Ciallo:

GPU-Accelerated Rendering of Vector Brush Strokes

2025/5

By SHY & ZZZ

Content:

1.Research focus.

2.Background.

3.Related Work.

4.Rendering Algorithms

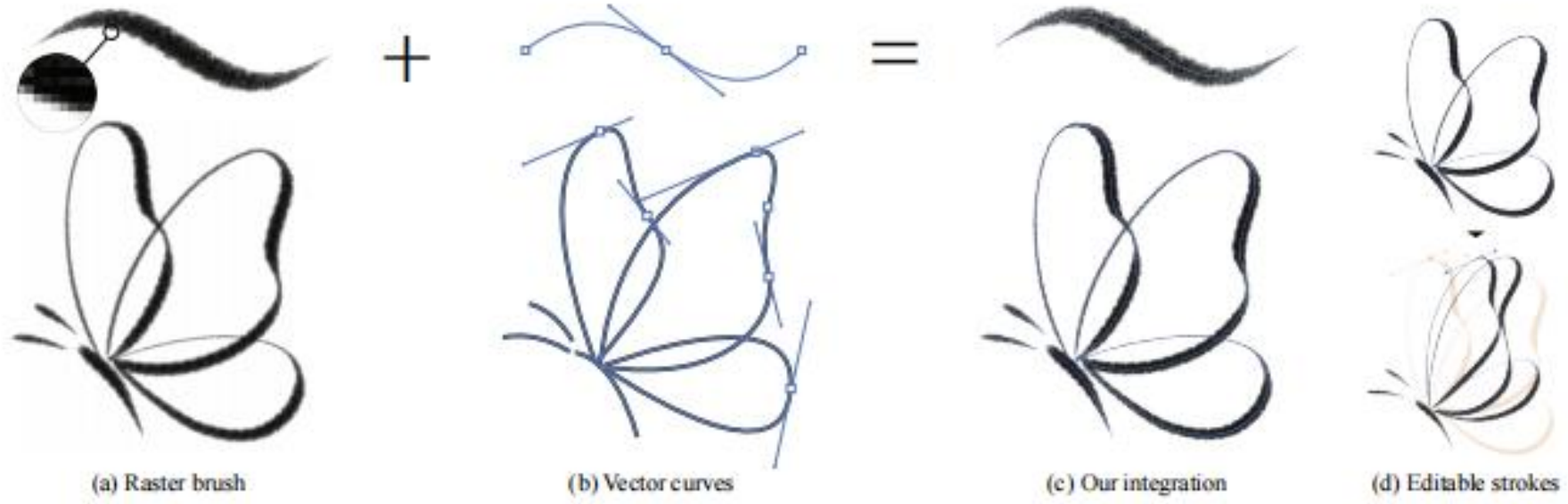
5.Painting system implementation

6.Experiment

7.Discussion

Research focus

This paper introduces novel GPU-based rendering techniques for digital painting and animation to bridge the gap between raster and vector stroke representations.



Background

Brush stroke rendering is fundamental in digital painting and is critical for supporting the desired painting process and result.

Contemporary digital painting applications represent drawings using either vector or raster graphics. Each representation offers complementary advantages and disadvantages.

For instance, vector graphics provide easier manipulation, whereas raster graphics provide higher expressiveness

Background

Many painting programs have attempted to integrate these benefits.

For example, the vector graphics program **Adobe Illustrator** provides versatile brushes, and the raster graphics program **Clip Studio Paint** develops vector layers that record brush strokes with vector curves.

However, their integration is unsatisfying due to technical limitations. Adobe users have long complained about the inability to use Photoshop's brushes in Illustrator [Adobe Commu_x0002_nity 2009]. Clip Studio Paint supports neither real-time rendering of strokes nor filling colors on vector layers.

Sec.	Program	Vector	GPU	Raster brush	Open-source
2.1.1	Inkscape	✓	–[1]	–	✓
	Synfig Studio	✓	–	–	✓
	Adobe Illustrator	✓	✓	–	–
	Affinity Designer	✓	✓	–	–
2.1.2	Adobe Photoshop	–	–	✓	–
	Adobe Fresco	–	–	✓	–
	Krita	–	–	✓	✓
	Clip Studio Paint	✓	–	✓	–
	OpenToonz	✓	–	✓	✓
2.1.3	Corel Painter	–	✓	✓	–
	BlackInk	–	✓	✓	–
2.1.4	Disney Meander	✓	✓	✓	–
	Ours	✓	✓	✓	✓

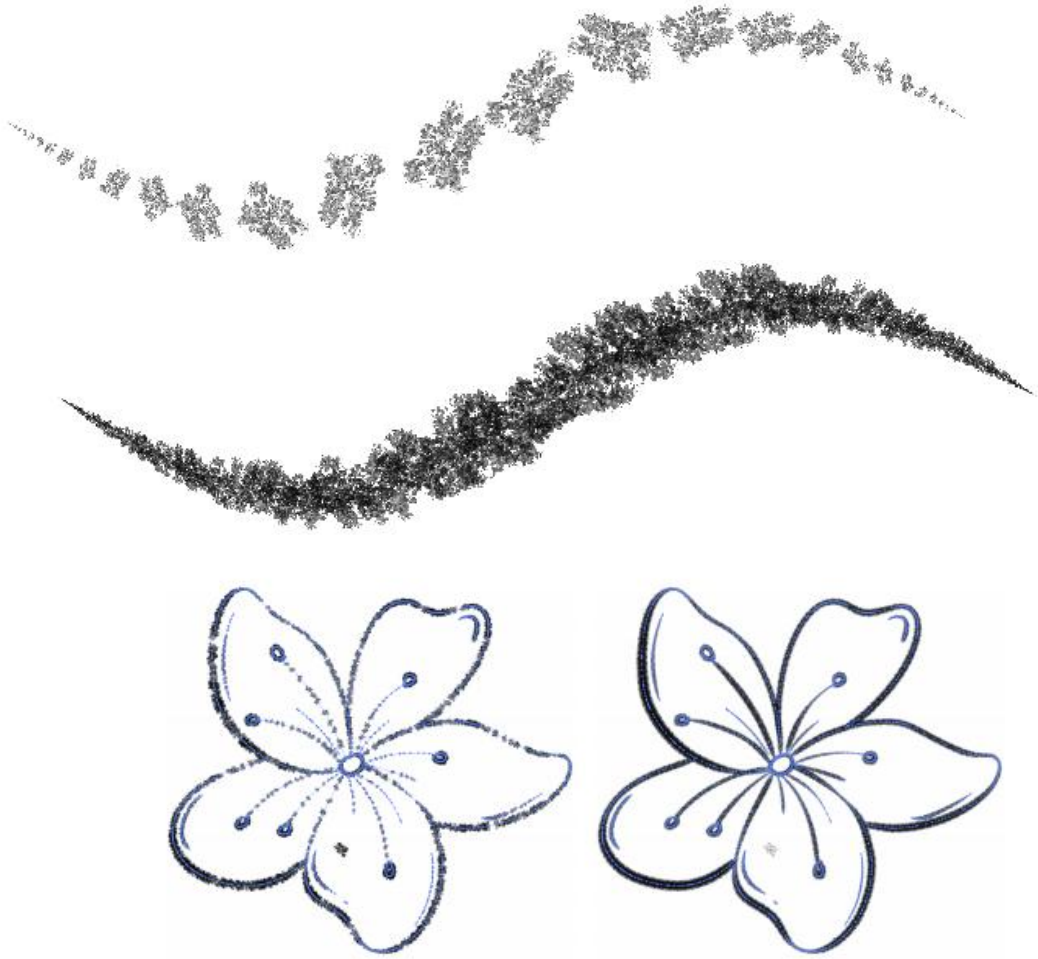
Stroke Rendering

Standard Vector Strokes
- SVG

Stamp Strokes
- sweep & **stamp**

GPU-Accelerated Stamp Strokes

Integration
- combine all three advantages
- and open-source

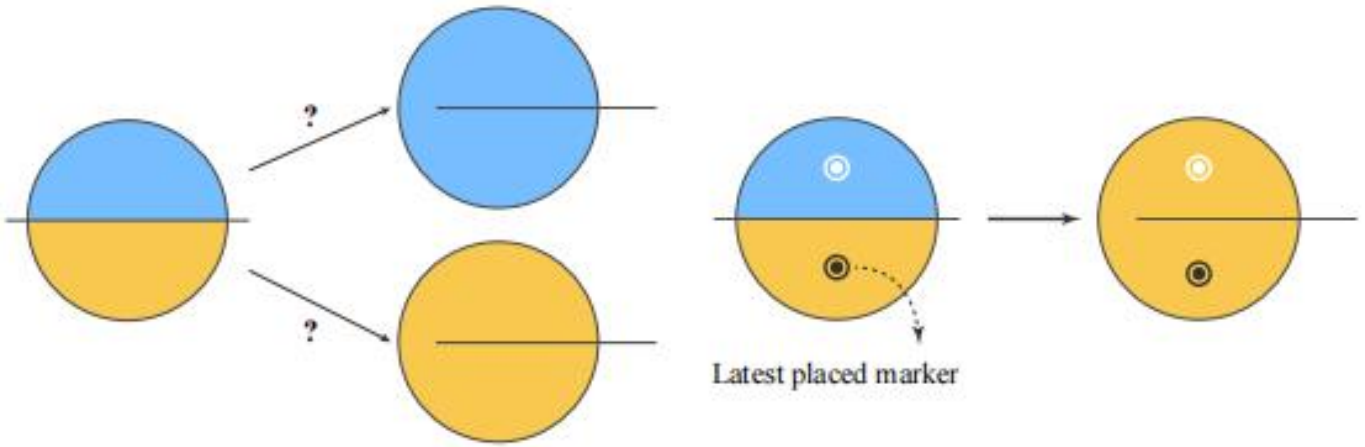


Blender Grease Pencil ours

Painting Systems

Besides stroke rendering, color filling is another fundamental element in paint programs.

