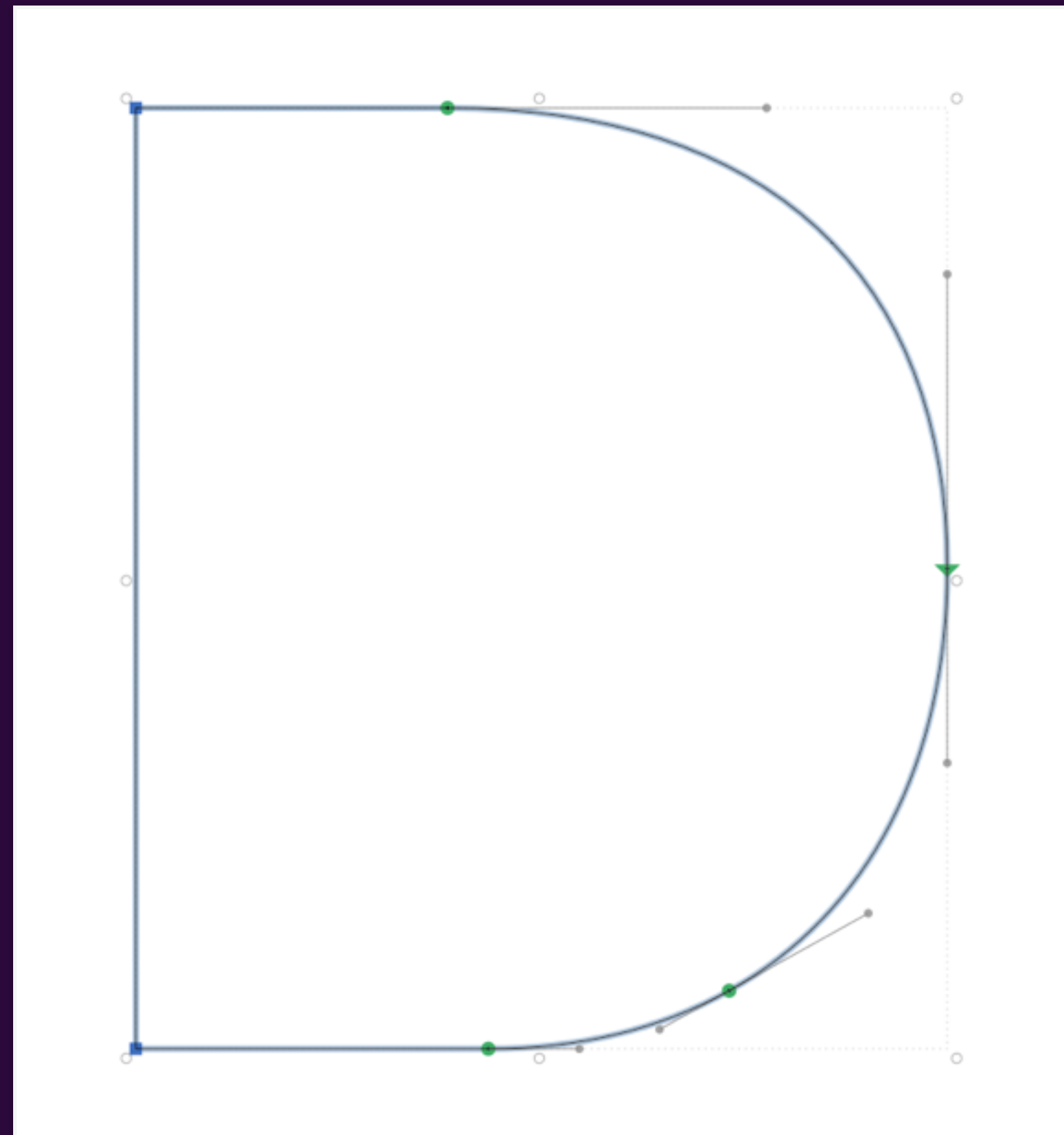
Computer Graphics
Interactive Techniques

ARM

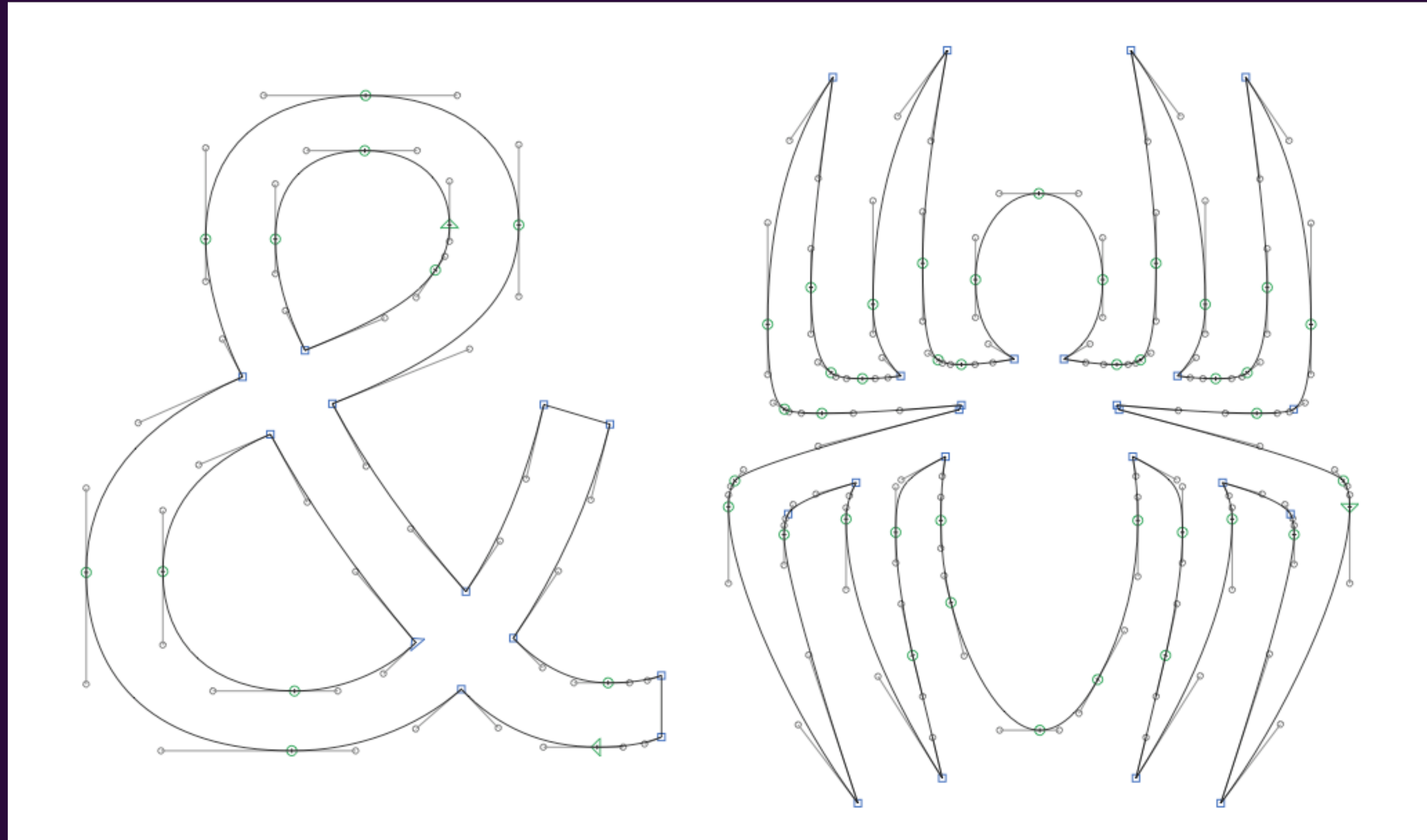
Practical Analytic 2D Signed Distance Field
Generation

Wasim Abbas
ARM - Staff Engineer

Path



Path



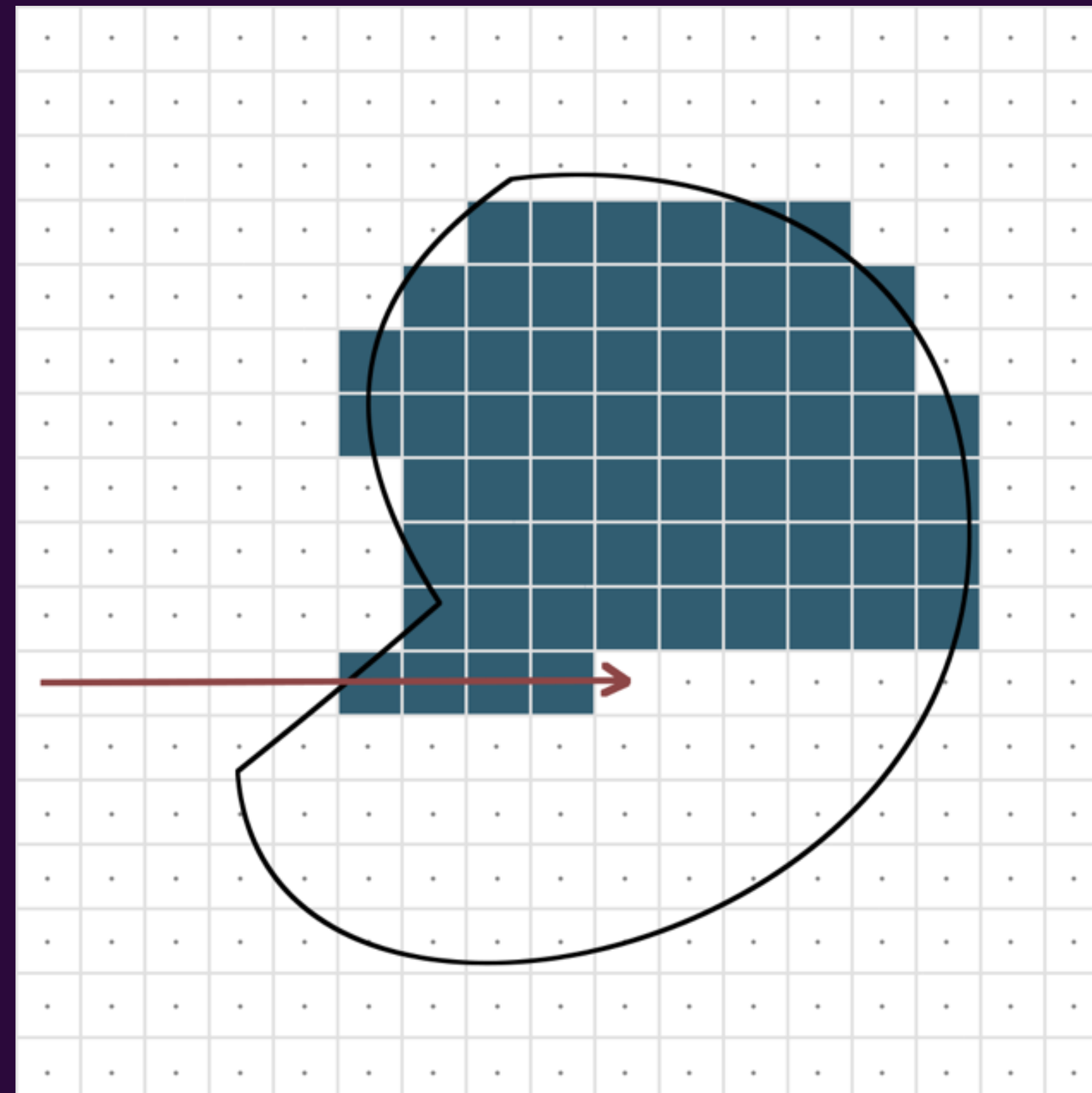
Path



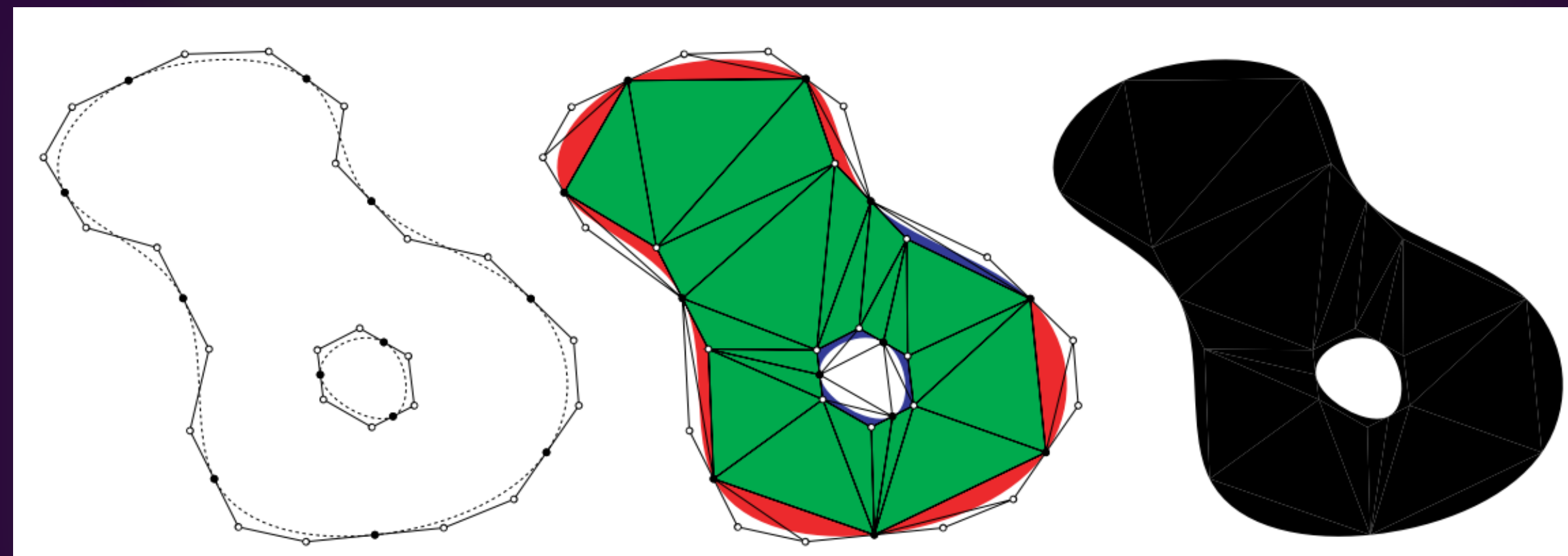
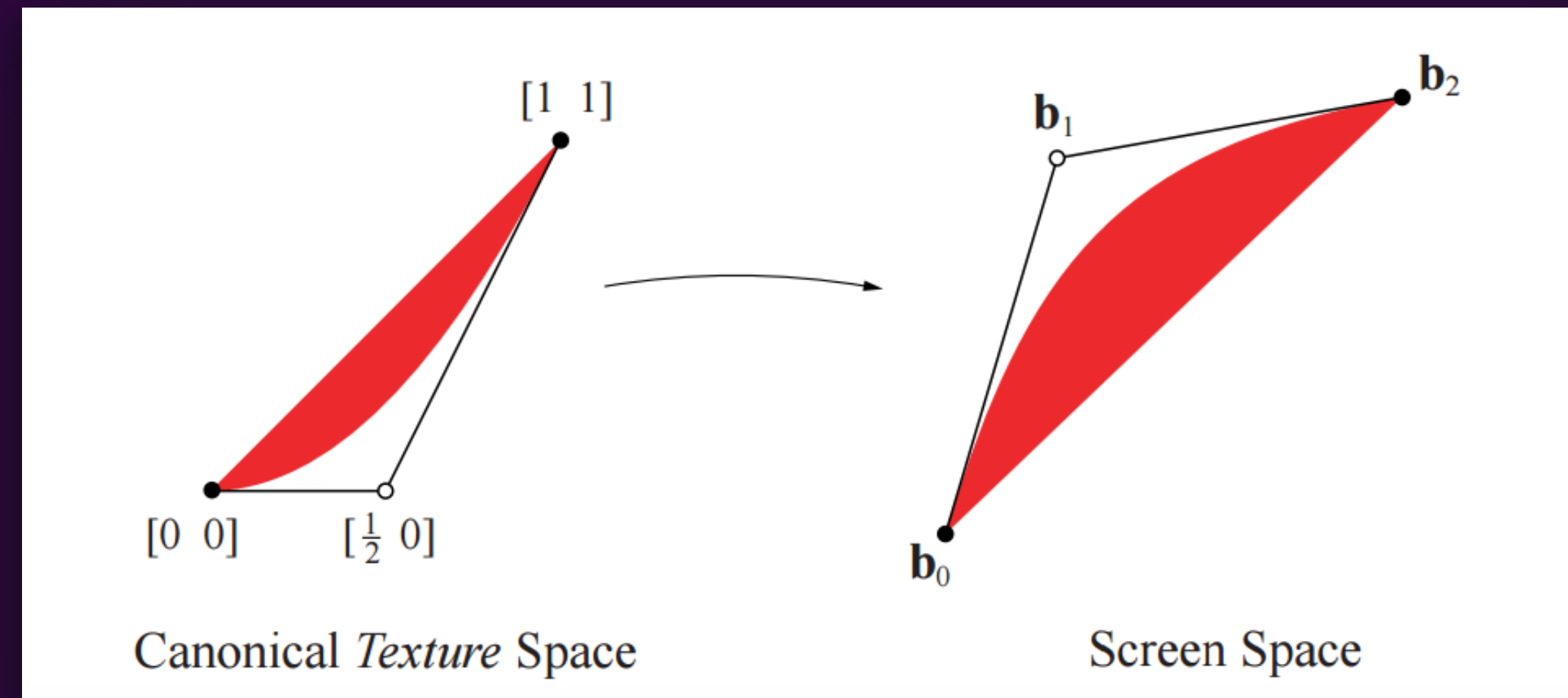
Outline