"the problem of fill and stroke assignment (matching) is inherently ill-posed"Asente et al. [2007]



(a) The ill-posed case (b) Our solution

Rendering Algorithms

- vanilla
- stamp
- airbrush

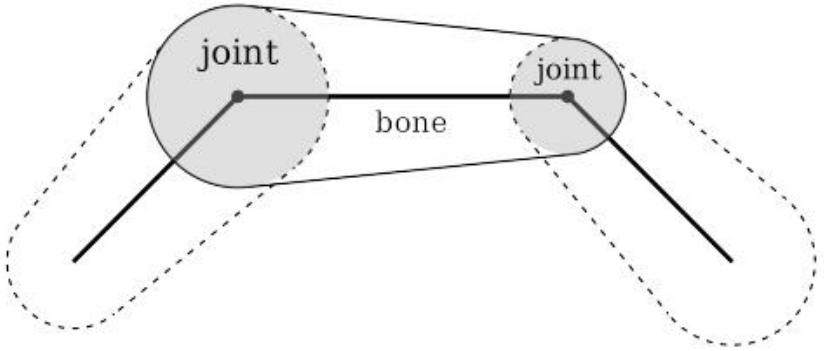
For the original stroke data during drawing, we represent it with a polyline, and the set of points on the polyline is denoted as $V = \{v_0, v_1, \ldots\}$。

And each vertex v_i corresponds to a position p_i and a radius r_i.

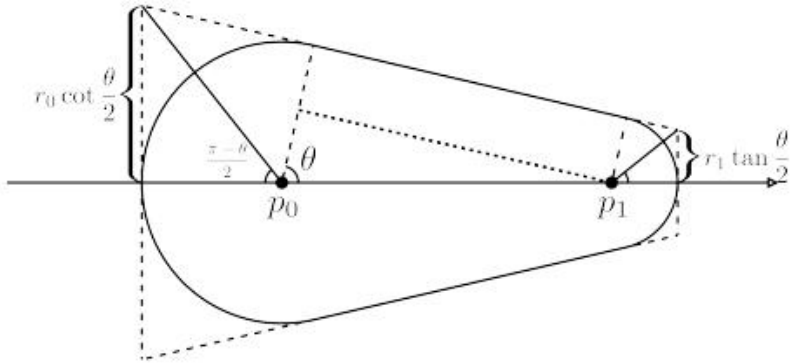
The order of draw calls follows the original stroke order.

Rendering Algorithms

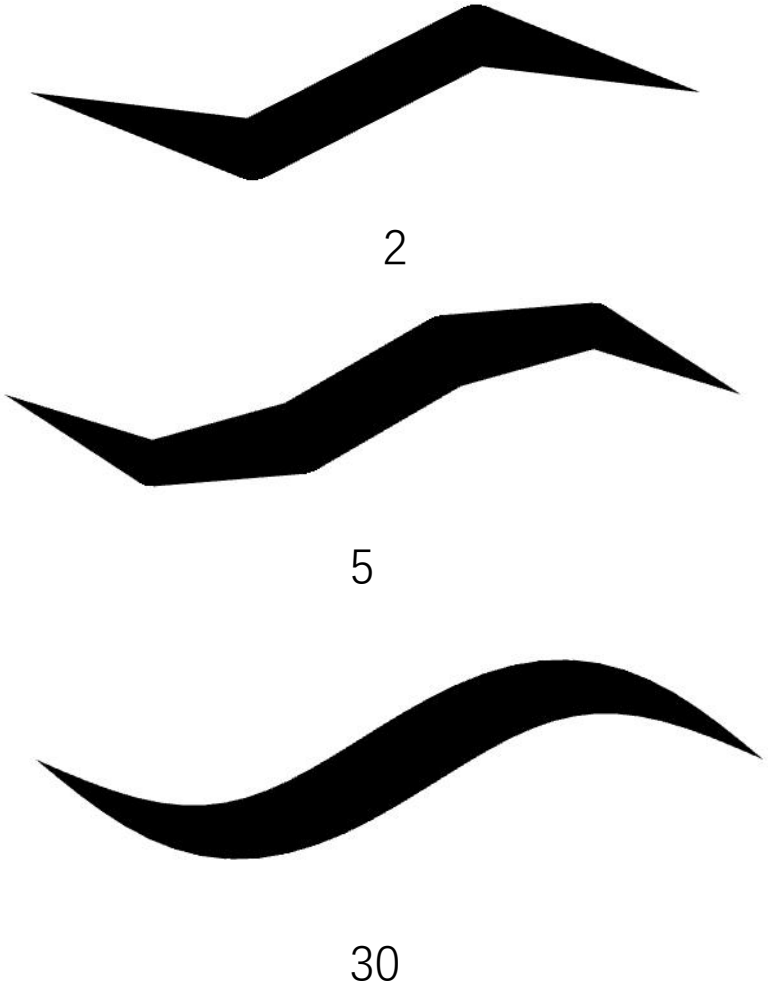
Vanilla



(a) Edge connection.



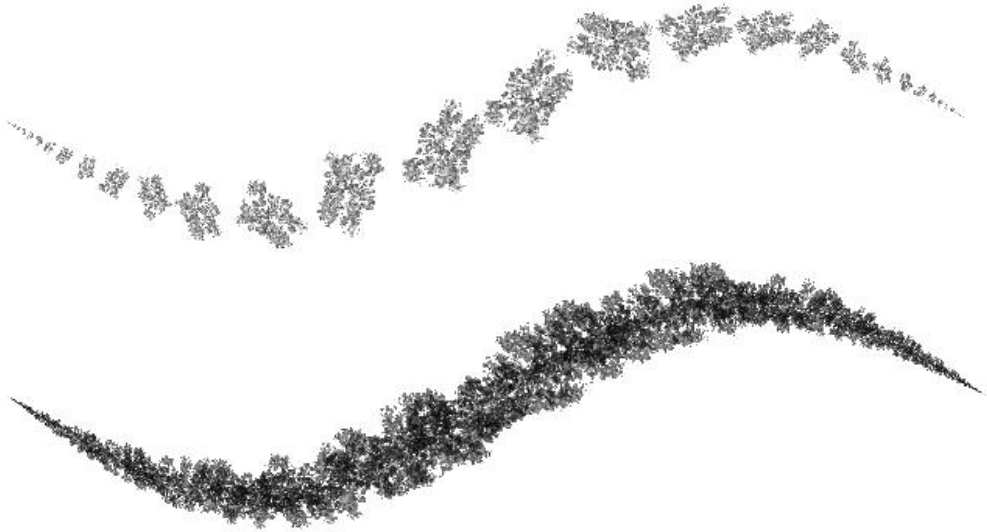
(b) Parameters to place the trapezoid.



Rendering Algorithms

Stamp

Typical
Our GPU-Accelerated

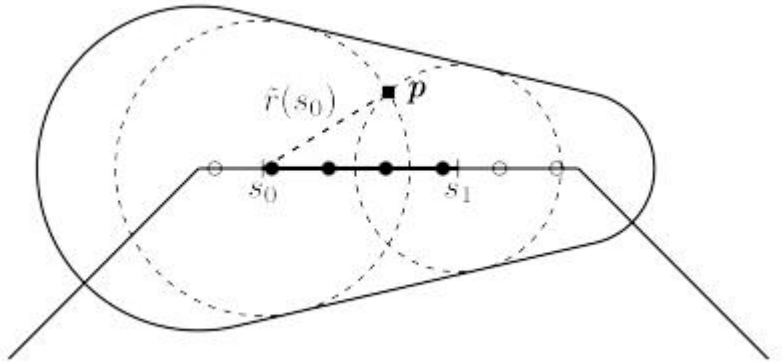


For better performance, an invoked pixel should only sample stamps that can cover it, rather than looping through all stamps.

$$\tilde{r}(s_0)^2 = (x - s_0)^2 + y^2$$

$$\tilde{r}(s_1)^2 = (s_1 - x)^2 + y^2$$

$$\tilde{r}(s_i) = (1 - s_i/L)r_0 + (s_i/L)r_1,$$



Airbrush

Airbrush is a special type of stamp brush whose footprint is a dot with transparency or a transparent gradient from its center to rim.