- What is the problem?
- What is our solution?
- Results and conclusion

Rendering a Path

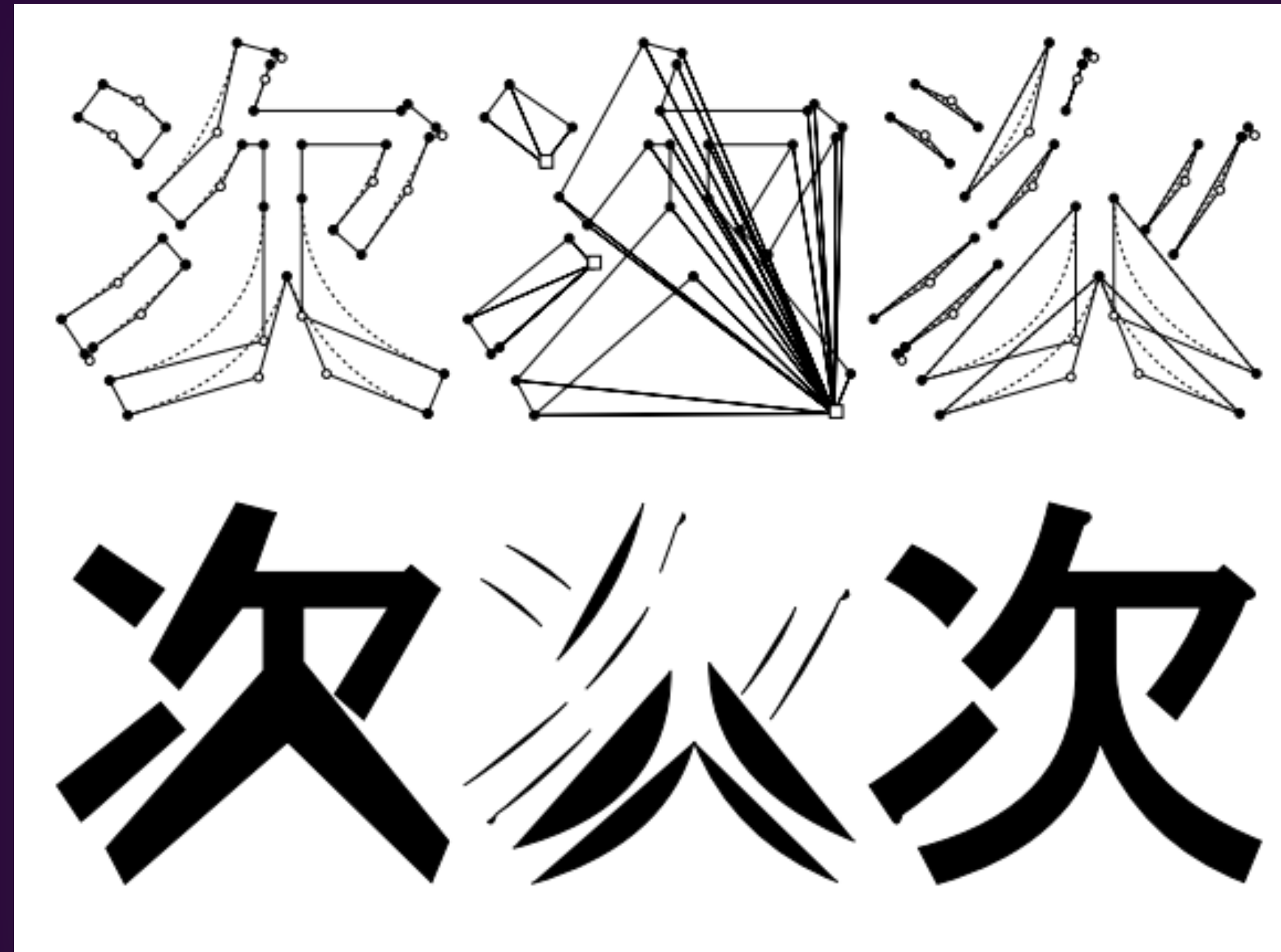


Rendering a Path



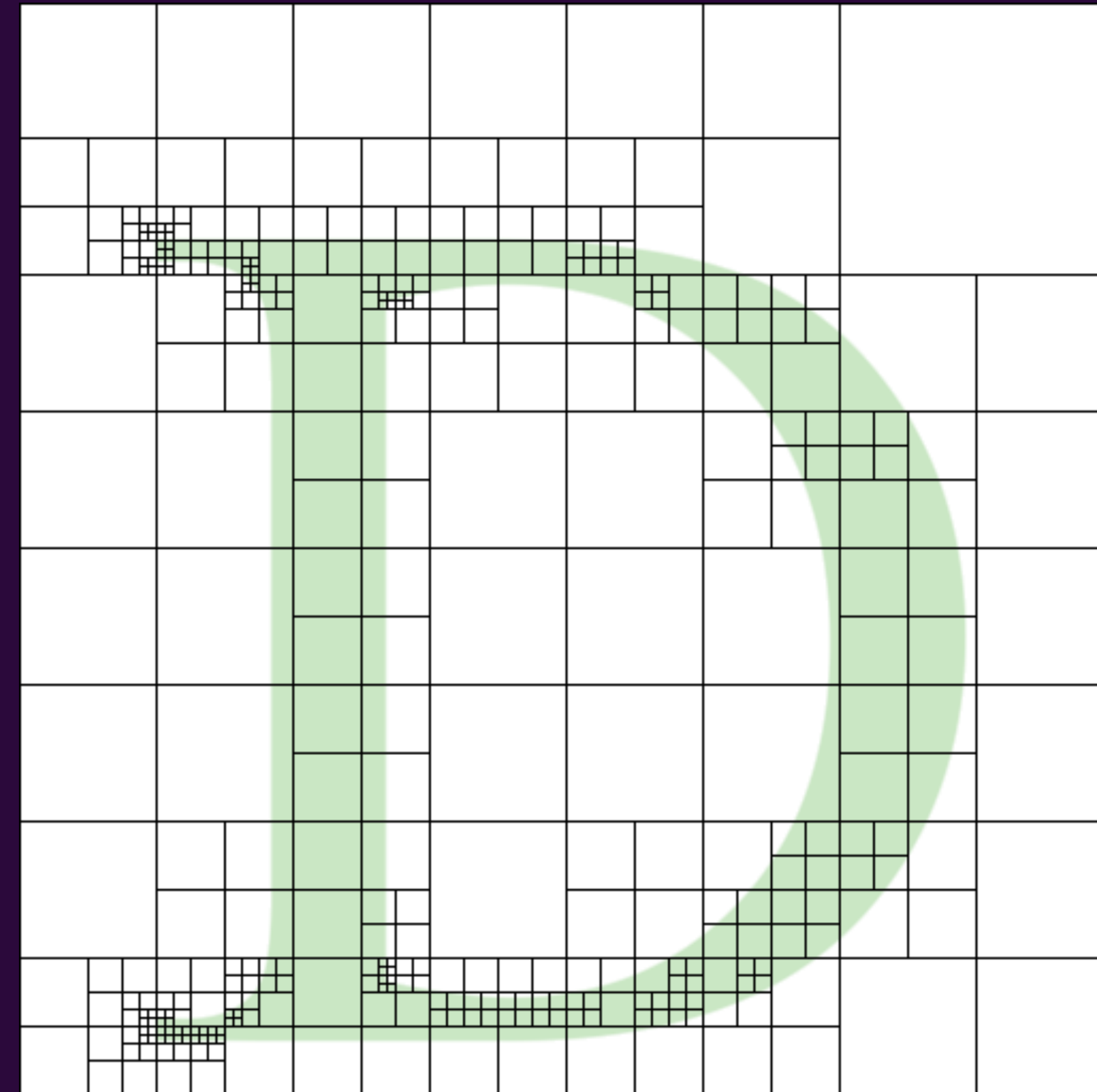
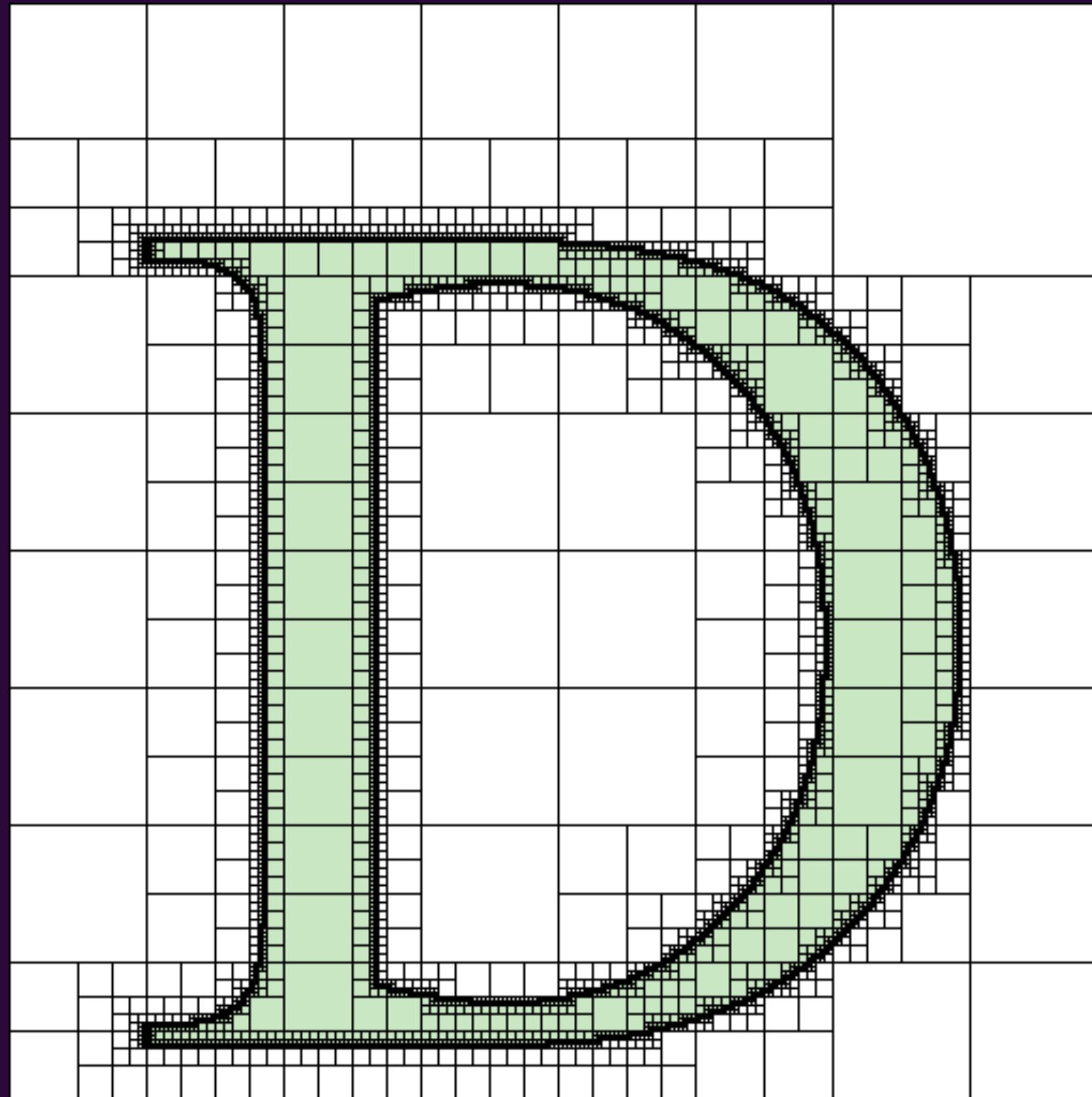
C. Loop, J. Blinn
Microsoft Research 2005

Rendering a Path



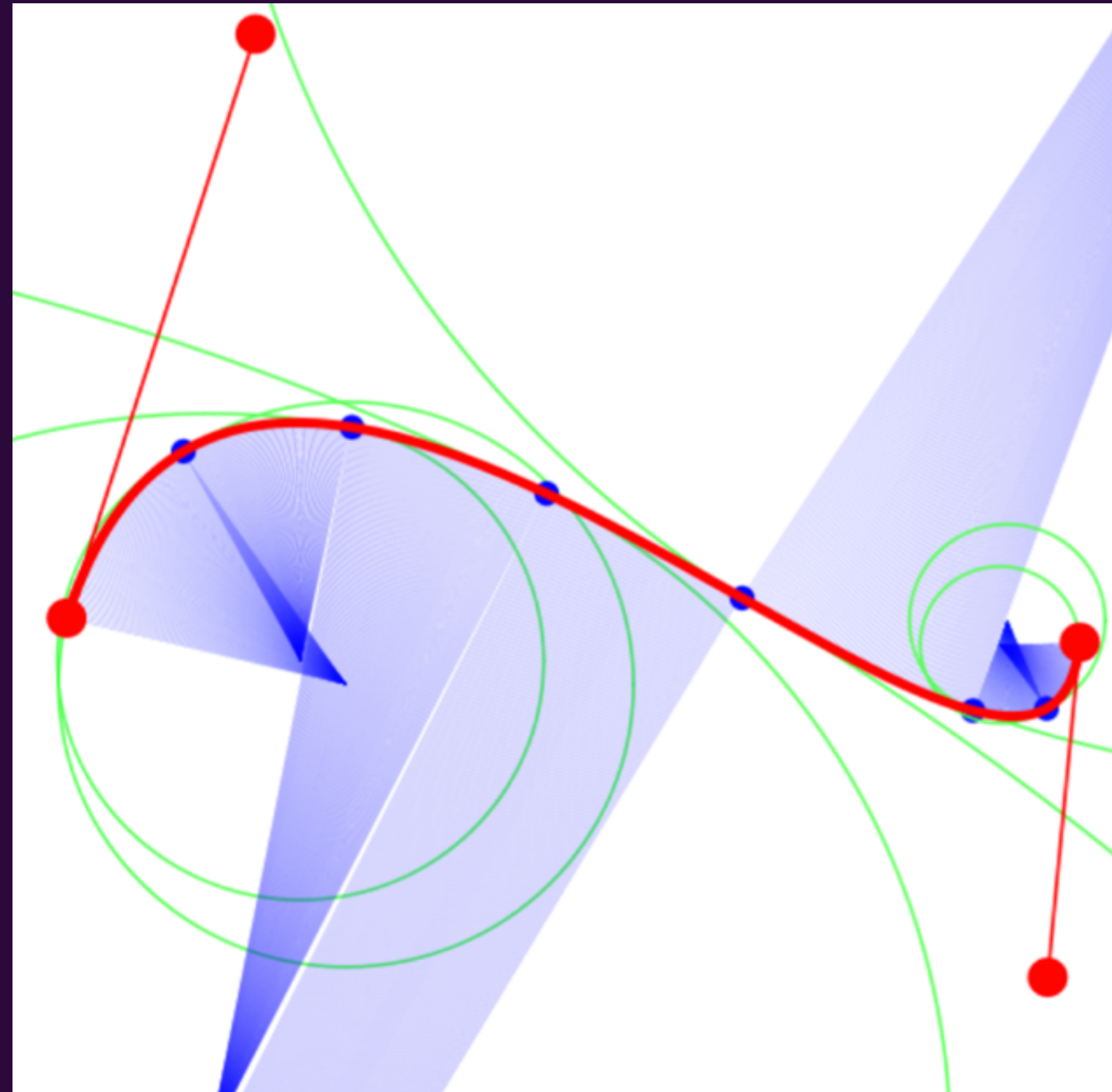
Y. Kokojima, K. Sugita, T. Saito, T. Takemoto
Toshiba 2006

Rendering a Path



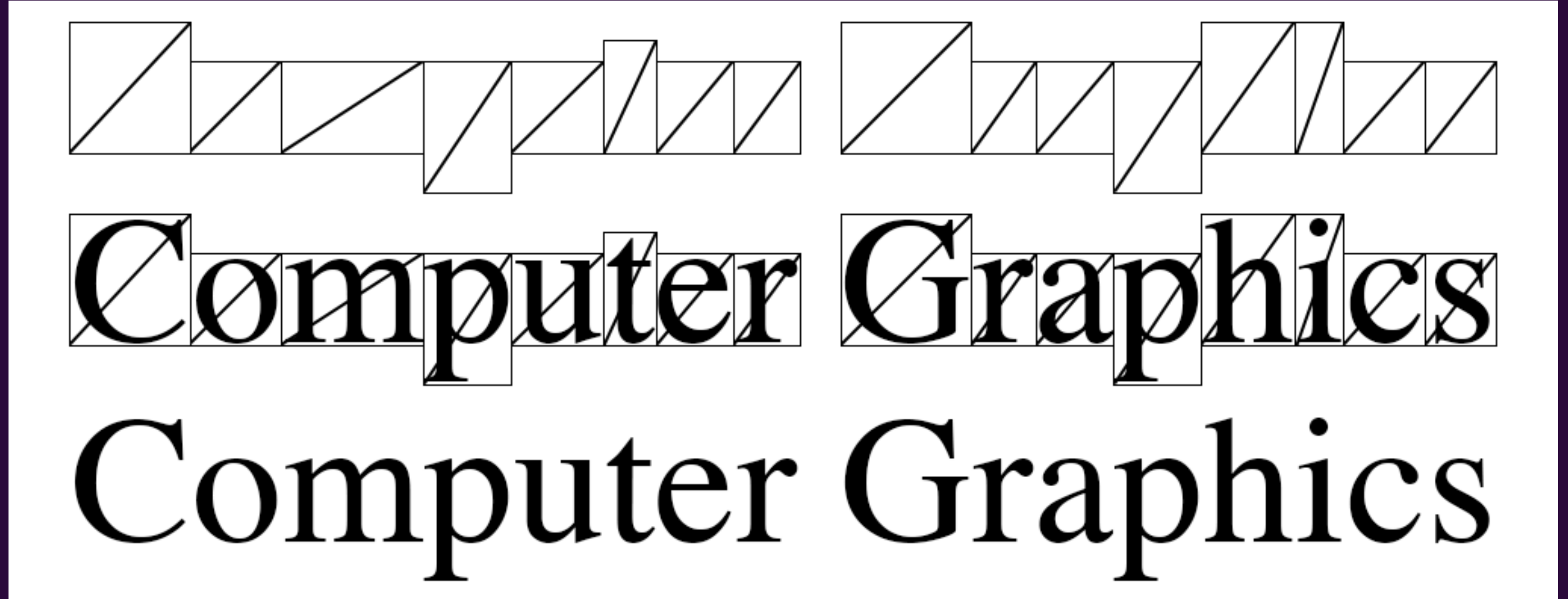
S. Frisken, R. Perry, A. Rockwood, T. Jones
Mitsubishi Electric Research Laboratory 2000

Rendering a Path

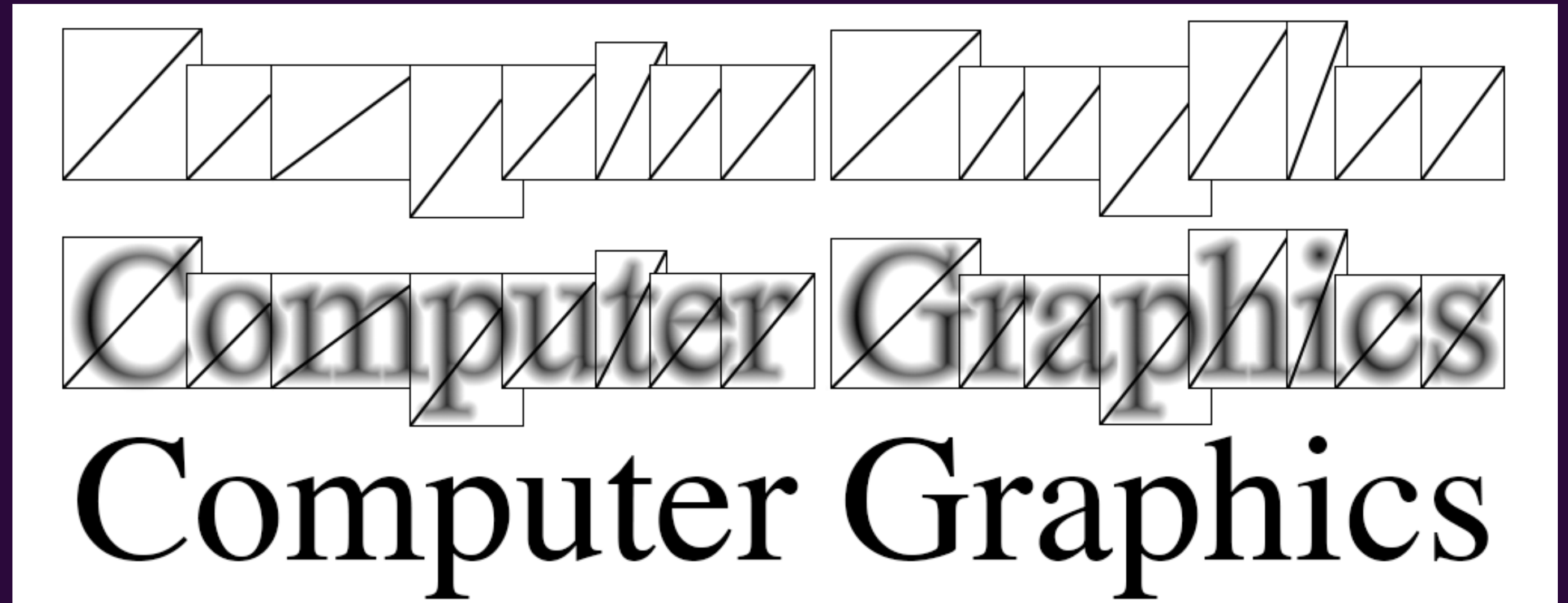


Rendering a Path

} " & [< >] , % {
? @ 3 6 \$ (8 9 # 4 5 2 0 !
X Z U Y W I V S T
N R M Q P O L K
H F J D G C E I A B
S Z t y r u v x . w o p q
. a b c d e f g h i j k l m n