For better performance, we derive a mathematically continuous representation of airbrush to avoid excessive sampling when the stamp interval is infinitely small.

edge whose length is L
the number of stamps as n
$\Delta L = L/n$
The transparency value alpha at position p is denoted as A(p).

$$A(\boldsymbol{p}) = 1 - \prod_{i=1}^{n} (1 - A_s(\boldsymbol{d}_i))$$

Denote the alpha density value as α .
Let $A\ s\ (\boldsymbol{d}\ i\ ) = \alpha\ s\ (\boldsymbol{d}\ i\ )\Delta$?

$$A(\boldsymbol{p}) = 1 - \prod_{i=1}^{n} (1 - \alpha_s(\boldsymbol{d}_i)\Delta L).$$

Airbrush

edge whose length is L
the number of stamps as n
$\Delta L = L/n$
The transparency value alpha at position p is denoted as A(p).

$$A(\boldsymbol{p}) = 1 - \prod_{i=1}^{n} (1 - A_s(\boldsymbol{d}_i))$$

Denote the alpha density value as α .
Let $A_s(\boldsymbol{d}_i) = \alpha_s(\boldsymbol{d}_i)\Delta L$

$$A(\boldsymbol{p}) = 1 - \prod_{i=1}^{n} (1 - \alpha_s(\boldsymbol{d}_i)\Delta L).$$

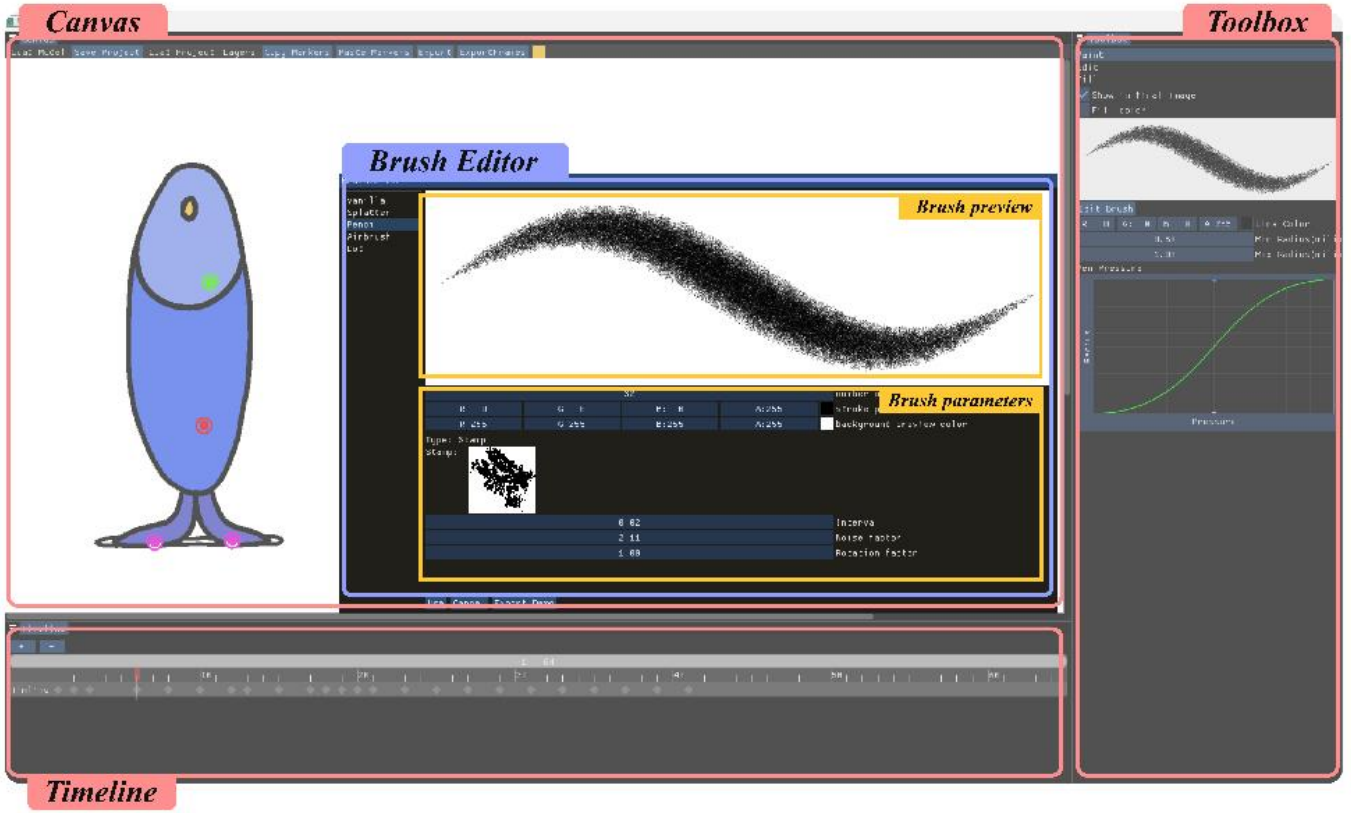
$\boldsymbol{p} = (x, y)$ $\boldsymbol{d}_i = (x - l_i, y)$

apply the Volterra product integral

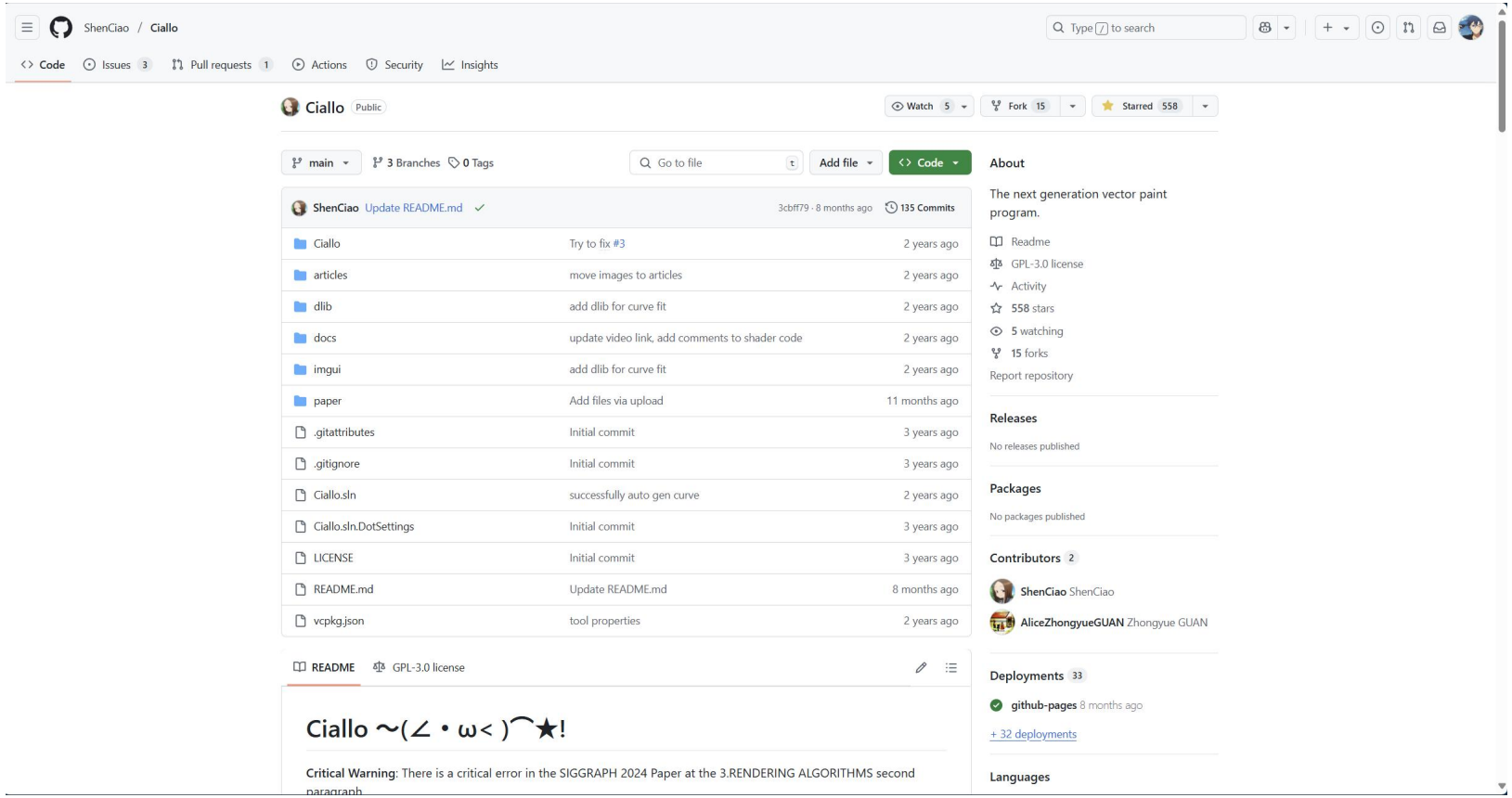
$$A(x, y) = 1 - \exp\left(-\int_0^L \alpha_s(x - l, y)\mathrm{d}l\right).$$

Painting system implementation

The writers developed a prototype painting program with vector fill to showcase our stroke rendering technique.