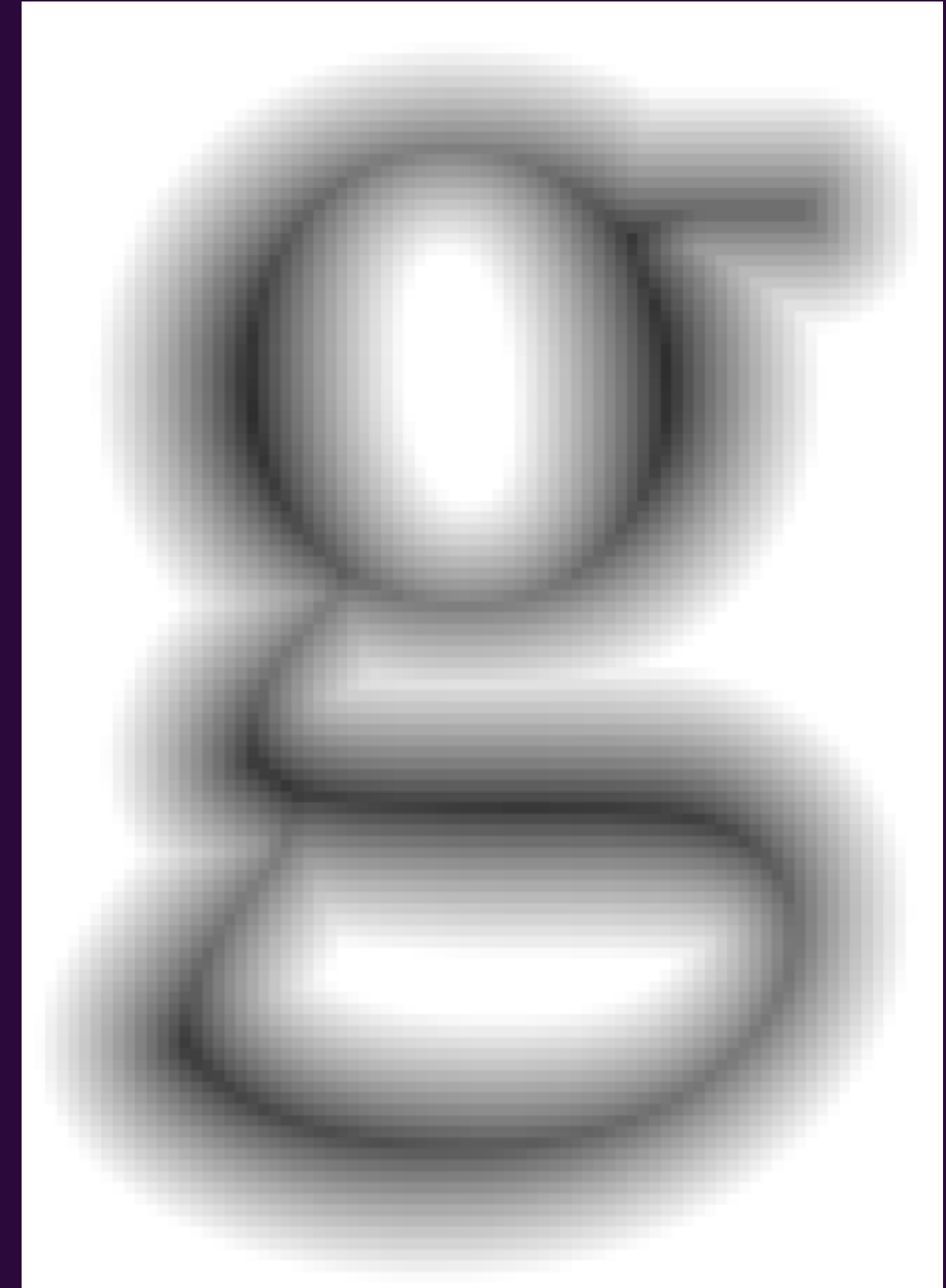
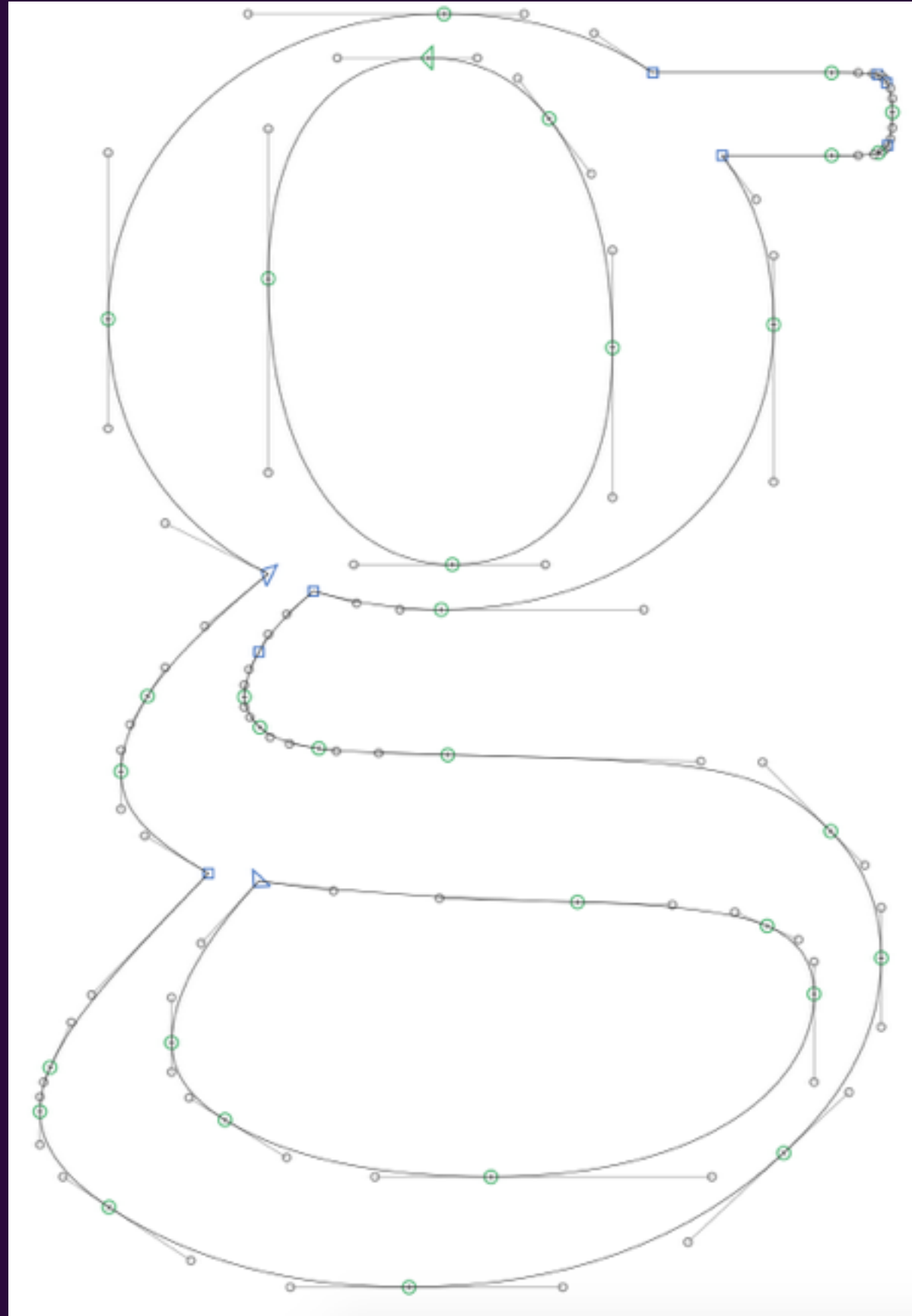
Rendering a Path



Rendering a Path

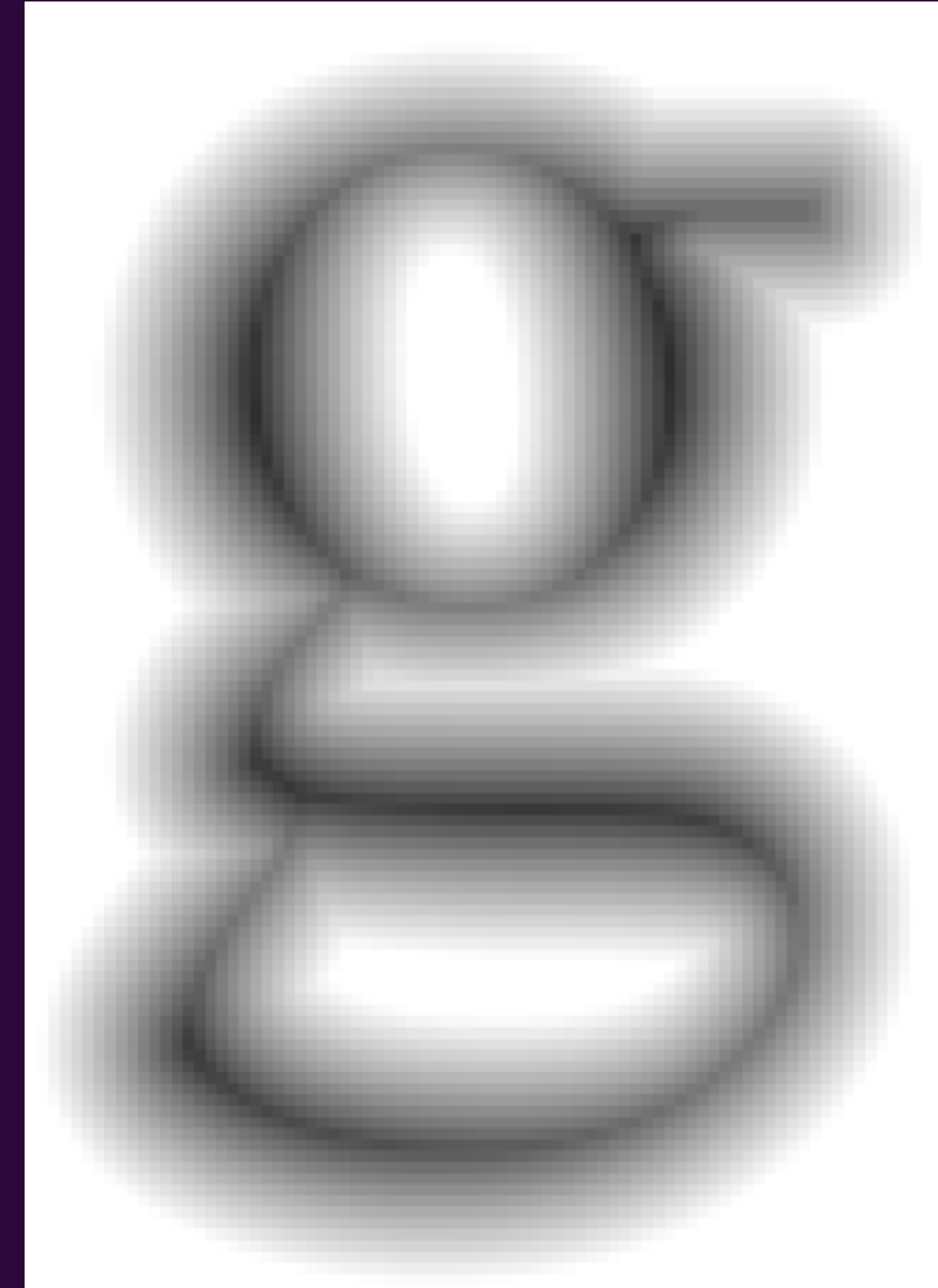
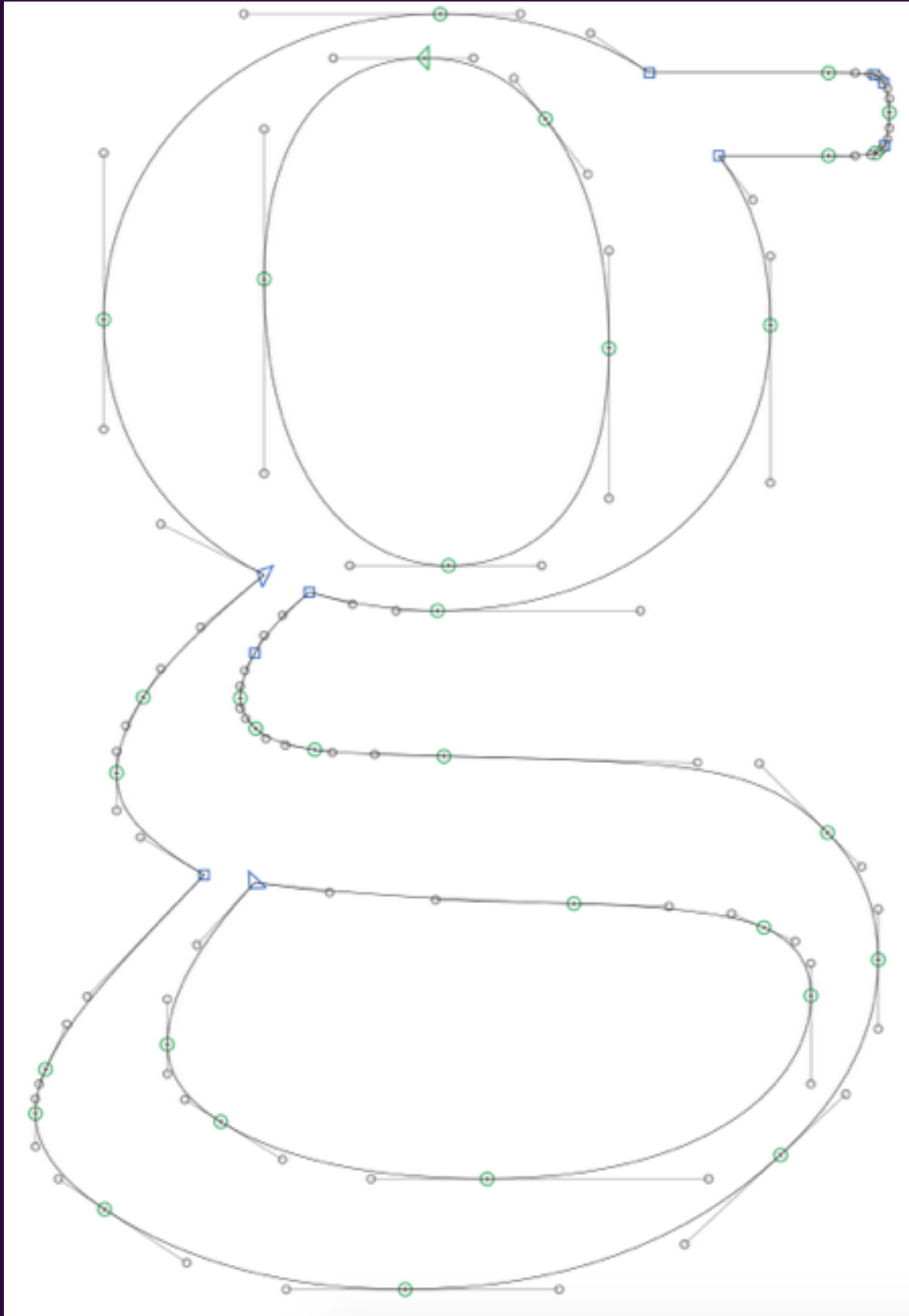


SDFs generation 8SSEDT



Gustavson 2011 and Danielsson 1980

SDFs generation ARM technique



ARM Technique

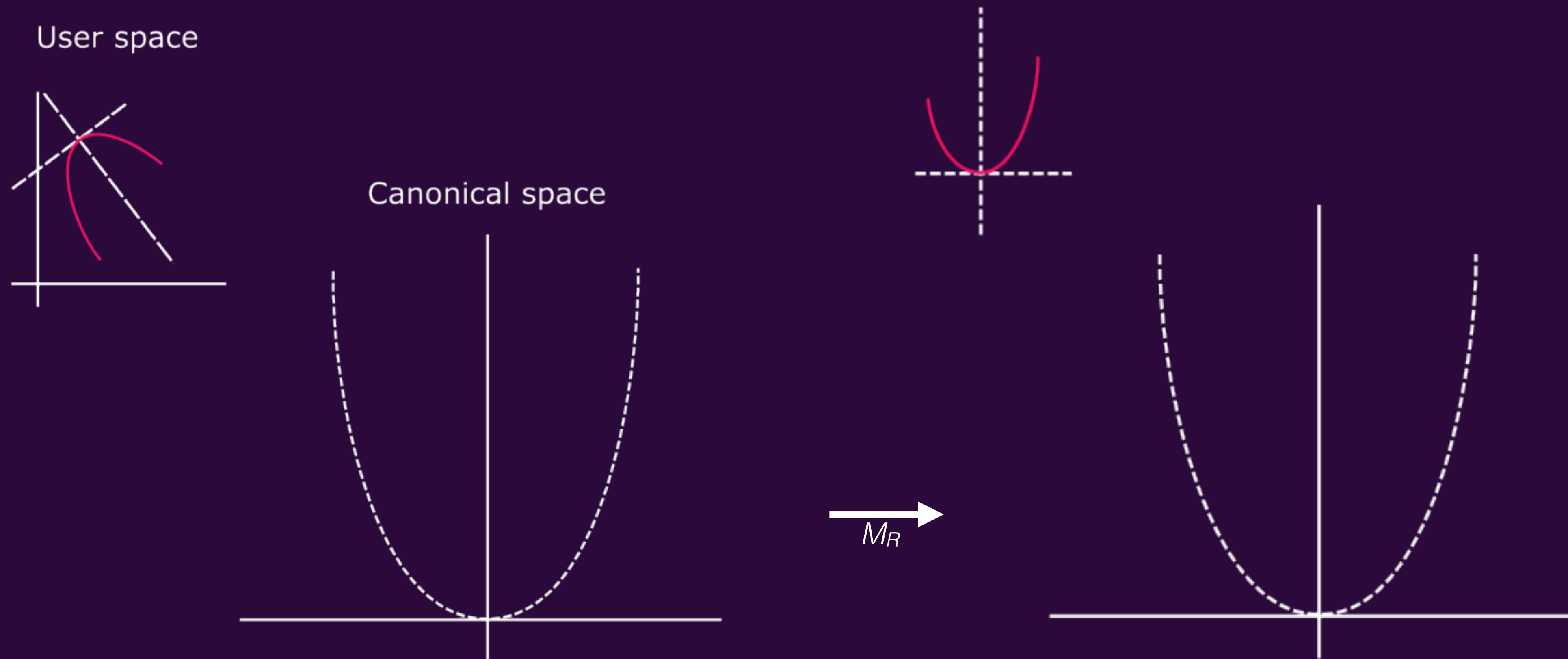
- Any Line or Quadratic Bézier curve can be transformed into a fixed, known form with a series of transformations (Translation, Rotation, Scaling)
- All lines can map to a simple horizontal line section of

$$y = 0$$

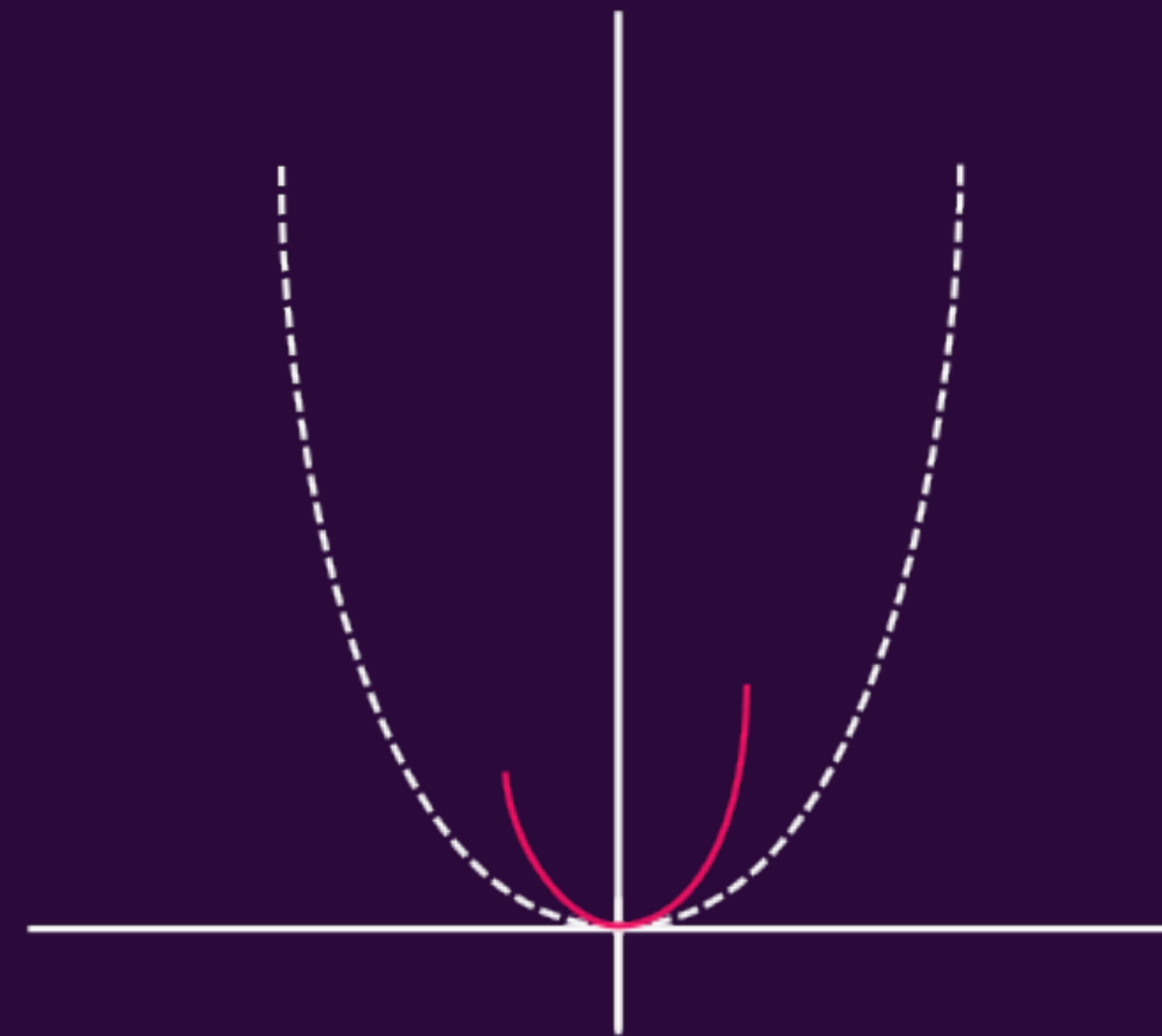
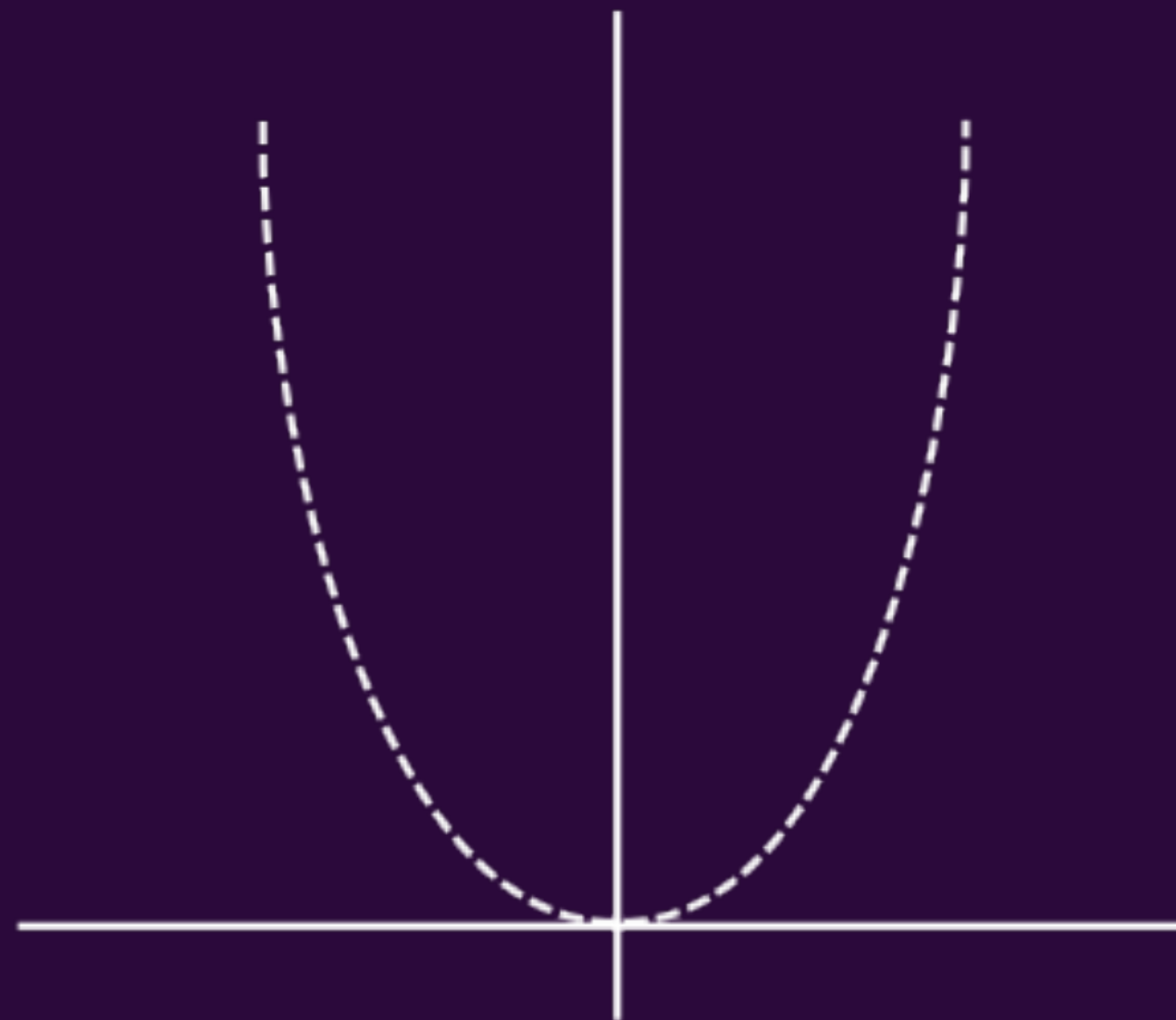
- All Quadratic Bézier curve can map onto a segment of the curve