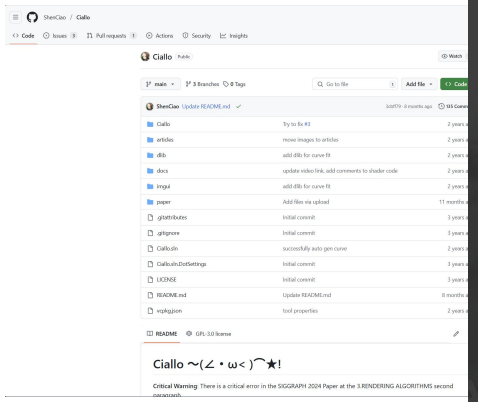
Experiment



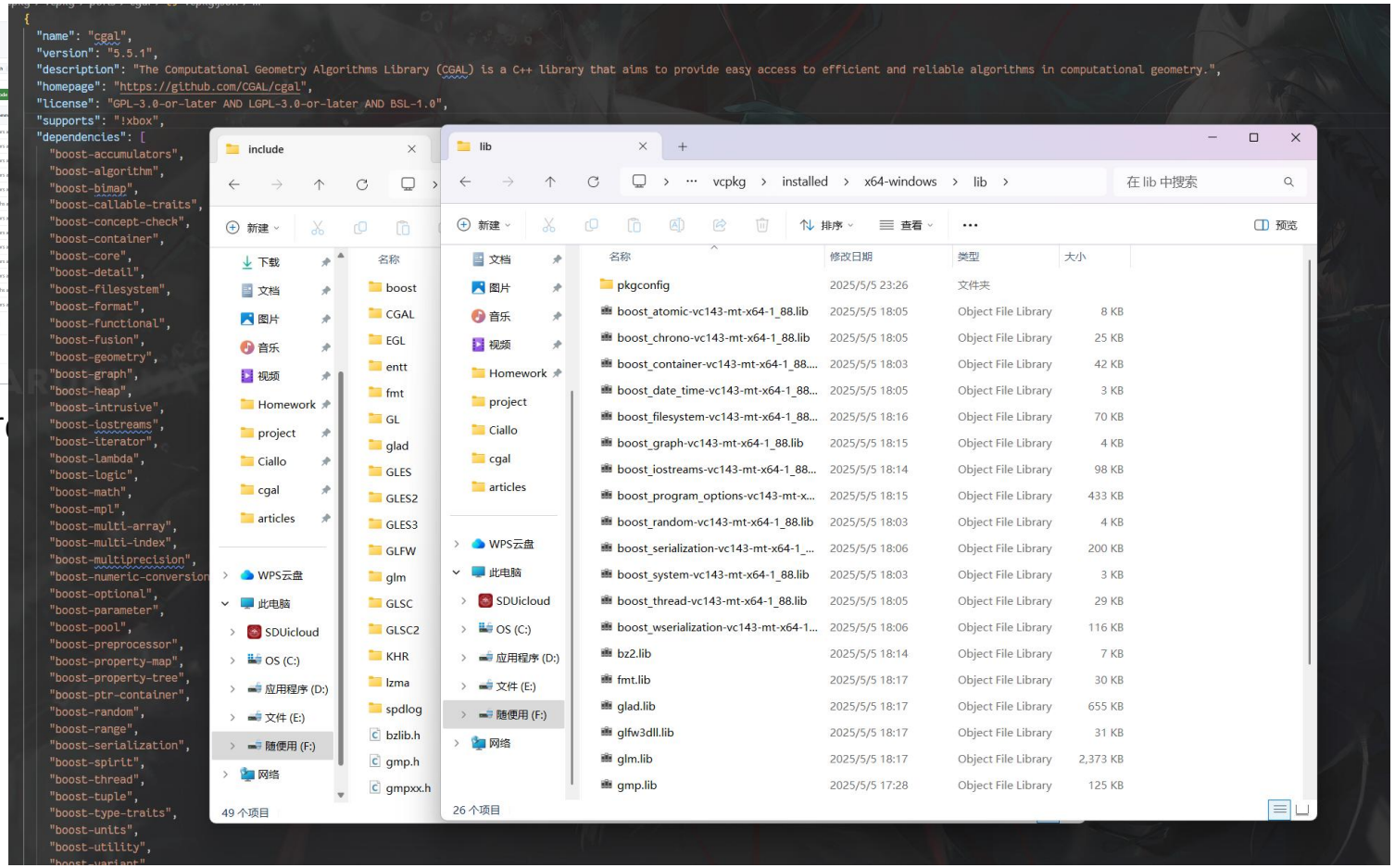
Ciallo	Try to fix #3	2 years ago
articles	move images to articles	2 years ago
dlib	add dlib for curve fit	2 years ago
docs	update video link, add comments to shader code	2 years ago
imgui	add dlib for curve fit	2 years ago
paper	Add files via upload	11 months ago
.gitattributes	Initial commit	3 years ago
.gitignore	Initial commit	3 years ago
Ciallo.sln	successfully auto gen curve	2 years ago
Ciallo.sln.DotSettings	Initial commit	3 years ago
LICENSE	Initial commit	3 years ago
README.md	Update README.md	8 months ago
vcpkg.json	tool properties	2 years ago

Get Source Code from github

Experiment

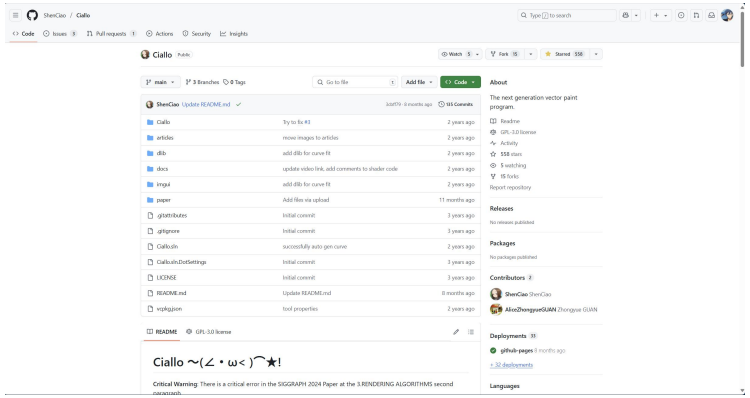


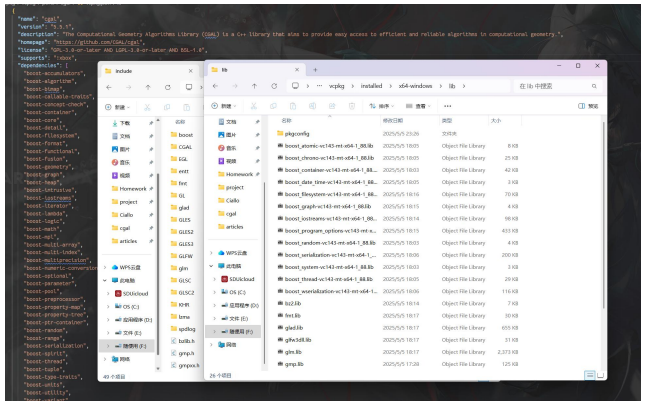
Get Source Code fr



Install the Dependencies

Experiment

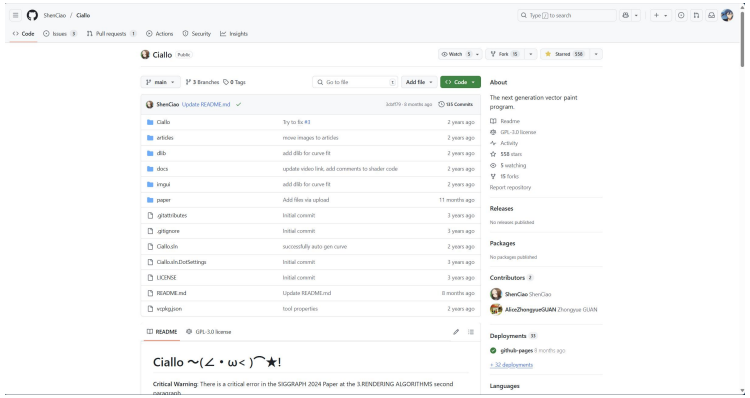




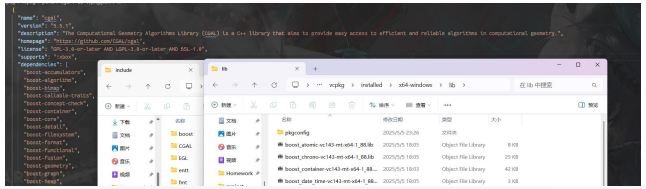
Get Source Code from github → Install the Dependencies

Fix the code error
at first it couldn't even compile!

Experiment



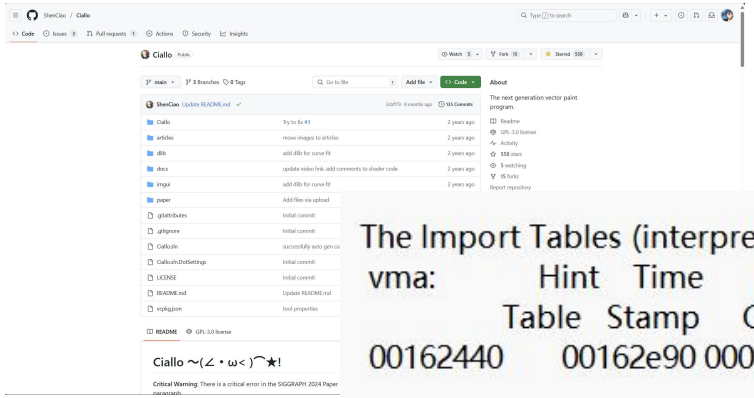
Get Source Code from github



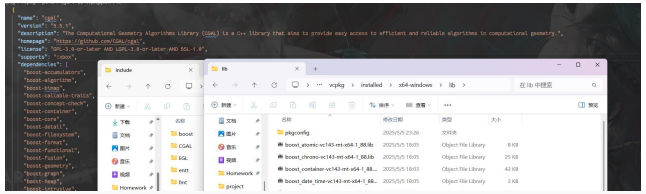
Fix the code error at first it couldn't even compile!

Fix the dll error

Experiment



Get Source C

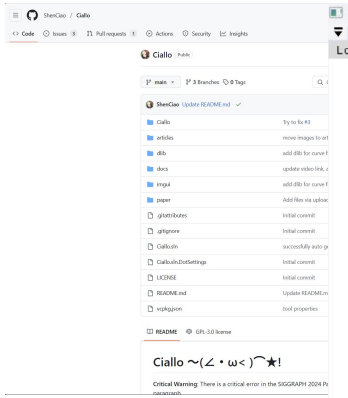



```
The Import Tables (interpreted .rdata section contents)
 vma:            Hint    Time      Forward  DLL       First
                 Table   Stamp     Chain    Name      Thunk
 00162440      00162e90 00000000 00000000 001631b2 001218a8

        DLL Name: fmt.dll
        vma:  Hint/Ord Member-Name Bound-To
        16316e     43  ?format_system_error@v11@fmt@@YAXAEAV?$buffer@D@detail@12@HPEBD@Z
        163040     58  ?vformat@v11@fmt@@YA?AV?$basic_string@DU?$char_traits@D@std@@V?$allocator@D@2
@@std@@V?$basic_string_view@D@12@V?$basic_format_args@Vcontext@v11@fmt@@@12@@Z
        1630de     10  ??$vformat_to@D@detail@v11@fmt@@YAXAEAV?$buffer@D@012@V?$basic_string_view@D@12@V?
$basic_format_args@Vcontext@v11@fmt@@@12@Vlocale_ref@012@@Z

 00162454      00162eb0 00000000 00000000 00163292 001218c8
```

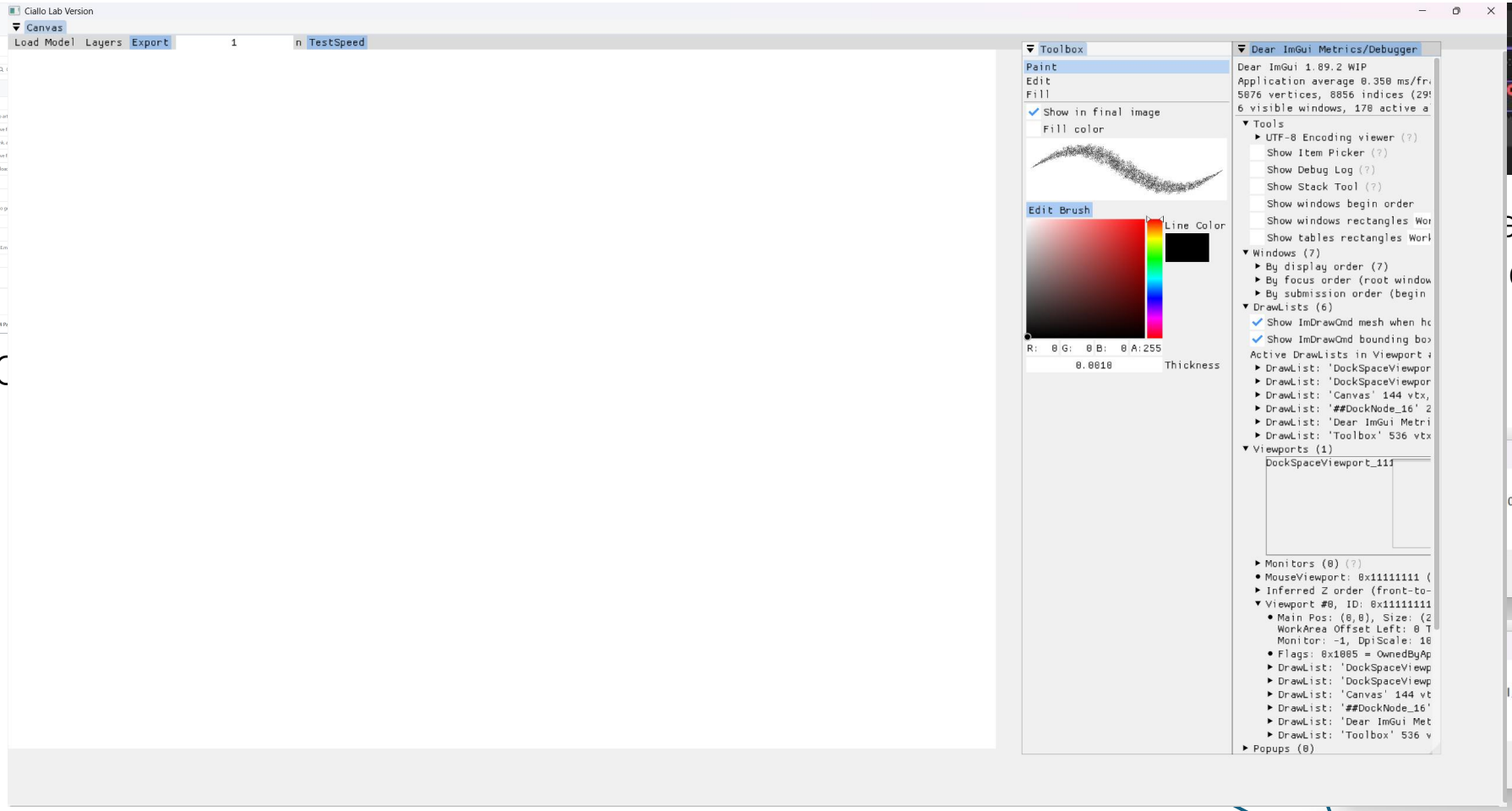
One more dll error
(to fix it we even used objdum!)

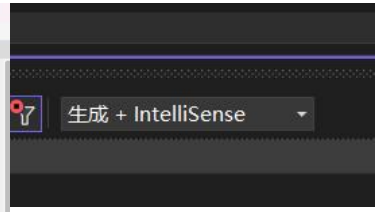
Fix the dll error

Experiment

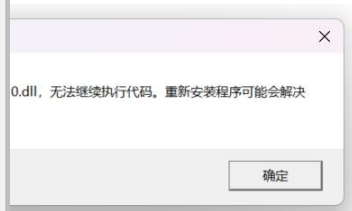


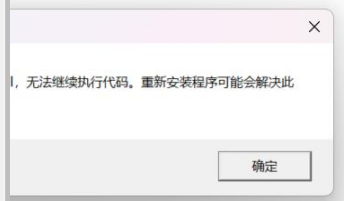
Get Source C





error
dn't
le!



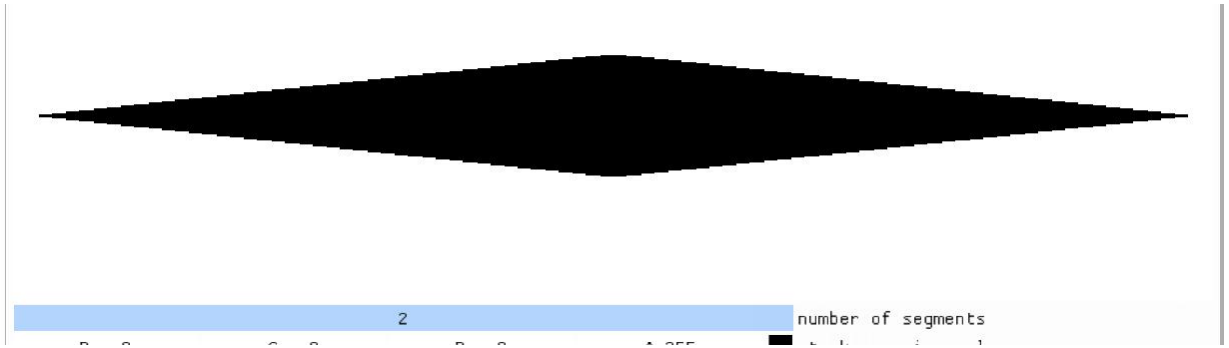


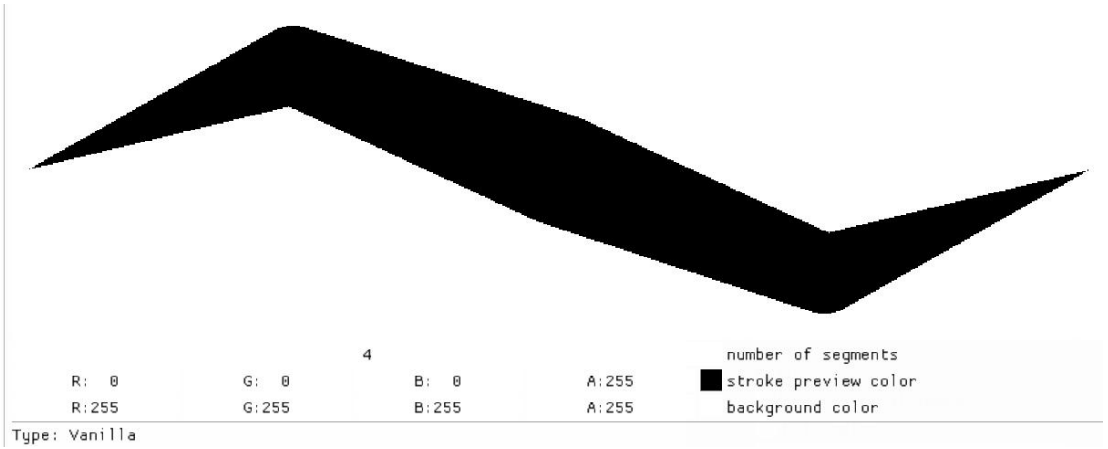
It finally worked!