$$y = x^2$$

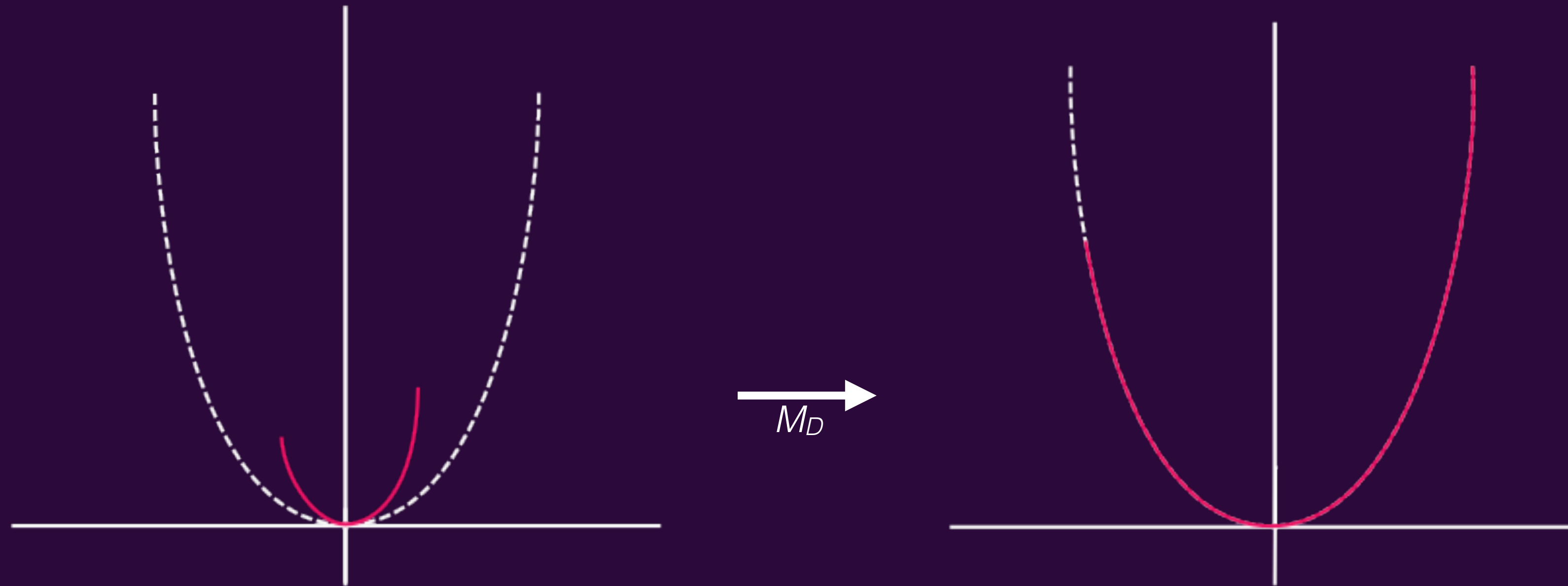
Fundamentals - Quadratic Beziars 1



Fundamentals - Quadratic Beziars 2



Fundamentals - Quadratic Beziars 3



Final Transformation Matrix

$$R = DTR_{\theta}$$

Distance calculation from (u, v)

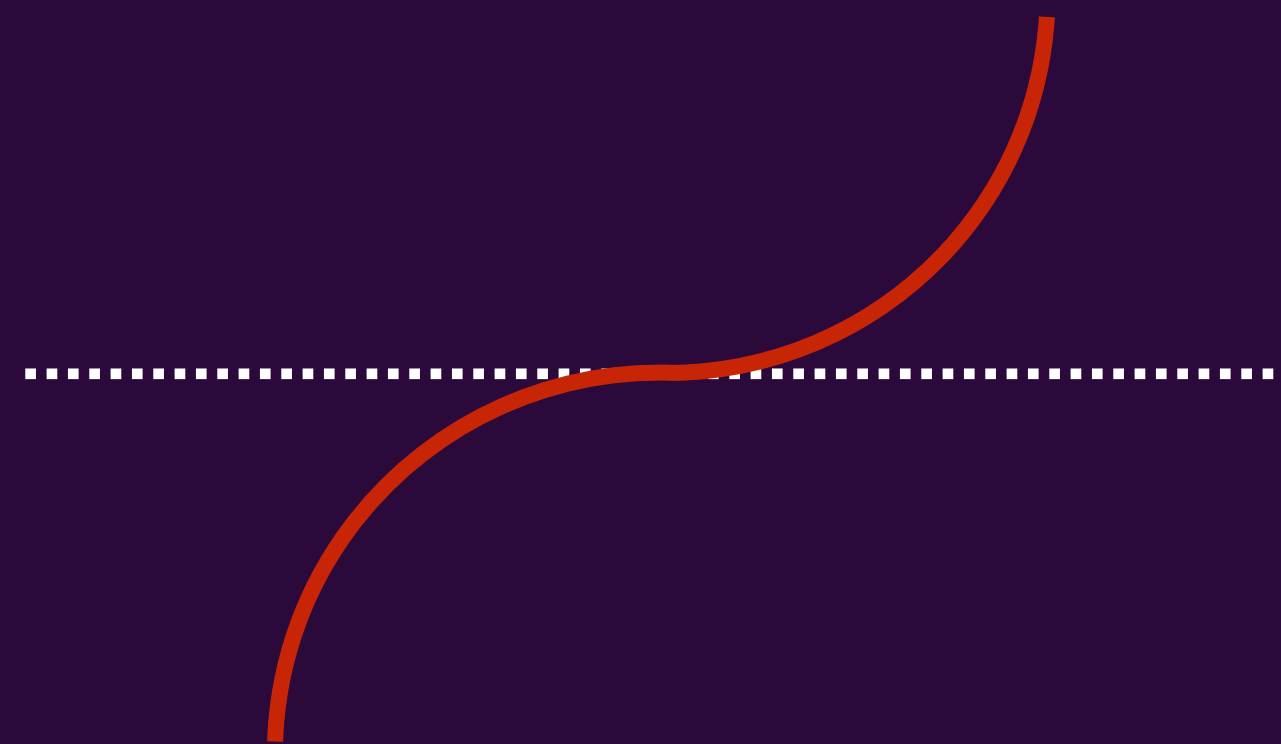
$$D = \left((x - u)^2 + (x^2 - v)^2 \right)^{1/2}$$

$$4x^3 + (1 - 2v)2x - 2u = 0$$

$$x^3 + ax + b = 0 \quad \text{where} \quad \begin{aligned} a &= \frac{1}{2} - v \\ b &= -\frac{u}{2} \end{aligned}$$

Roots case 1

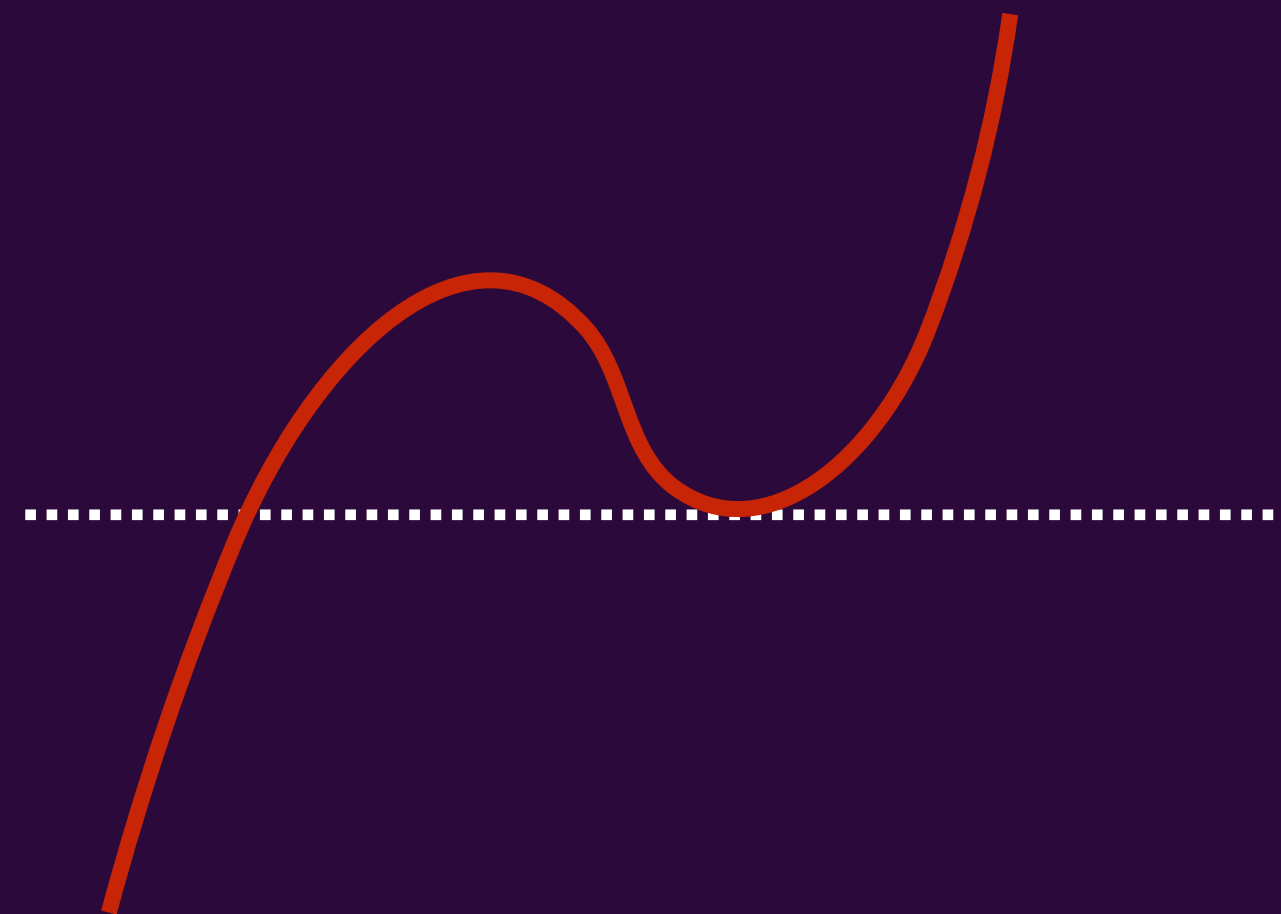
$$b^2/4 + a^3/27 > 0$$



$$x_1 = \left(-\frac{b}{2} + \sqrt{\frac{b^2}{4} + \frac{a^3}{27}} \right)^{1/3} + \left(-\frac{b}{2} - \sqrt{\frac{b^2}{4} + \frac{a^3}{27}} \right)^{1/3}$$

Roots case 2

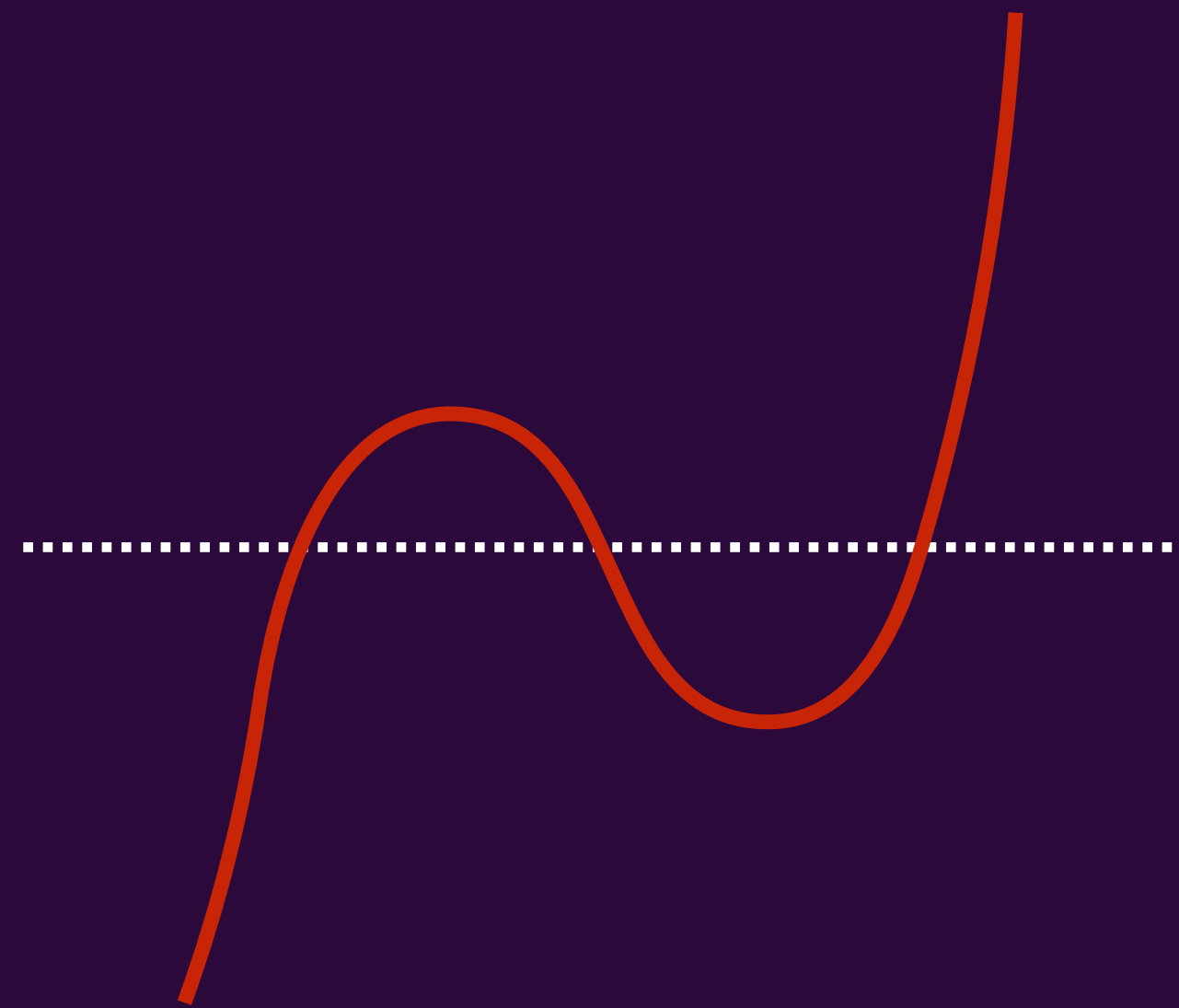
$$b^2/4 + a^3/27 = 0$$



$$x_1 = \left(-\frac{b}{2} + \sqrt{\frac{b^2}{4} + \frac{a^3}{27}} \right)^{1/3} + \left(-\frac{b}{2} - \sqrt{\frac{b^2}{4} + \frac{a^3}{27}} \right)^{1/3}$$

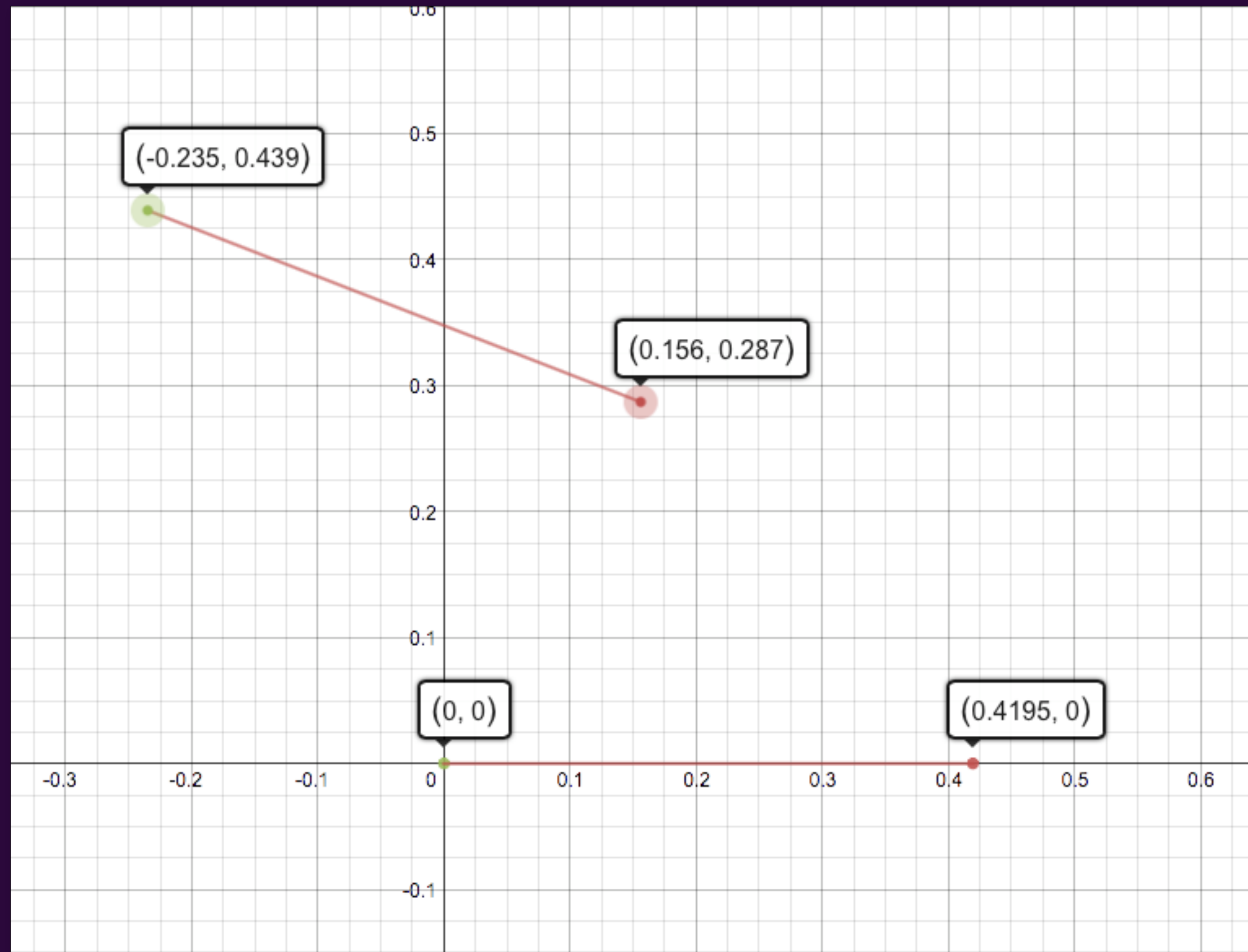
Roots case 3

$$b^2/4 + a^3/27 < 0$$



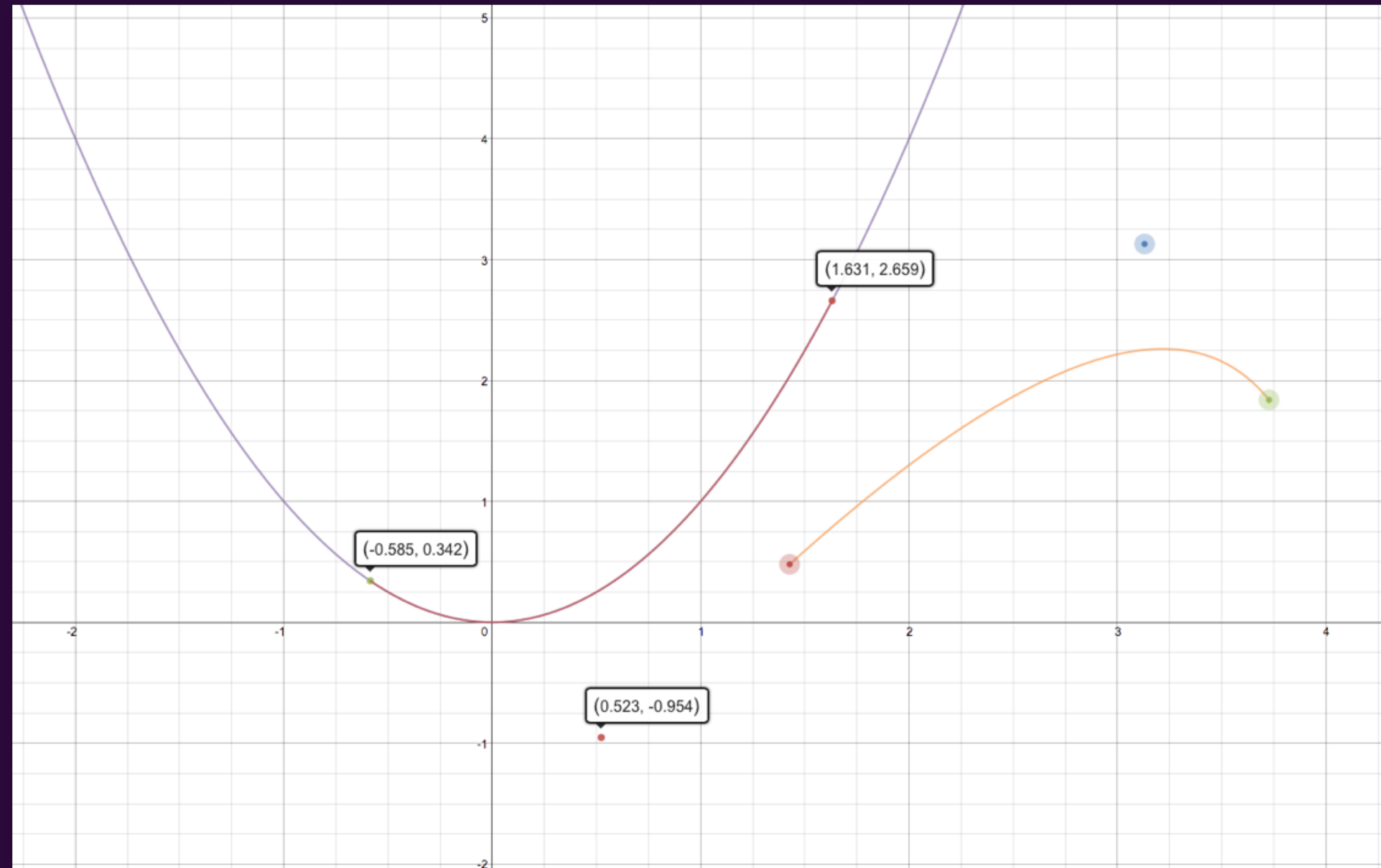
$$x_k = 2\sqrt{\frac{-a}{3}} \cos\left(\frac{\phi}{3} + \frac{2k\pi}{3}\right) \quad k = 0, 1, 2 \quad \cos \phi = \begin{cases} -\sqrt{\frac{b^2/4}{-a^3/27}} & \text{if } b > 0 \\ \sqrt{\frac{b^2/4}{-a^3/27}} & \text{if } b < 0 \end{cases}$$

Live Demo: Lines



<https://www.desmos.com/calculator/lvww689bx0>

Live Demo: Curves

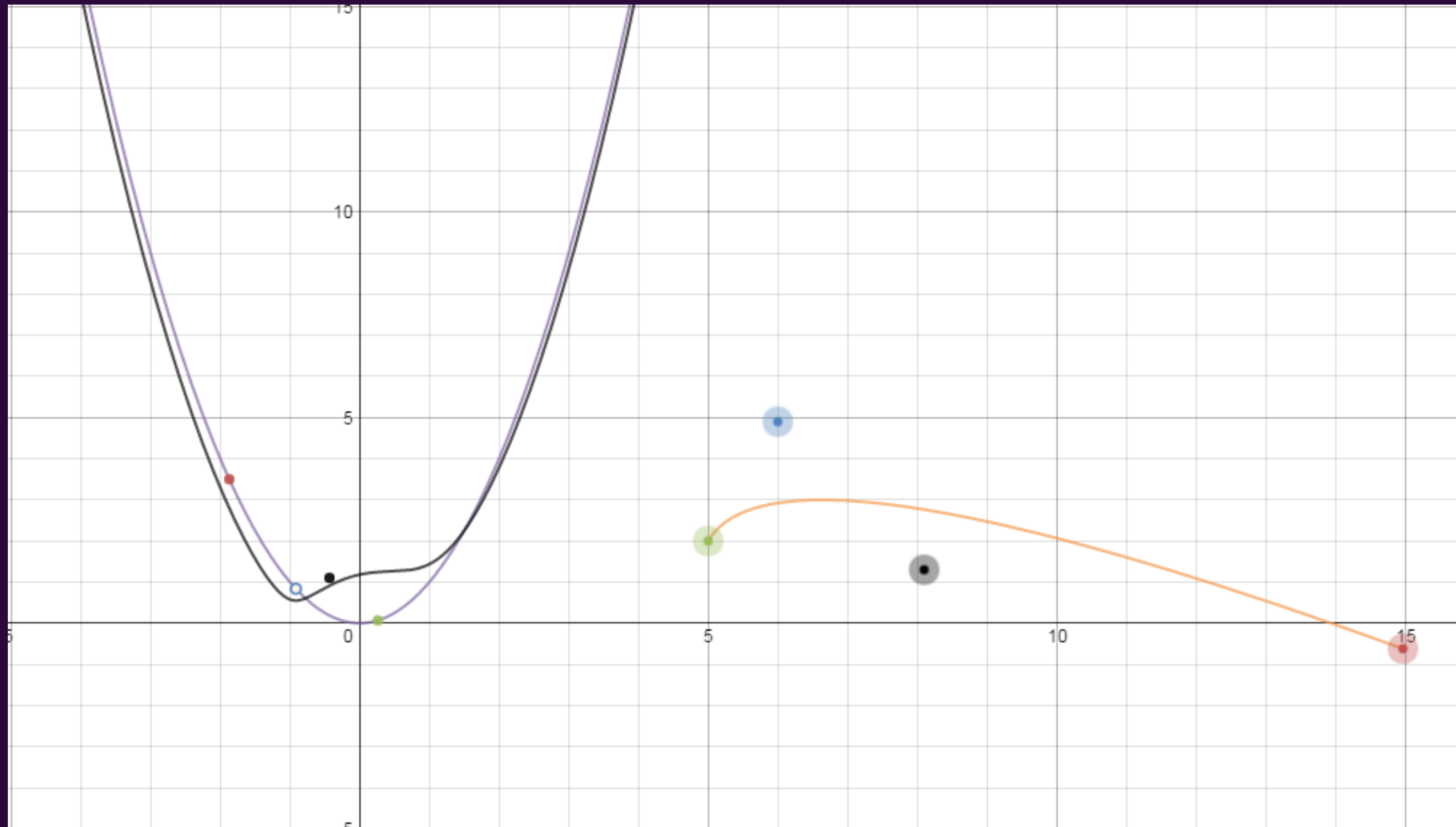


<https://www.desmos.com/calculator/l9ssbp9xqh>

Implementation Theory

- For any given segment
 - We can take an arbitrary point in space around it
 - Using the transformed point, calculate distance to canonical form
 - Scale back to give real distance
 - We can tell if the point is inside or outside the segment
 - Lines: sign of y
 - Quadratic Bézier's side of tangent from nearest point on curve
- Sample around a segment in a pixel grid and we have signed distance values!

Live Demo! complete

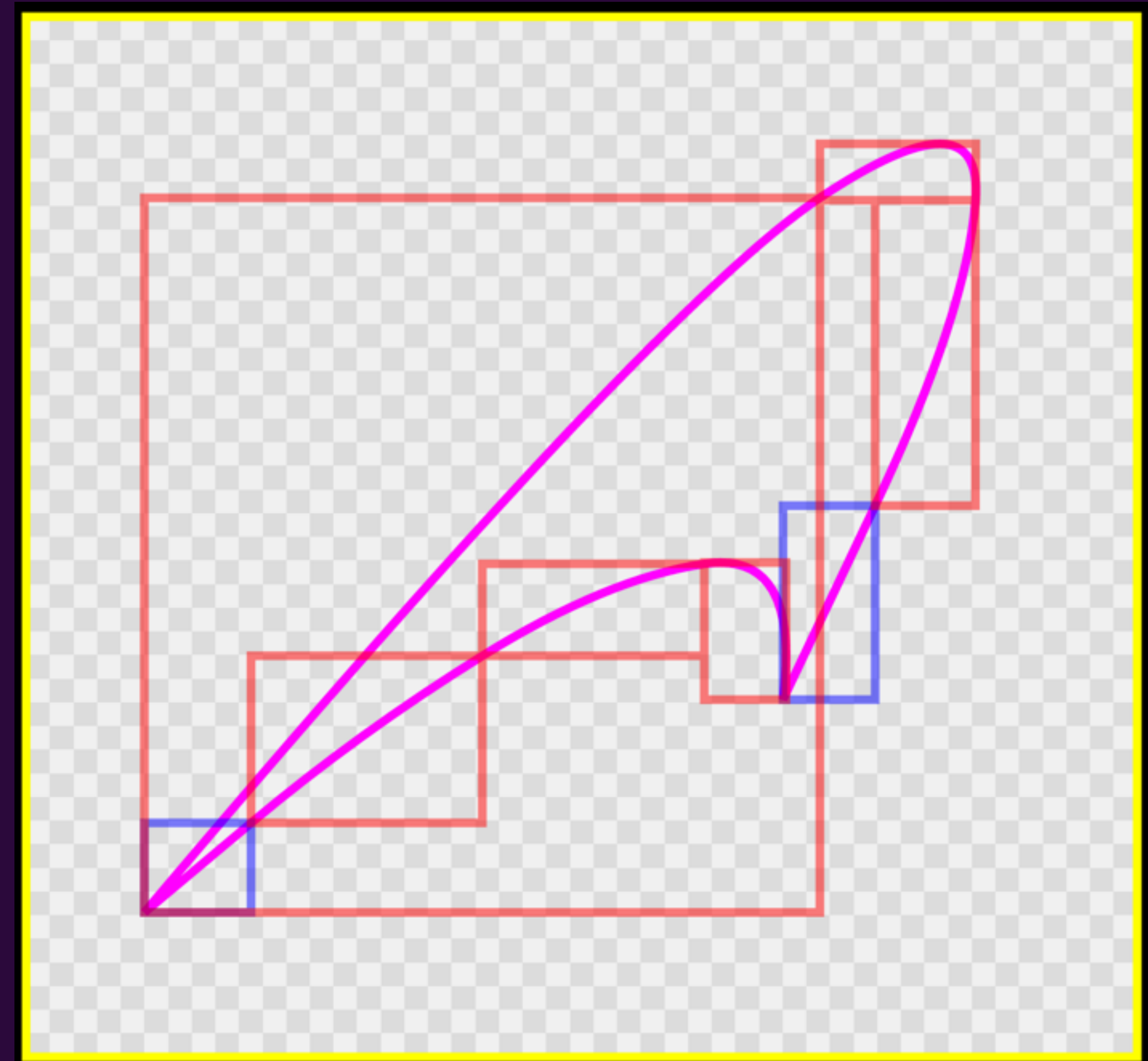
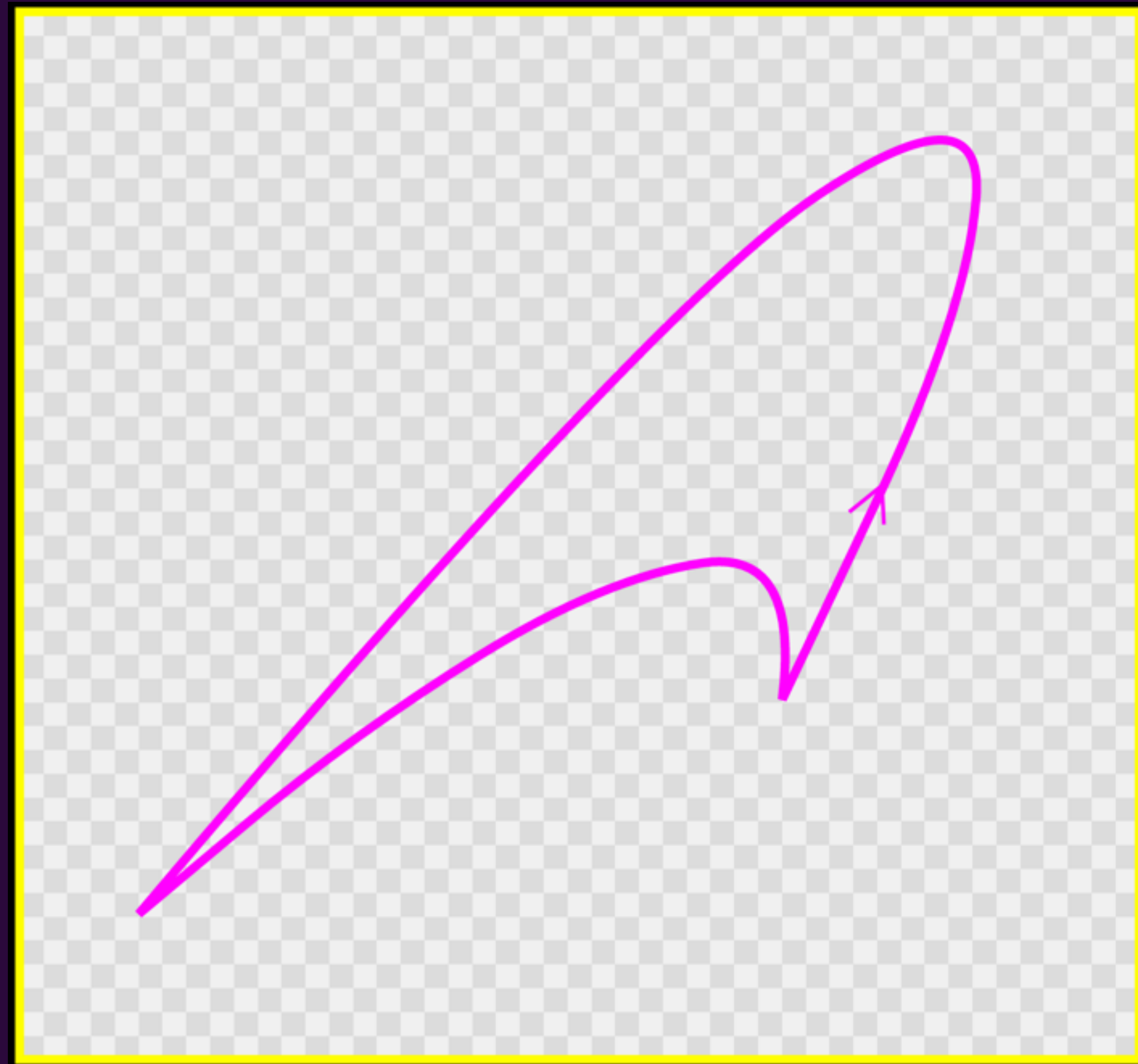


<https://www.desmos.com/calculator/wxqfhychxu>

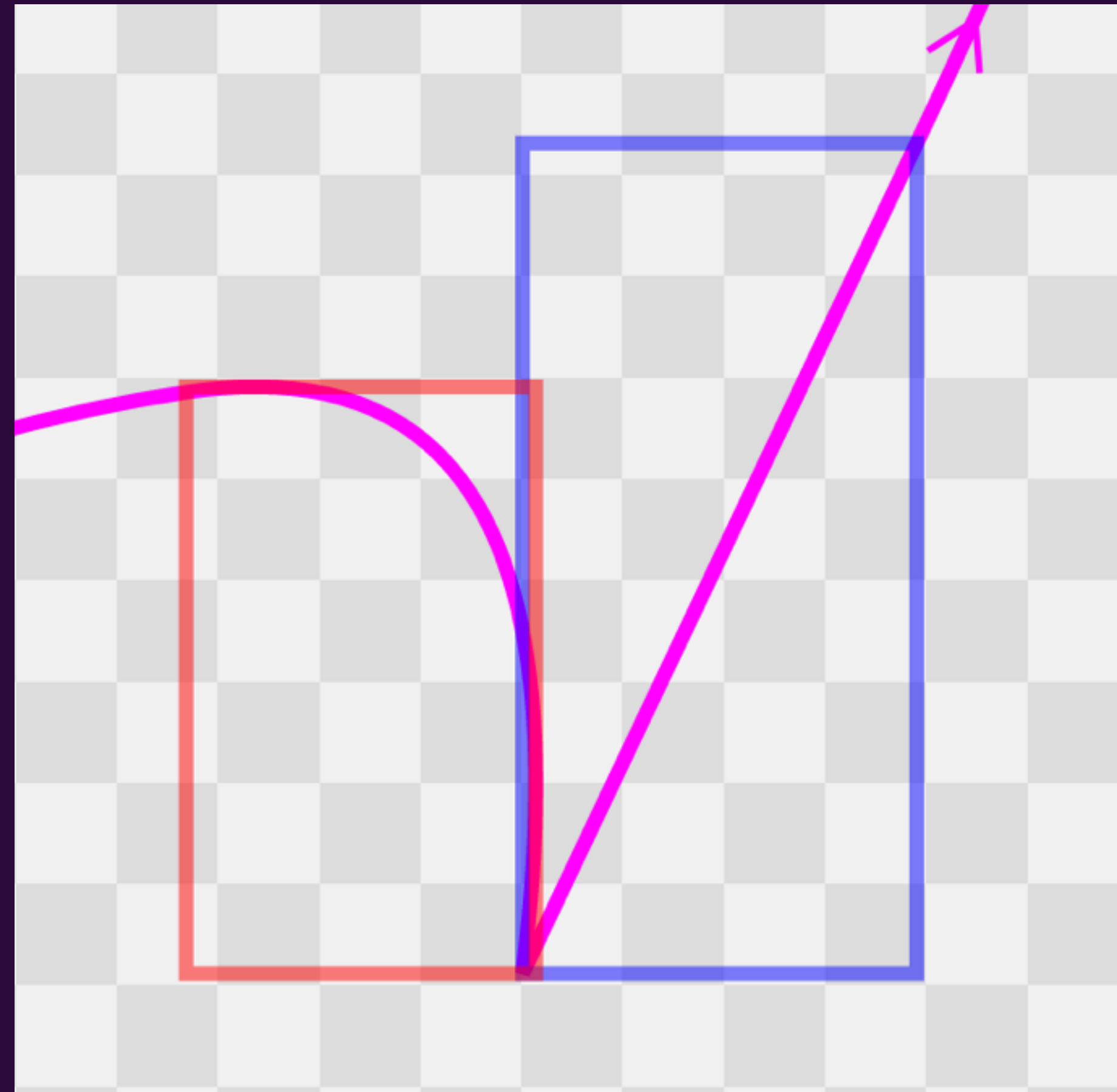
Implementation – Signed Distance Fields

```
// for each segment initialise object and calculate transformation matrices
// calculate bounding box this becomes the sampling grid around segment
// for each sampling point in its bounding box
for each segment
    for each bounds_row
        for each bounds_column
            calculate_distance_and_winding_score();
// for each sampling point in its bounding box
for each row
    for each column
        resolve_to_distance_map();
```