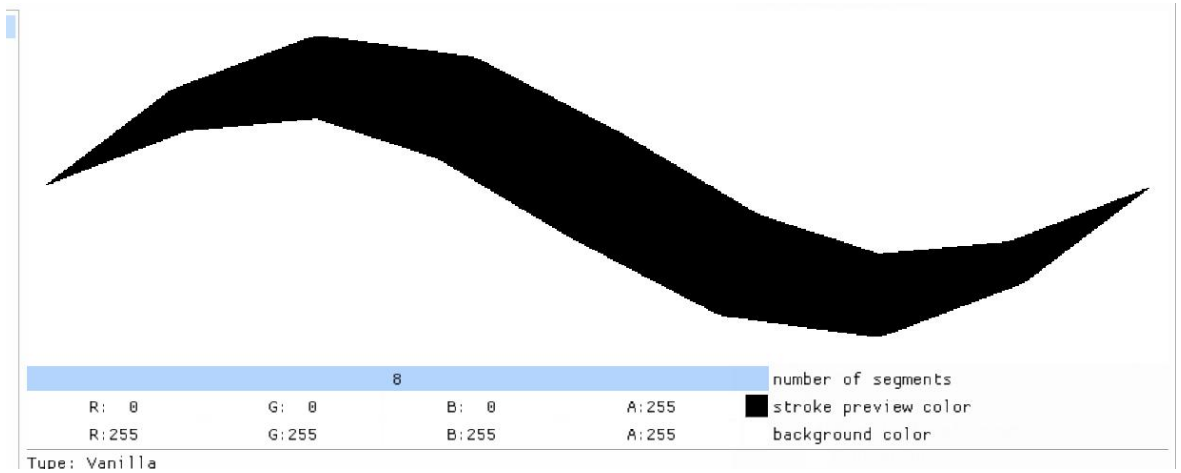
Fix the dll error

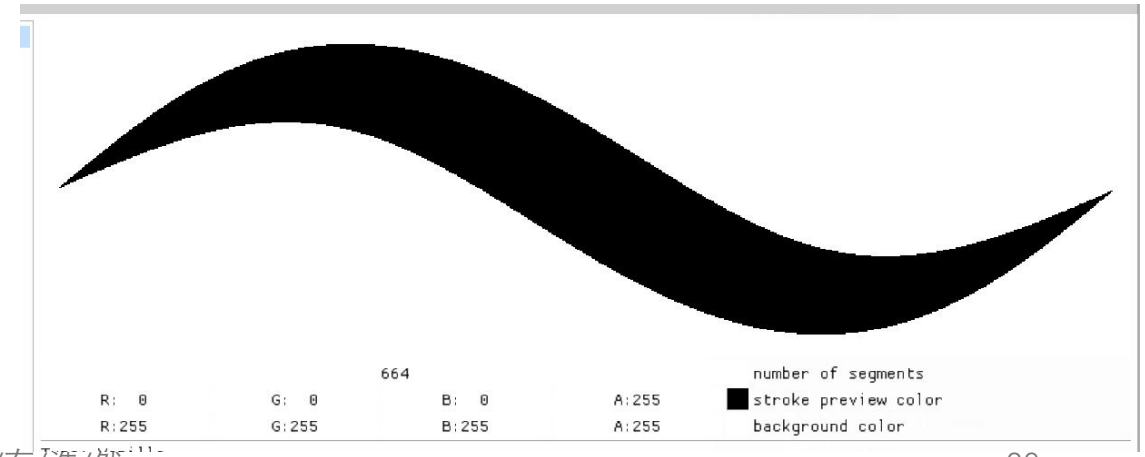
Experiment

Vanila



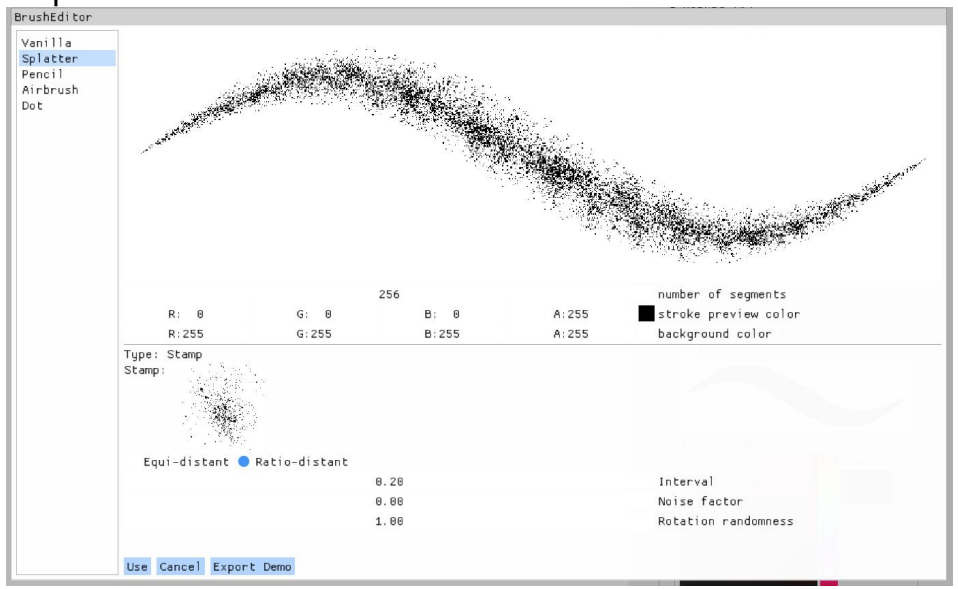


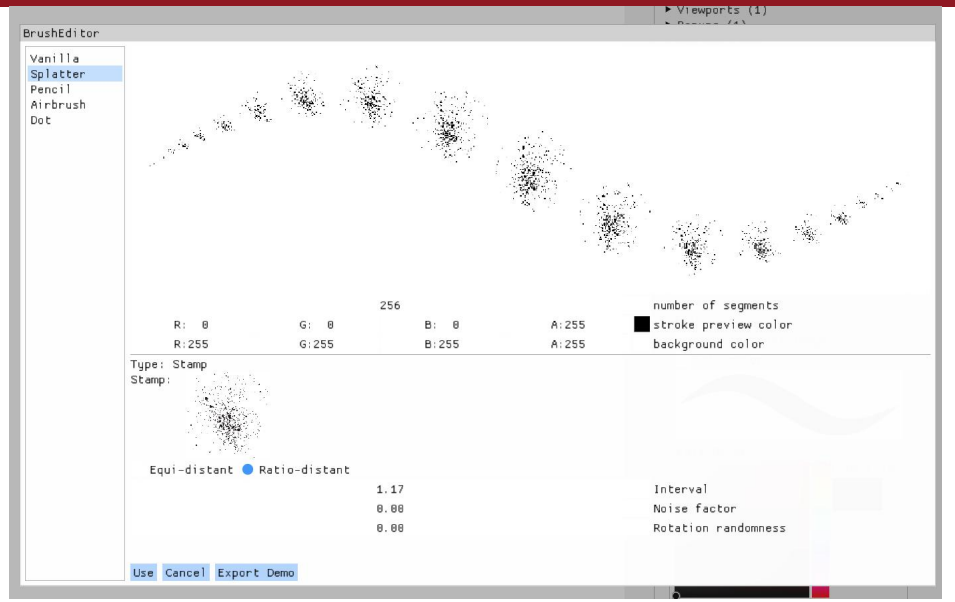


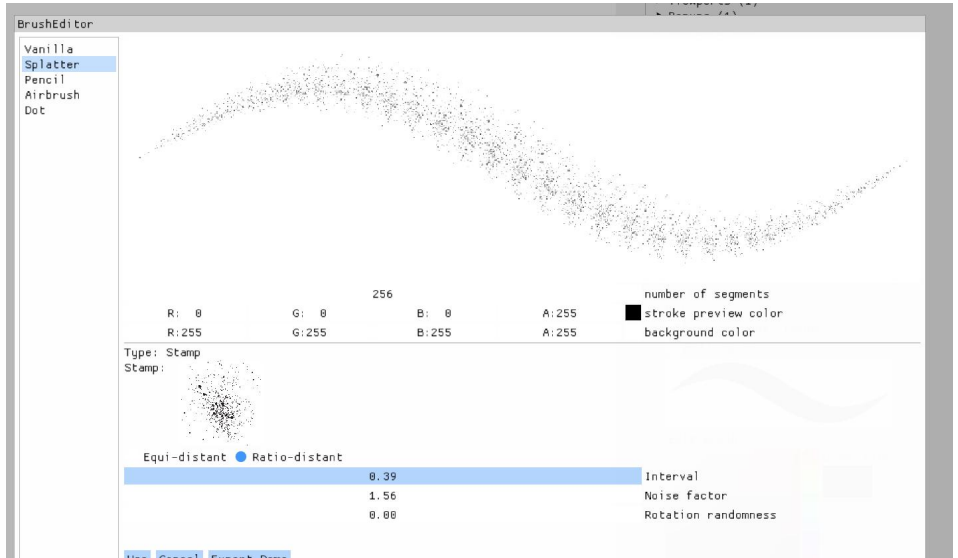