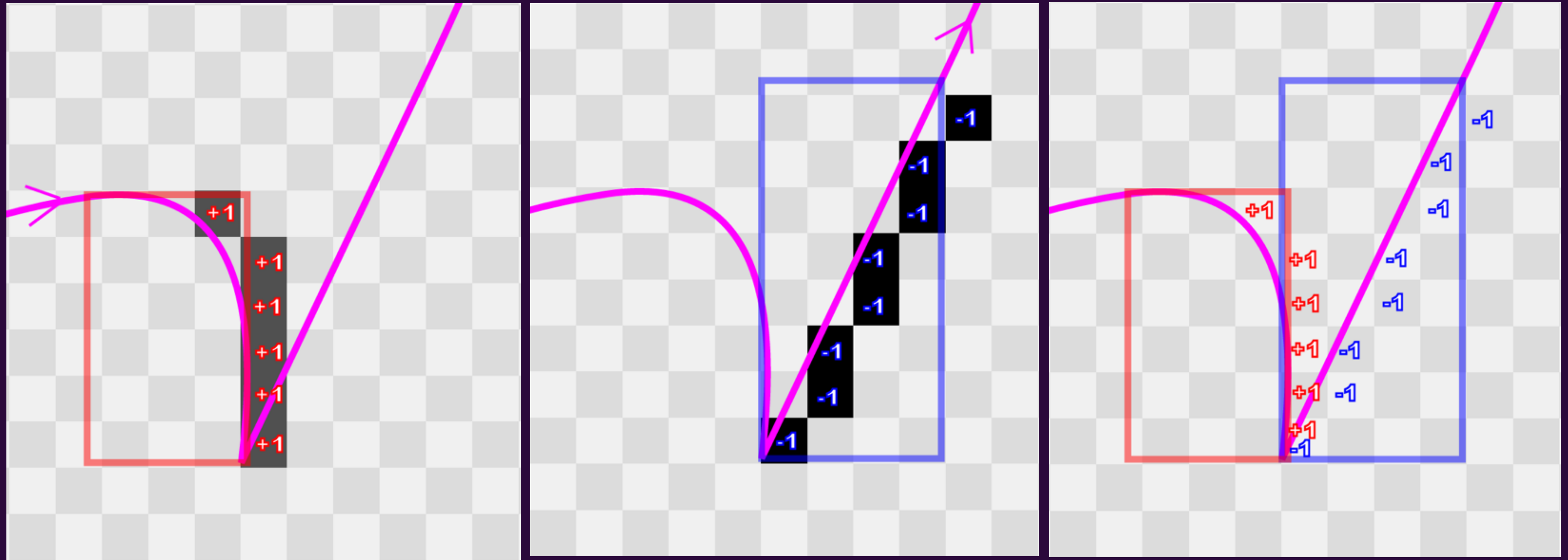
Winding Number



Zooming in on a Section

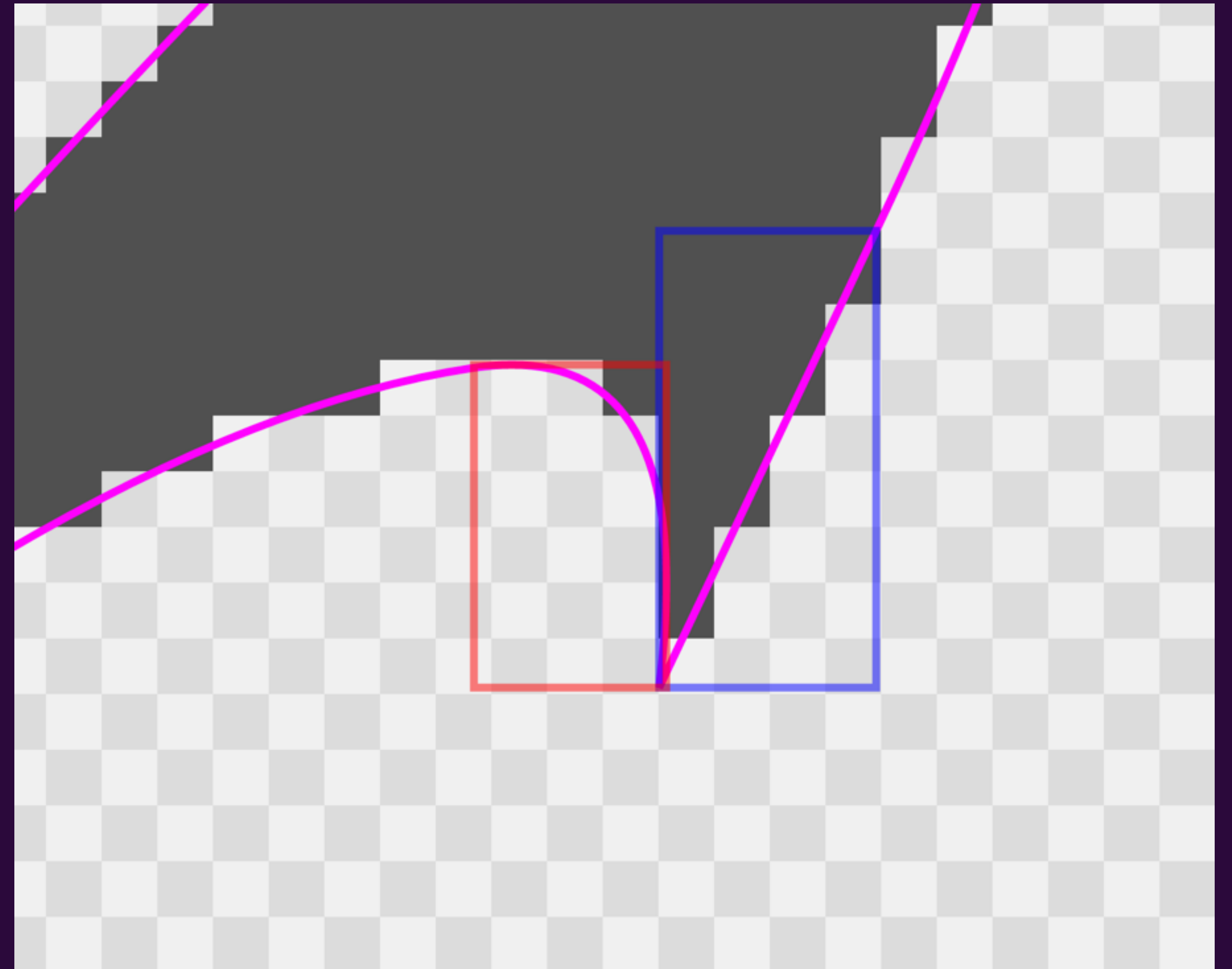
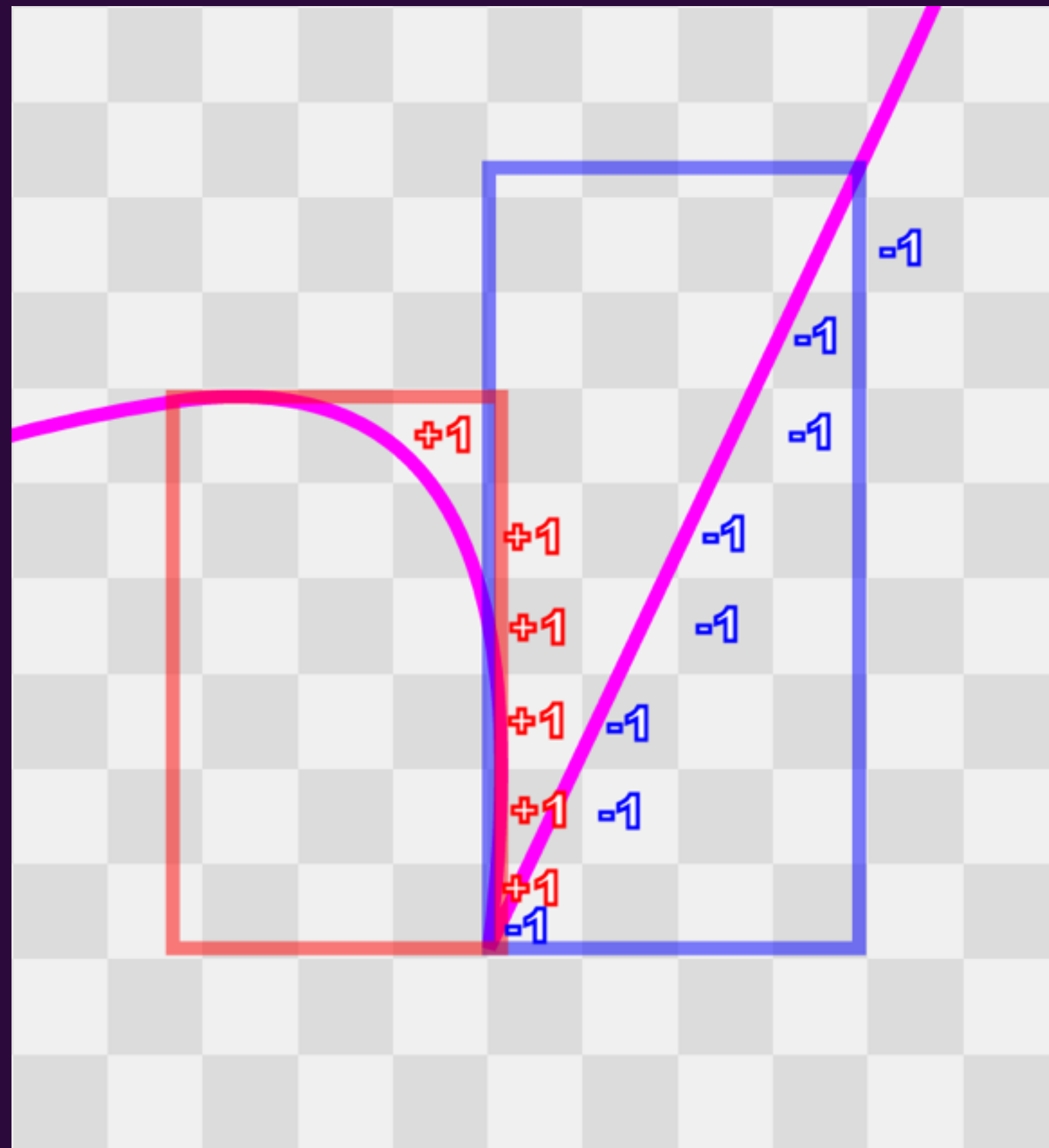


Calculate Delta Winding Score



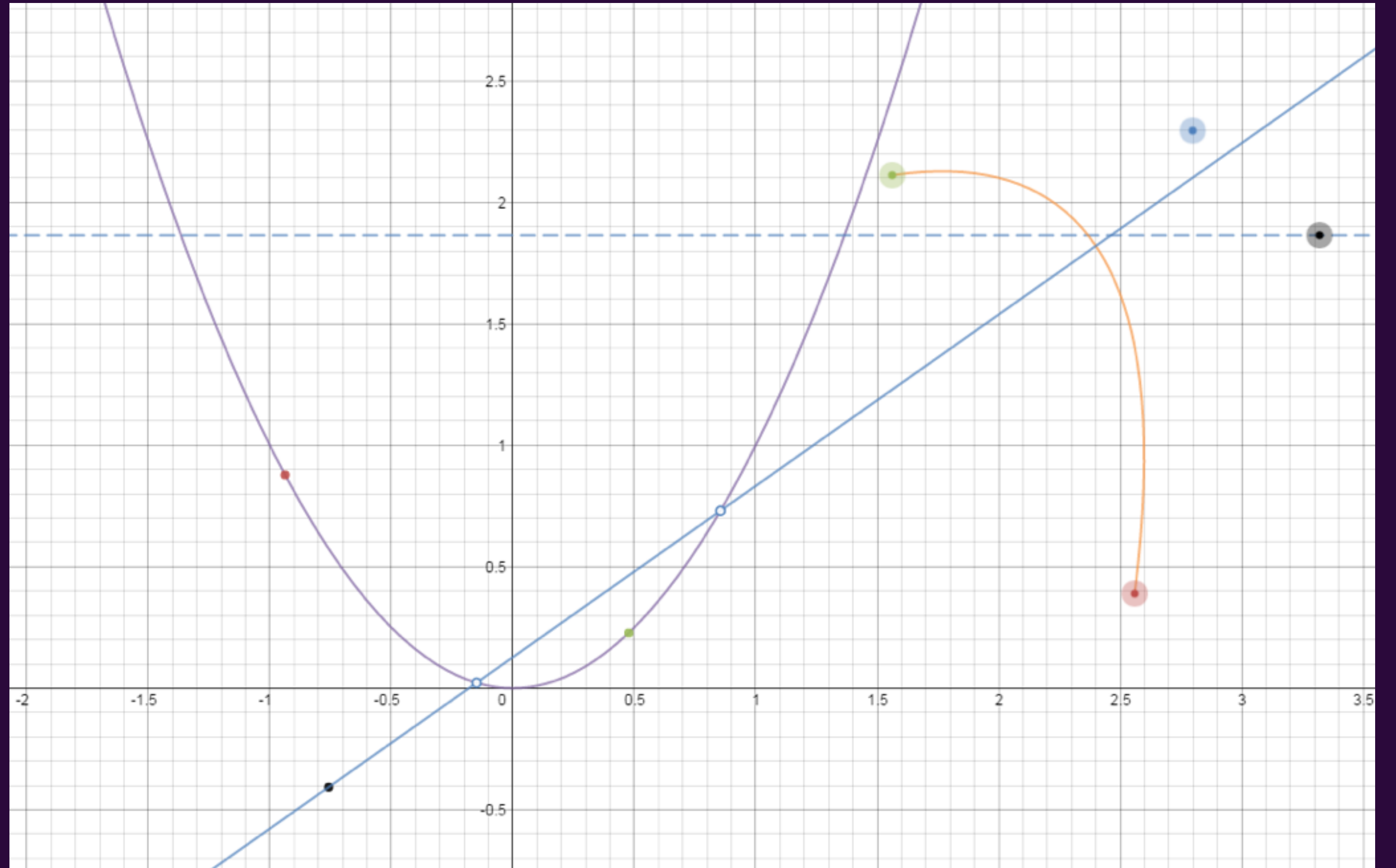
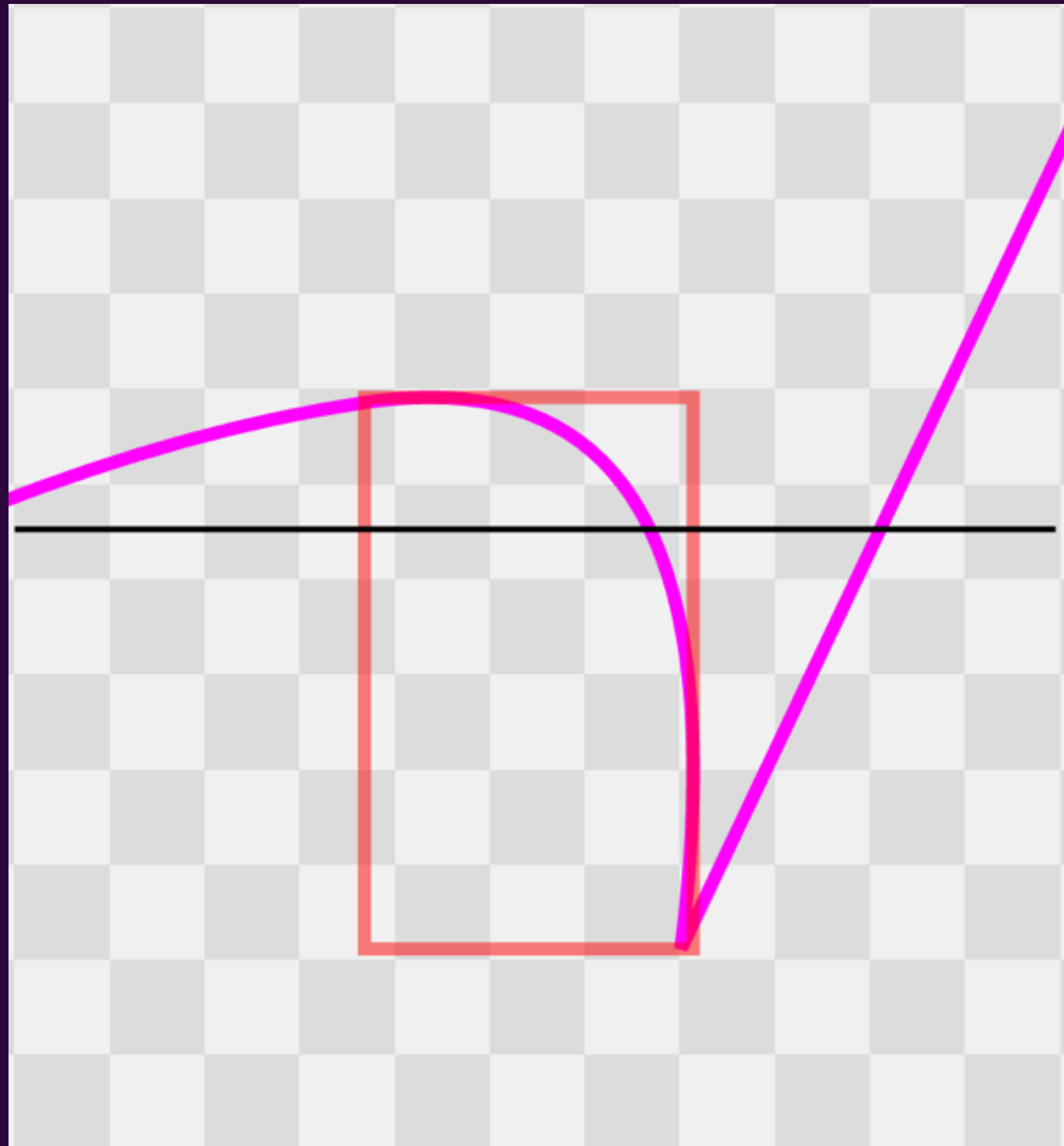
Right to Left **+1**, Left to Right **-1**

Winding number



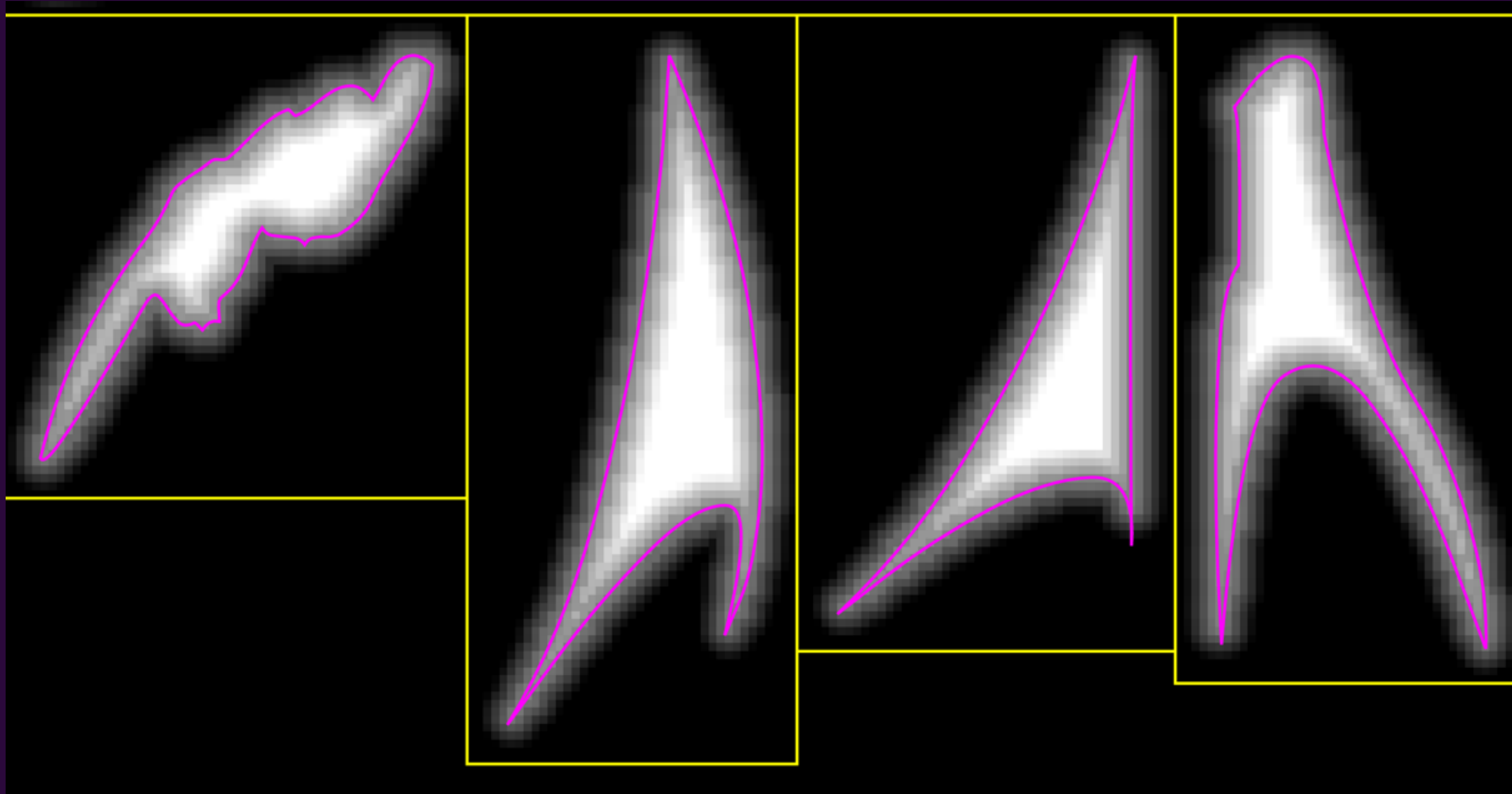
Summary the delta winding score from left to right