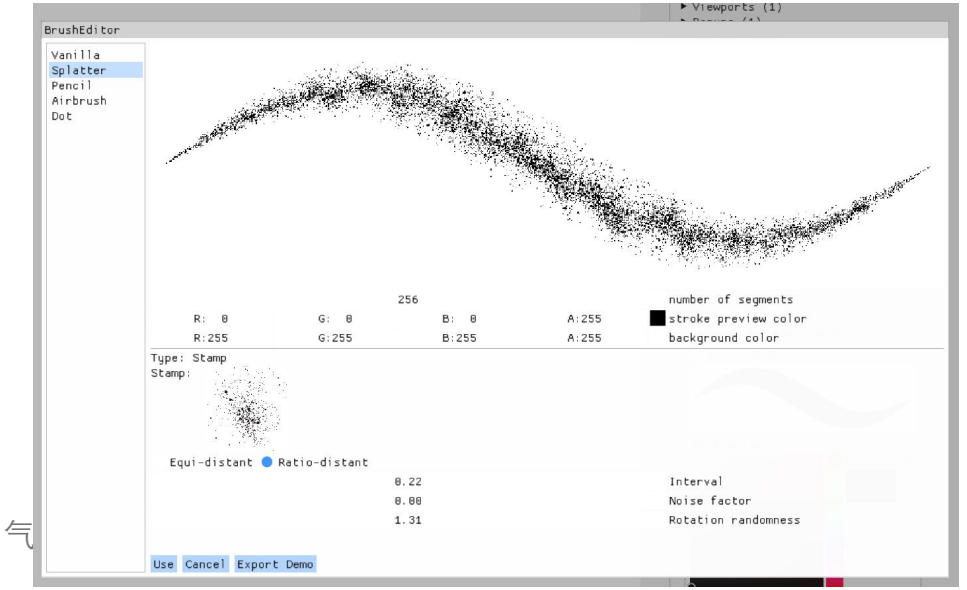
Experiment

Splatter



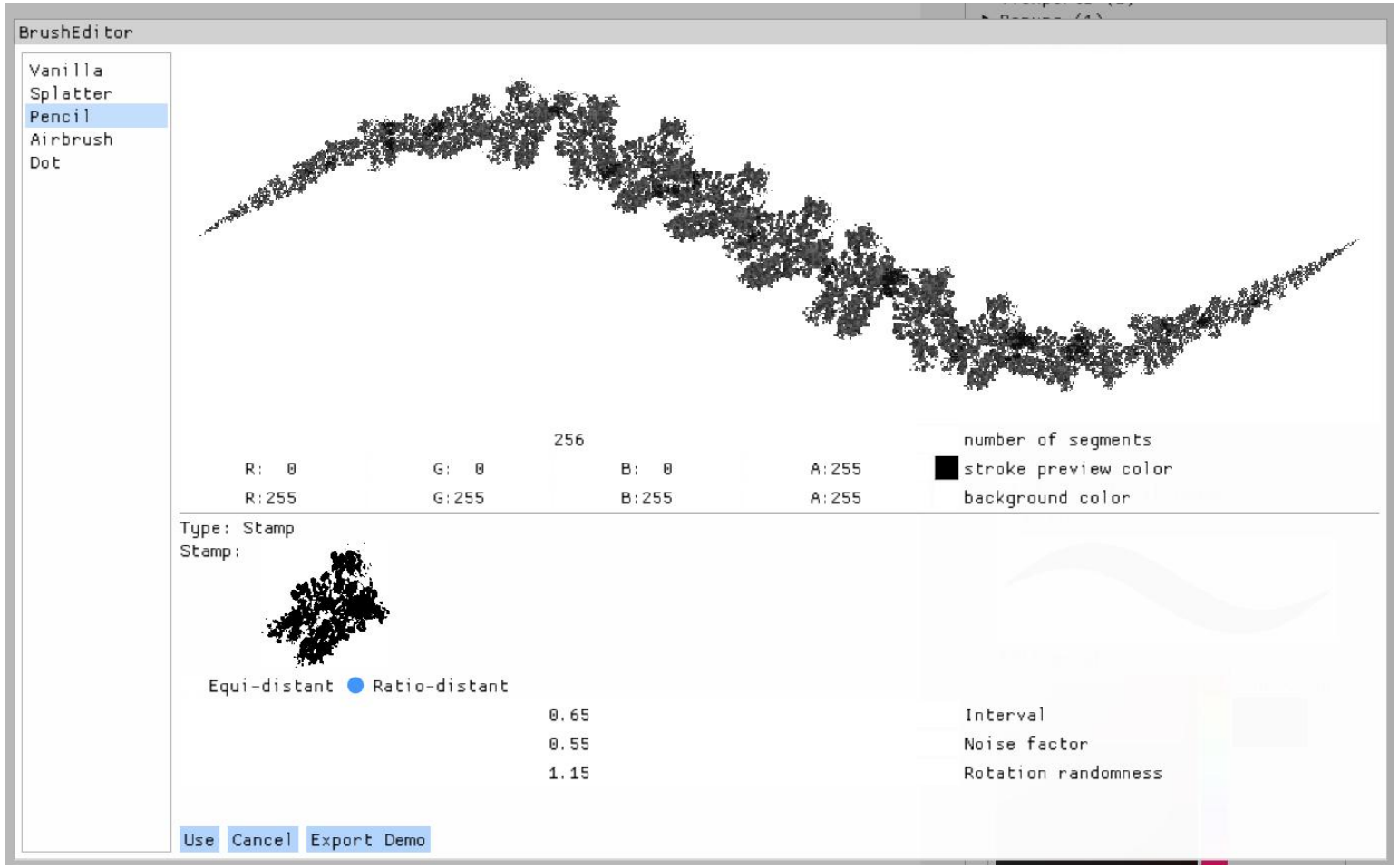






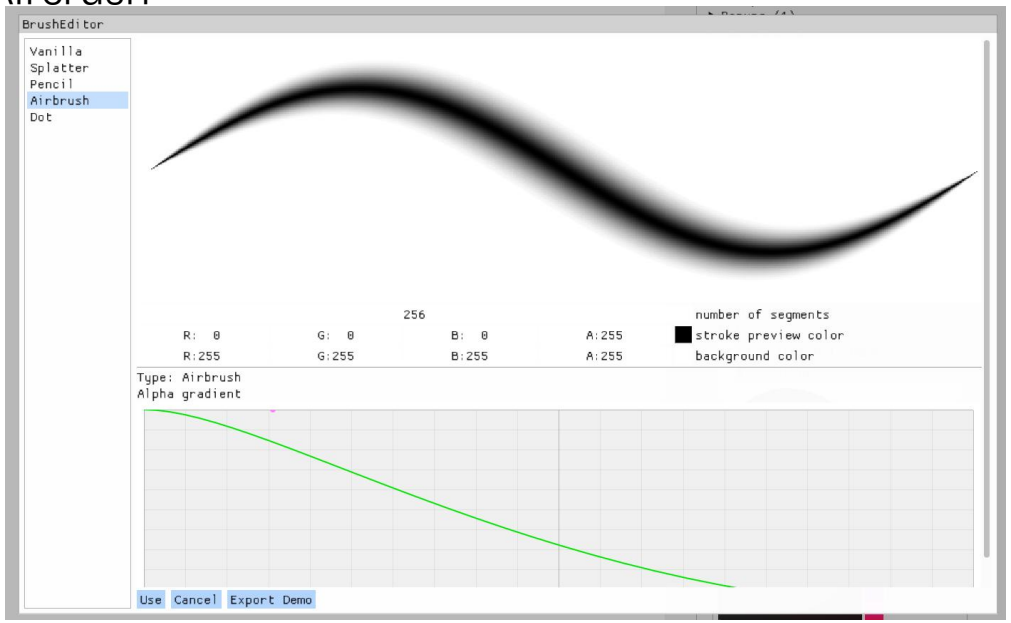
Experiment

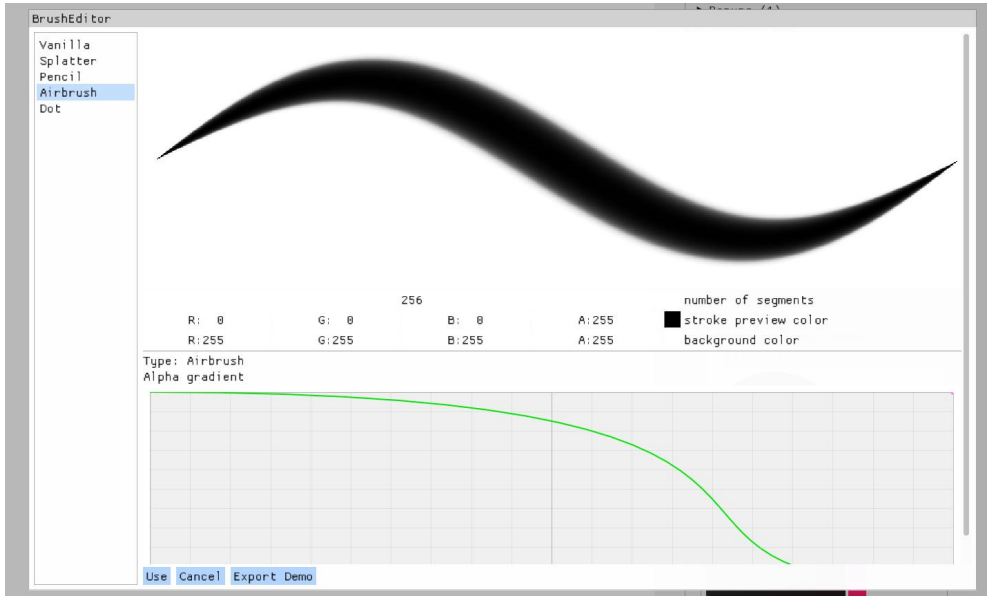
Pencil

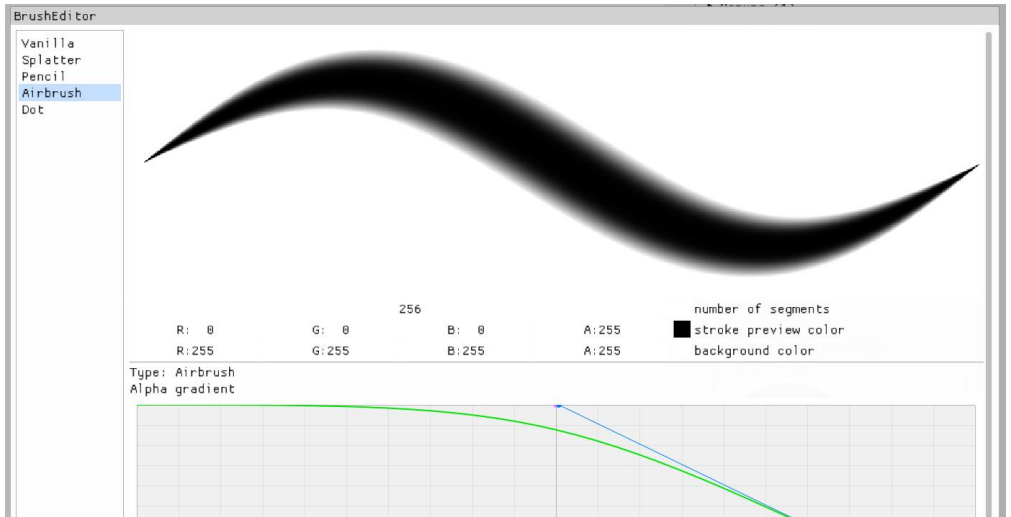