Side of quadratic curve



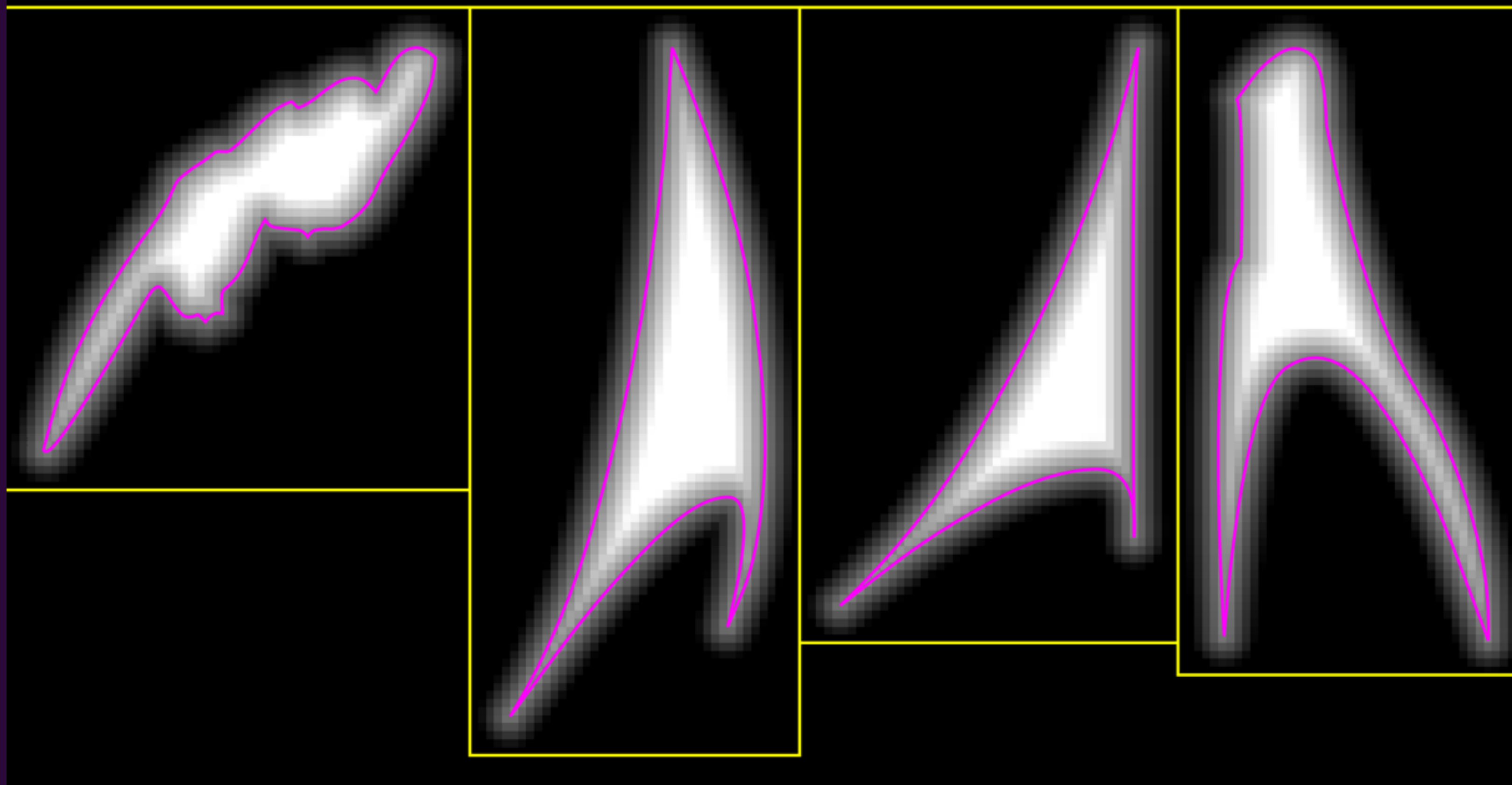
<https://www.desmos.com/calculator/wuugaixiwh>

Quality

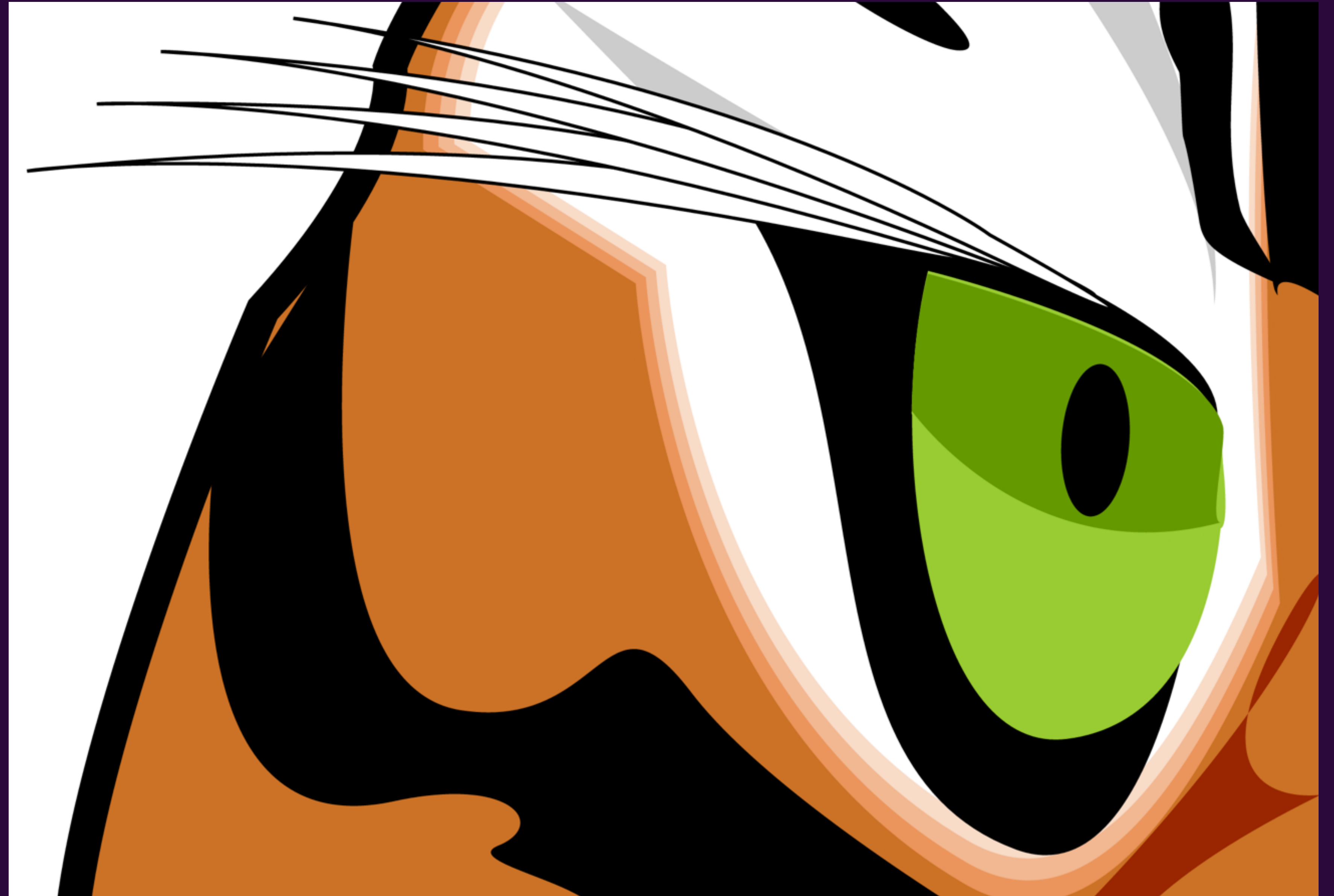


Skia

Quality



ARM

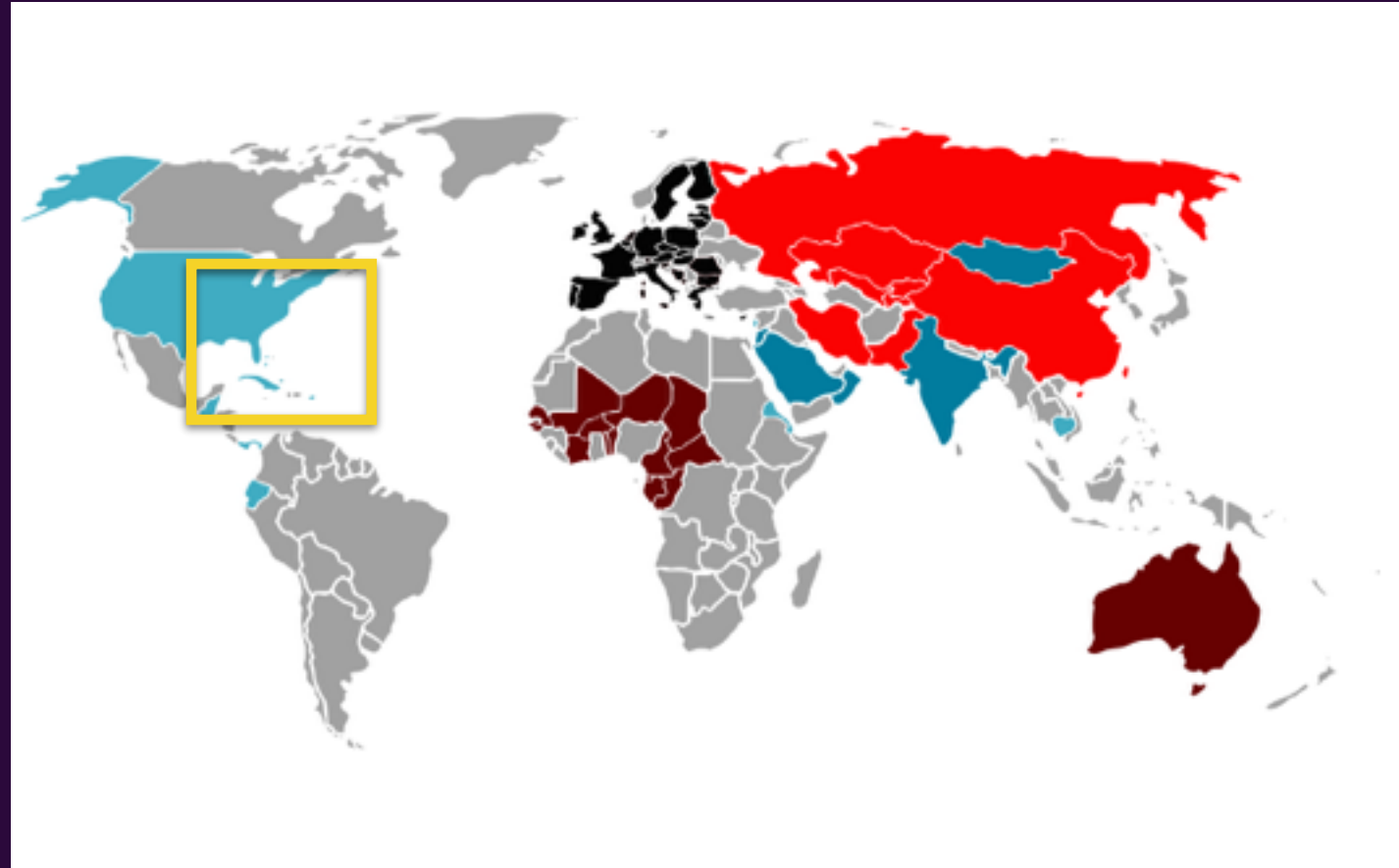


Skia, Reference

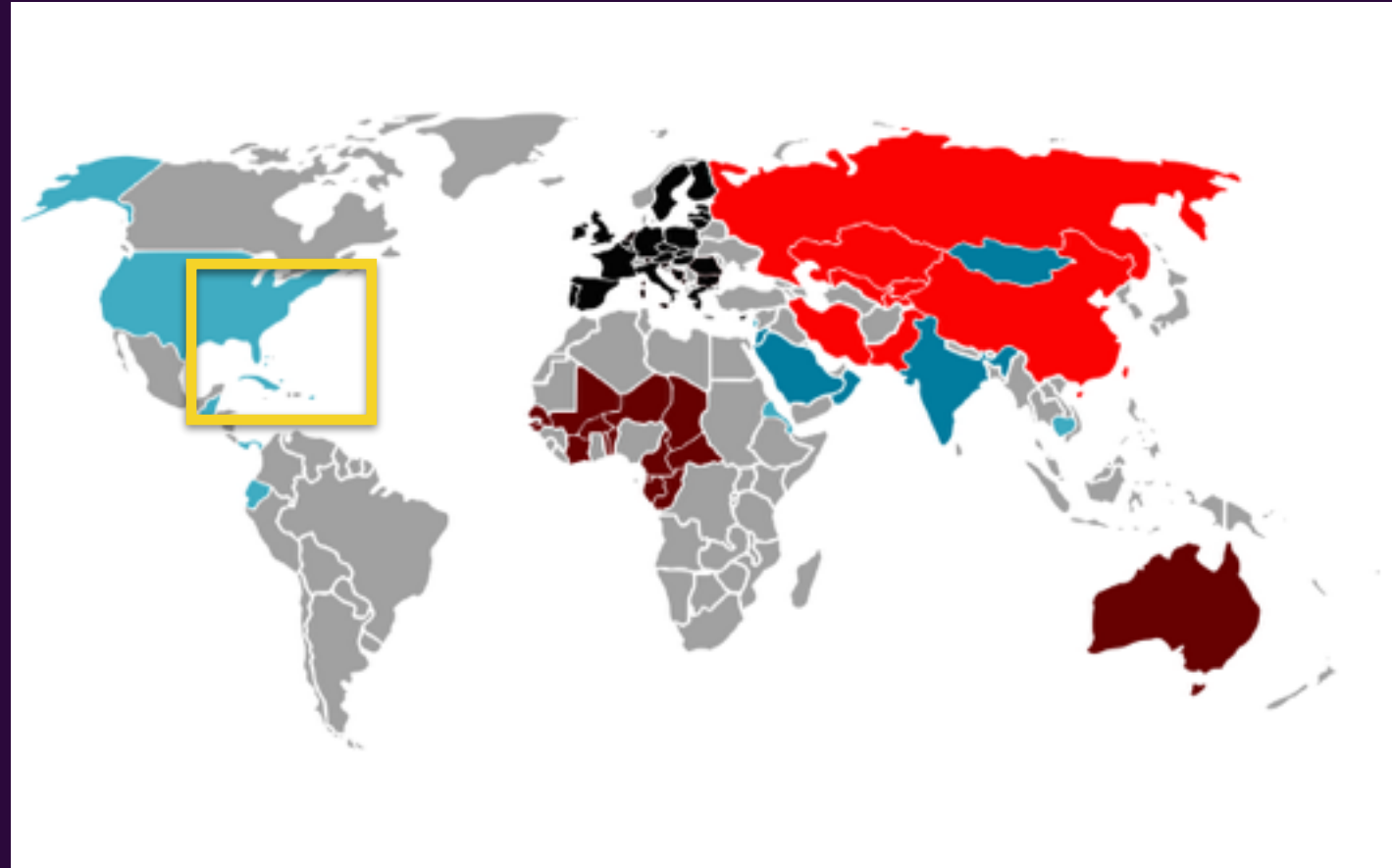




ARM



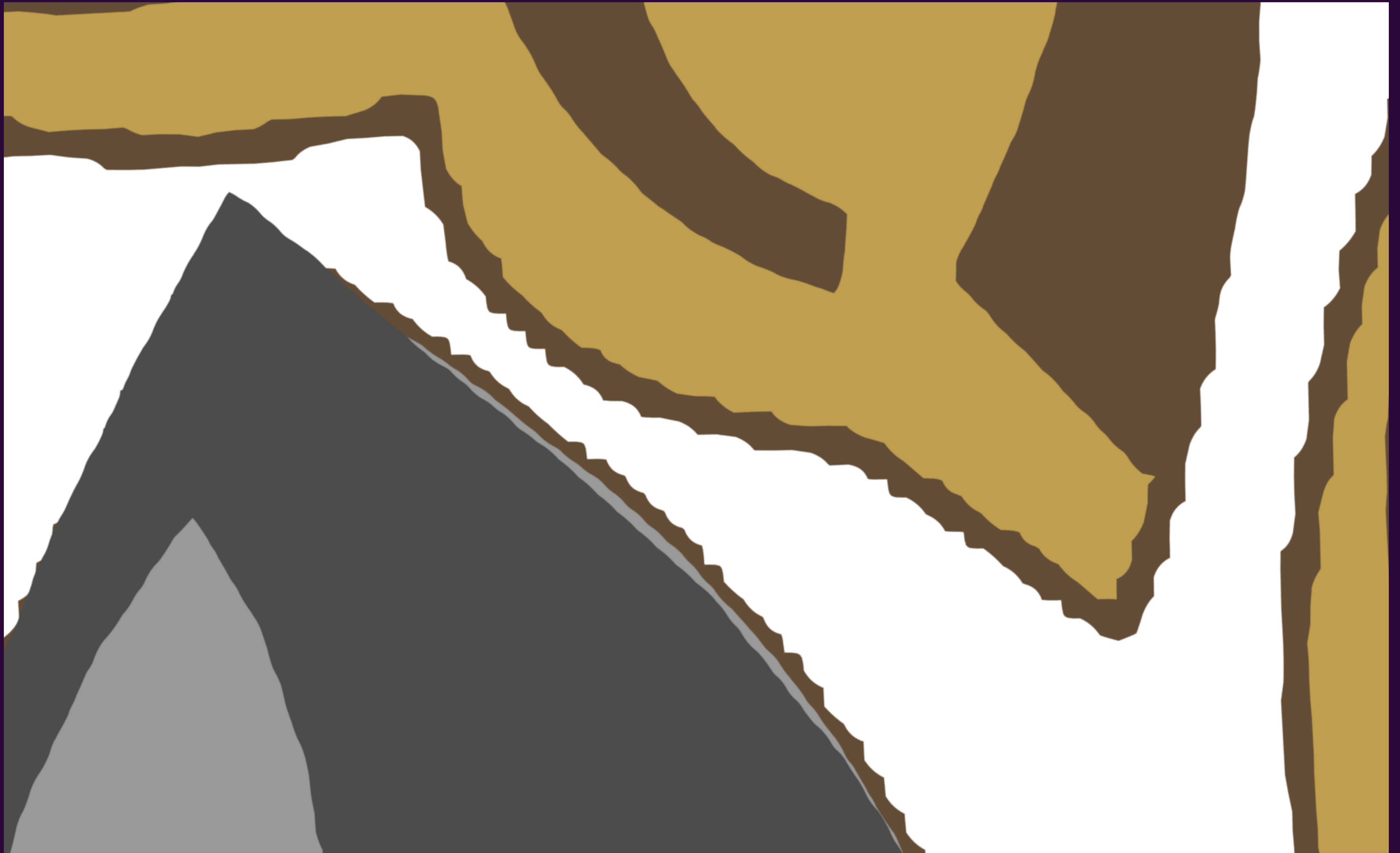
Skia



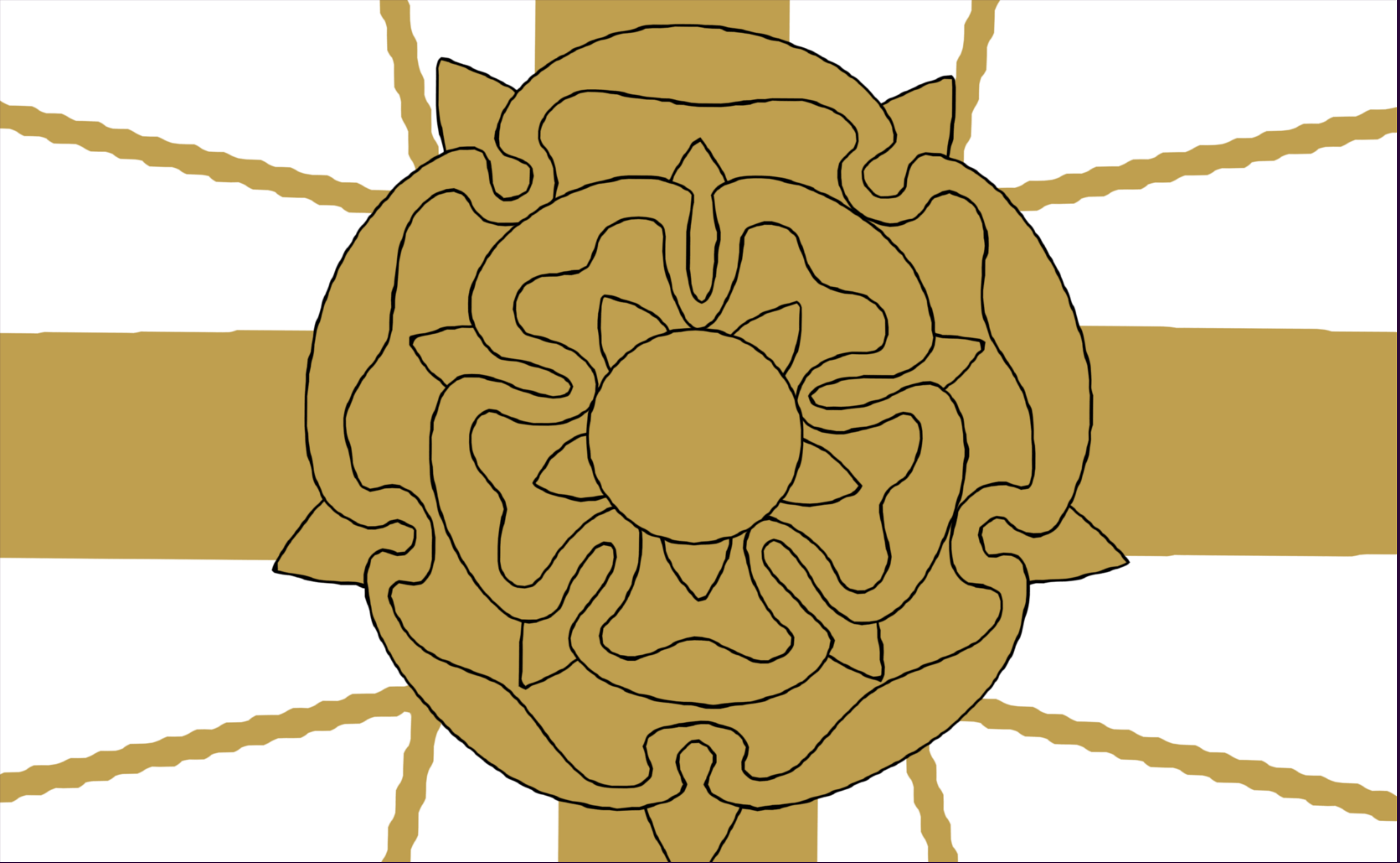
ARM

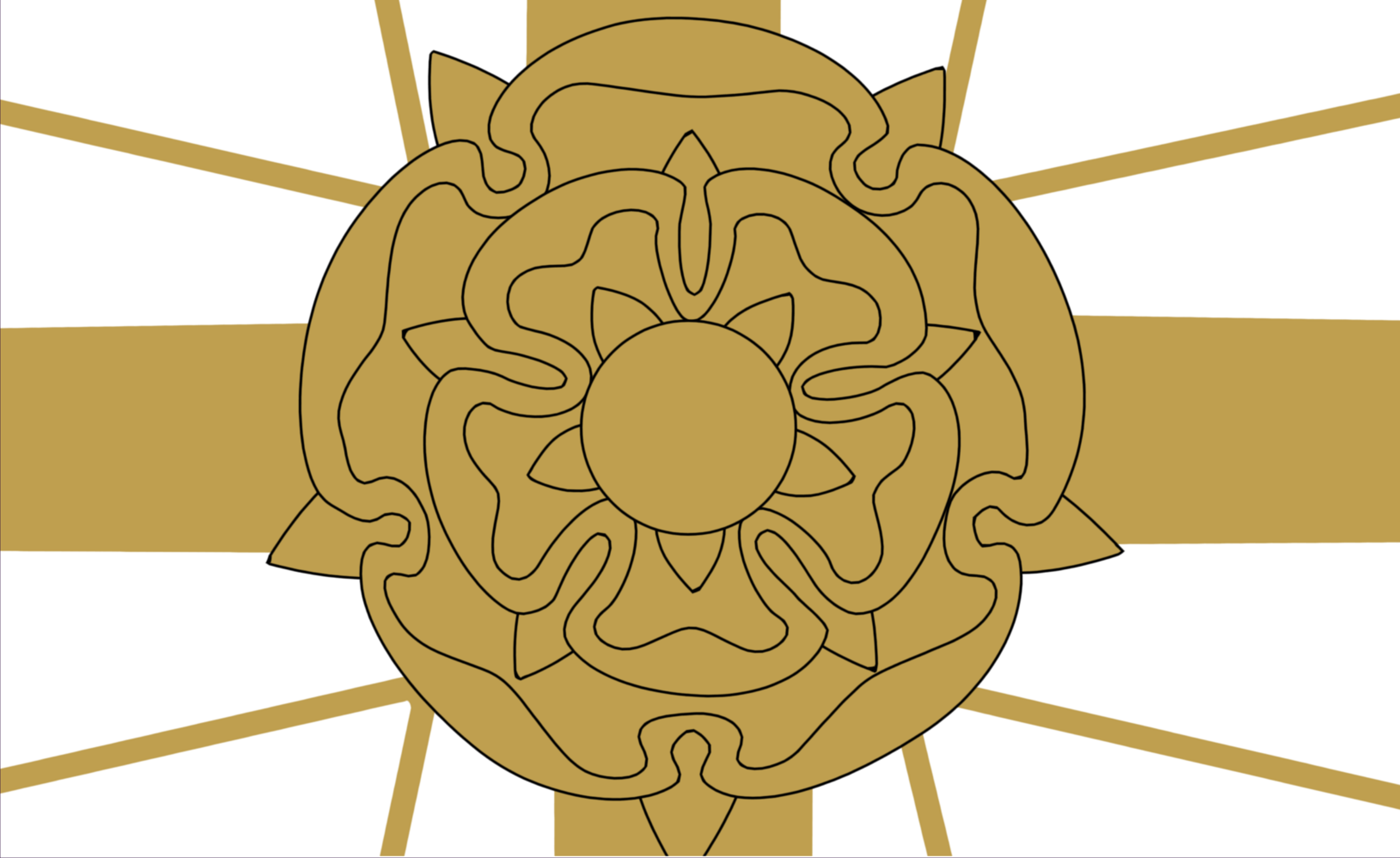








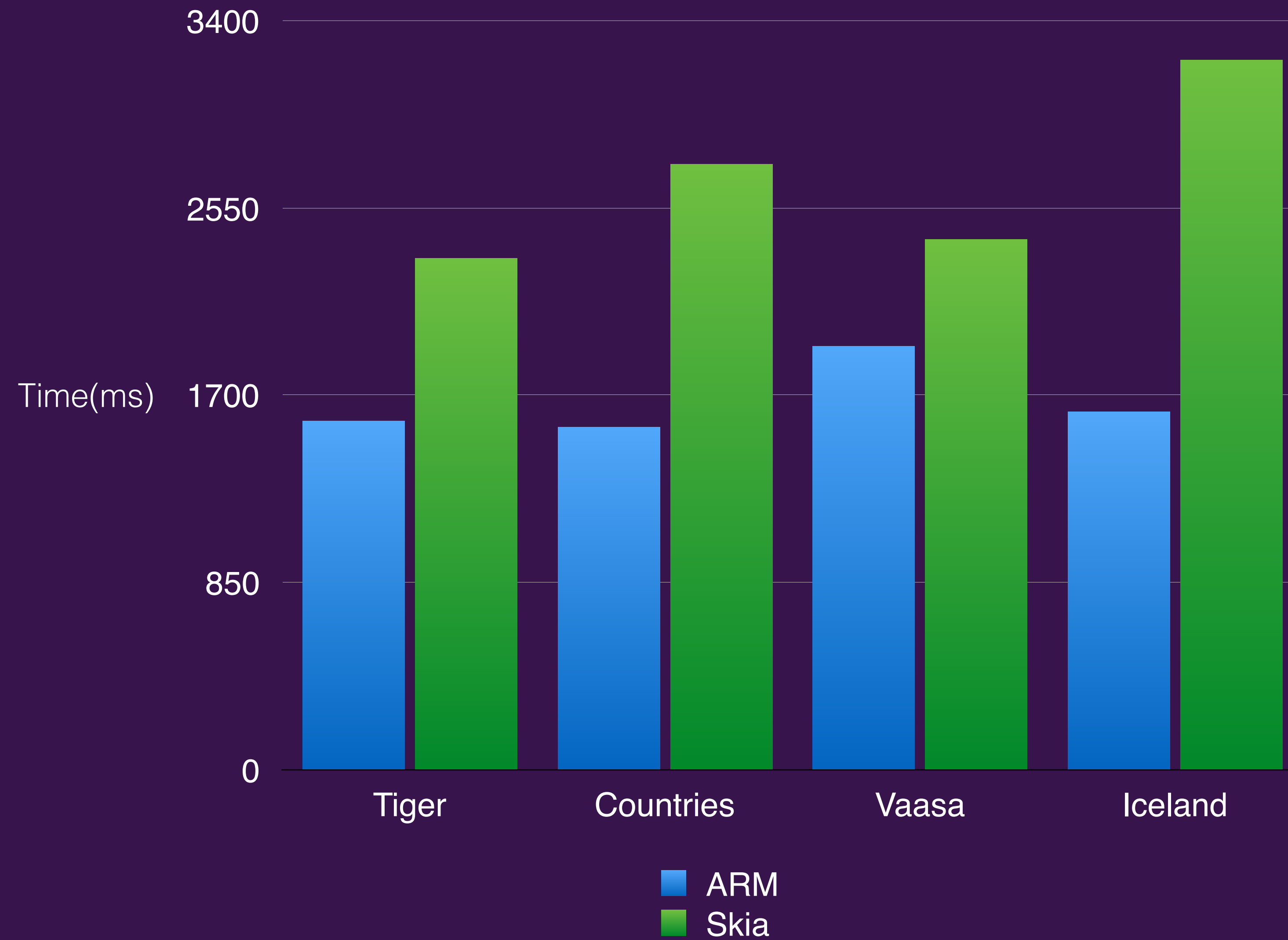








Results in Numbers



Where is the Code?

Chromium Code Reviews [Help](#) | [Chromium Project](#) | [Sign in](#) (39)

Issues Search
[Open Issues](#) | [Closed Issues](#) | [All Issues](#) | [Sign in](#) with your [Google Account](#) to create issues and add comments

Side by Side Diff: src/gpu/GrDistanceFieldGenFromVector.cpp
Issue 1643143002: Generate Signed Distance Field directly from vector path Base URL: <https://skia.googlesource.com/skia.git@master>
Patch Set: Transform tolerance to fix winding number assertion issue. Created 3 months, 4 weeks ago
Use n/p to move between diff chunks; N/P to move between comments. Please [Sign in](#) to add in-line comments.

Context: 10 lines Column Width: 80 Tab Spaces: 8 [Adjust View](#)
Jump to: [src/gpu/GrDistanceFieldGenFromVector.cpp](#)
[View unified diff](#) | [Download patch](#)

« no previous file with comments | [src/gpu/GrDistanceFieldGenFromVector.h](#) ('k') | [src/gpu/batches/GrAADistanceFieldPathRenderer.cpp](#) ('f') | no next file with comments »
[Toggle Intra-line Diffs](#) ('T') | [Expand Comments](#) ('e') | [Collapse Comments](#) ('c') | [Hide Comments](#) ('s')

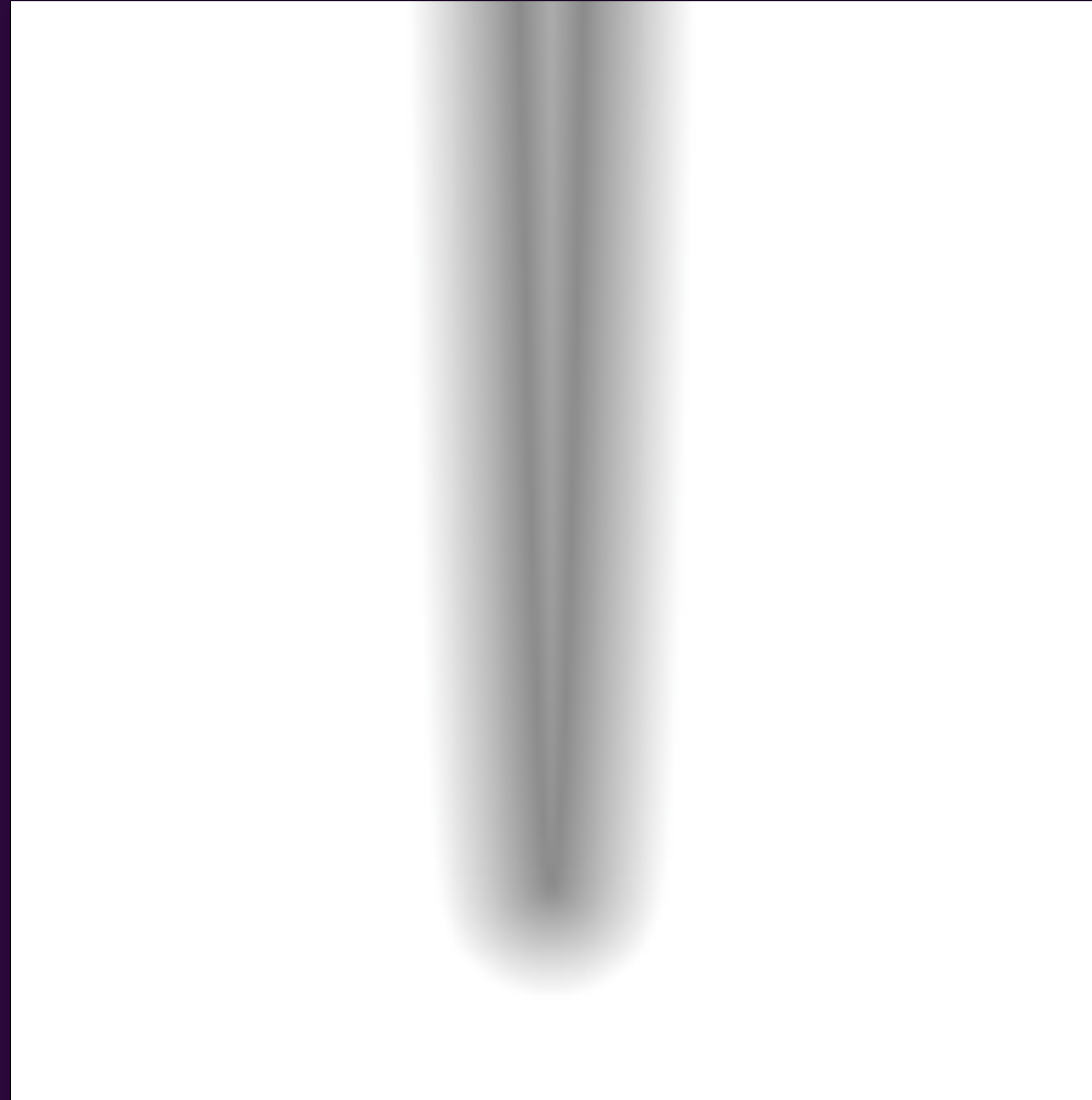
OLD	NEW
(Empty)	<pre>1 /* 2 * Copyright 2016 ARM Ltd. 3 * 4 * Use of this source code is governed by a BSD-style license that can be 5 * found in the LICENSE file. 6 */ 7 8 #include "GrDistanceFieldGenFromVector.h" 9 #include "SkPoint.h" 10 #include "SkGeometry.h" 11 #include "SkPathOps.h" 12 #include "GrPathUtils.h" 13 #include "GrConfig.h" 14 15 /** 16 * If a scanline (a row of texel) cross from the kRight_SegSide 17 * of a segment to the kLeft_SegSide, the winding score should 18 * add 1. 19 * And winding score should subtract 1 if the scanline cross 20 * from kLeft_SegSide to kRight_SegSide. 21 * Always return kNA_SegSide if the scanline does not cross over 22 * the segment. Winding score should be zero in this case. 23 * You can get the winding number for each texel of the scanline 24 * by adding the winding score from left to right. 25 * Assuming we always start from outside, so the winding number 26 * should always start from zero. 27 * 28 * 29 * ...R L.....L R.....L R.....R L..... <= Scanline & side of segment 30 * +1 -1 -1 +1 <= Winding score 31 * 0 1 0 -1 0 <= Winding number 32 */</pre>

<https://codereview.chromium.org/1643143002>

Summary

- Much better quality
- Higher performance. In its current state we are above 75% faster.
- Highly parallel algorithm (currently single threaded but can be easily multi-threaded)
- Lots of scope for improvements and optimisations (still in beta)
- Only CPU version at the moment but the whole algorithm is based around GPU architecture so it will be much faster with GPU
- Not limited to Font glyphs but any path can be converted to SDF

GPU version



<https://www.desmos.com/calculator/lqn6g1tpty>

Acknowledgements

- Chris Doran chris.doran@arm.com
- Roberto Lopez Mendez roberto.lopezmendez@arm.com
- Rich Evans rcb.evans@outlook.com
- Joel Liang joel.liang@arm.com

Bonus content

Standard form theory

A general second-degree curve is defined by an equation of the form

$$X^t C X = (x, y, 1) \begin{pmatrix} a & h & g \\ h & b & f \\ g & f & c \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

This is equivalent to the component equation

$$ax^2 + 2hxy + by^2 + 2gx + 2fy + c = 0$$

In order to define a parabola (a second-degree Bezier) the coefficients must also satisfy

$$ab - h^2 = 0$$

Computing the standard form

We assume the input consists of three points in 2D, b_0 , b_1 and b_2 where b_1 is control point.
Find an equation which takes 2D vector and returns a 6D vector

$$F(x, y) = (x^2, xy, y^2, x, 1, y)$$

Now we turn to the Bezier control points (b_0, b_1, b_2) and from these define 3 new points along the curve:

$$c_1 = \frac{1}{16}(9b_0 + 6b_1 + b_2)$$

$$c_2 = \frac{1}{4}(b_0 + 2b_1 + b_2)$$

$$c_3 = \frac{1}{16}(b_0 + 6b_1 + 9b_2)$$

Computing the standard form

We now define the 5×6 matrix A

$$A = \begin{pmatrix} F(b_0) \\ F(c_1) \\ F(c_2) \\ F(c_3) \\ F(b_2) \end{pmatrix}$$

$$B[i] = \det A_i$$

$$[a, h, b, g, c, f] = [B_0, -1/2B_1, B_2, -1/2B_3, B_4, -1/2B_5]$$

Standard form Equation Accelerated version

$$a = (y_0 - 2y_1 + y_2)^2$$

$$b = (x_0 - 2x_1 + x_2)^2$$

$$c = x_0^2 y_2^2 - 4x_0 x_1 y_1 y_2 - 2x_0 x_2 y_0 y_2 + 4x_0 x_2 y_1^2 + 4x_1^2 y_0 y_2 - 4x_1 x_2 y_0 y_1 + x_2^2 y_0^2$$

$$h = -(y_0 - 2y_1 + y_2)(x_0 - 2x_1 + x_2)$$

$$g = x_0 y_0 y_2 - 2x_0 y_1^2 + 2x_0 y_1 y_2 - x_0 y_2^2 + 2x_1 y_0 y_1 - 4x_1 y_0 y_2 + 2x_1 y_1 y_2 \\ - x_2 y_0^2 + 2x_2 y_0 y_1 + x_2 y_0 y_2 - 2x_2 y_1^2$$

$$f = -(x_0^2 y_2 - 2x_0 x_1 y_1 - 2x_0 x_1 y_2 - x_0 x_2 y_0 + 4x_0 x_2 y_1 - x_0 x_2 y_2 + 2x_1^2 y_0 \\ + 2x_1^2 y_2 - 2x_1 x_2 y_0 - 2x_1 x_2 y_1 + x_2^2 y_0)$$

Transformation matrices (Rotation)

$$R_{\theta} = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\cos \theta = \sqrt{\frac{a}{a+b}}$$

$$\sin \theta = -\text{signum}((a+b)h) \sqrt{\frac{b}{a+b}}$$

$$\text{signum}(x) = \begin{cases} -1 & \text{if } x < 0 \\ +1 & \text{otherwise} \end{cases}$$

Transformation matrices (Translation)

$$T = \begin{pmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$x_0 = \frac{g'}{a + b}$$

$$y_0 = \frac{1}{2f'} \left(c - \frac{g'^2}{a + b} \right)$$

$$\begin{pmatrix} g' \\ f' \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} g \\ f \end{pmatrix}$$

Transformation matrices (Dilation/Scaling)

$$D = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\lambda = -\frac{a+b}{2f'}$$

More numbers

Test cases	SDF Generation time(ms)					avg.(ms)	Performance
Tiger (ARM)	1591.877103	1570.867658	1585.281014	1592.140317	1574.57006	1582.94723	1.47x
Tiger (Skia)	2323.360443	2324.164629	2330.242872	2332.234621	2301.774502	2322.355413	
Countries(ARM)	1543.180108	1552.965522	1568.936348	1563.890934	1554.955125	1556.785607	1.76x
Countries(Skia)	2783.946037	2717.692852	2723.564148	2740.092278	2770.129919	2747.085047	
Vaasa(ARM)	1940.449476	1916.500211	1915.157199	1922.863245	1907.956362	1920.585299	1.25x
Vaasa(Skia)	2417.225361	2453.290224	2302.508593	2434.435368	2424.861908	2406.464291	
Iceland(ARM)	1641.598582	1633.31306	1623.172998	1633.633852	1586.660147	1623.675728	1.98x
Iceland(Skia)	3170.907497	3288.432121	3247.562647	3211.12895	3196.832895	3222.972822	

questions?