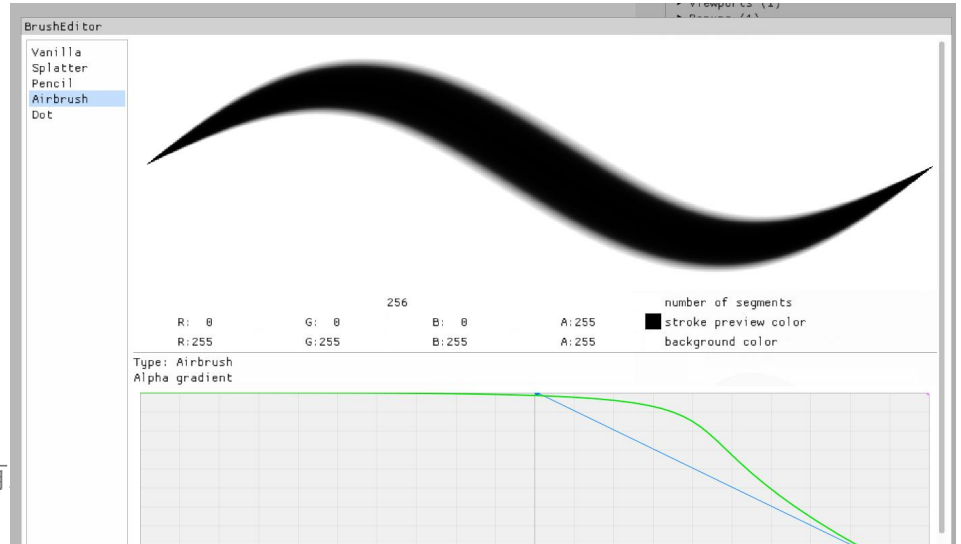
Experiment

Aircrush

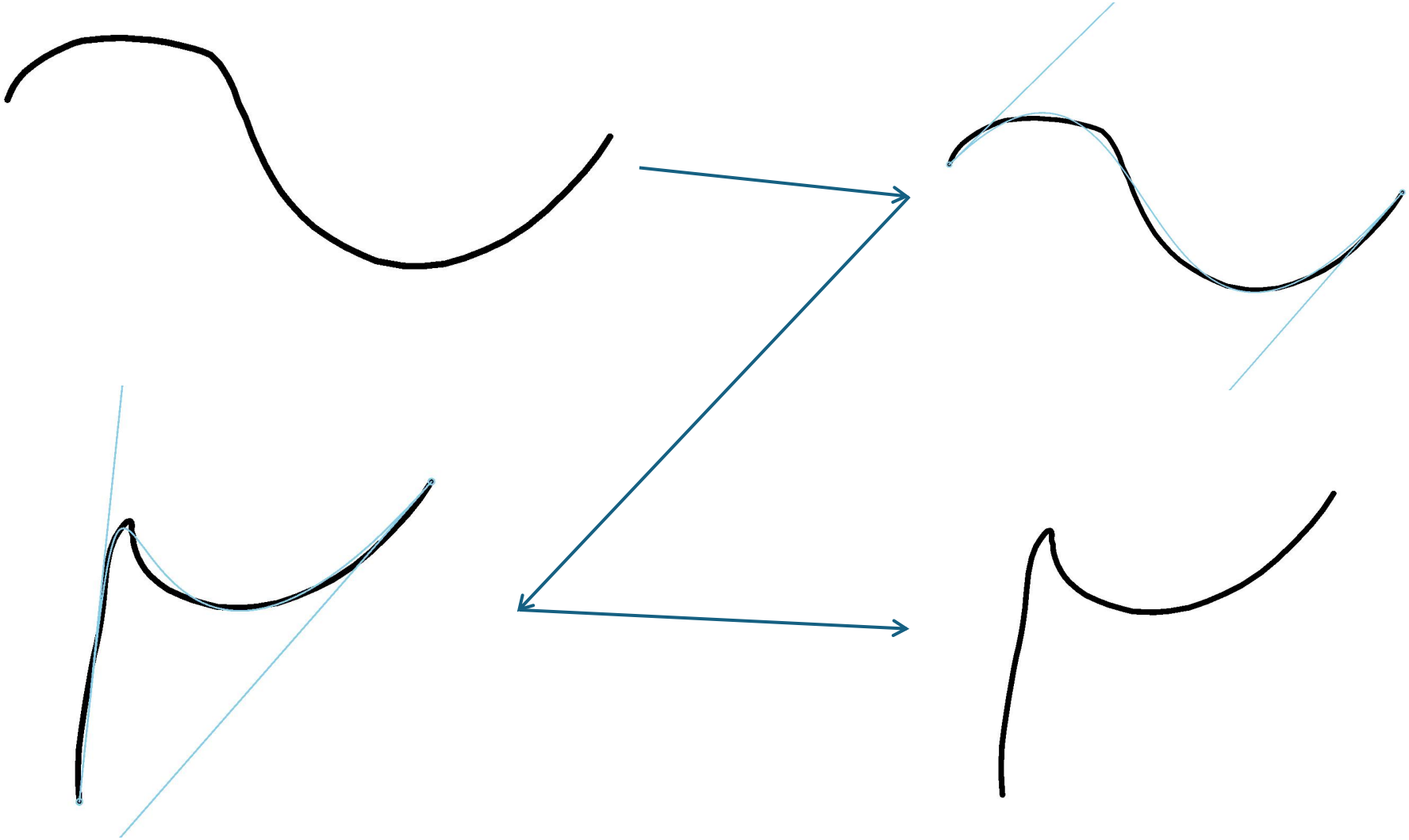


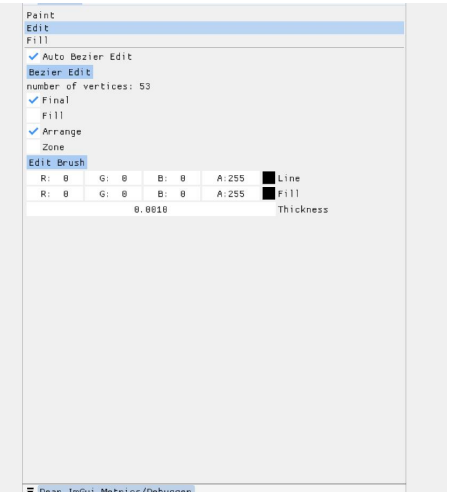




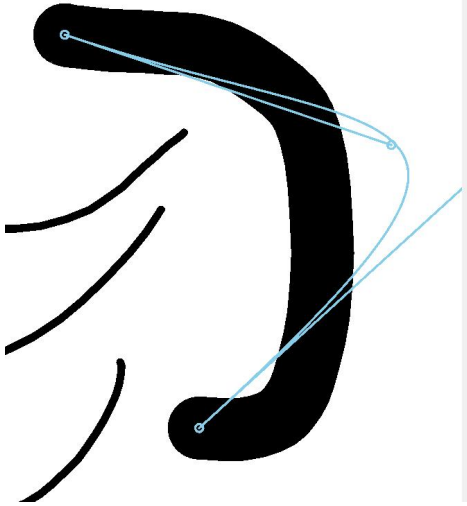


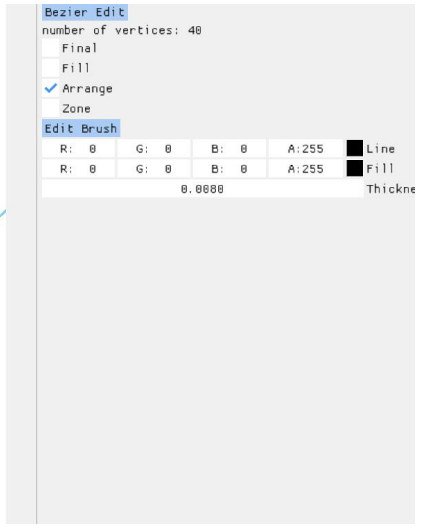
Experiment

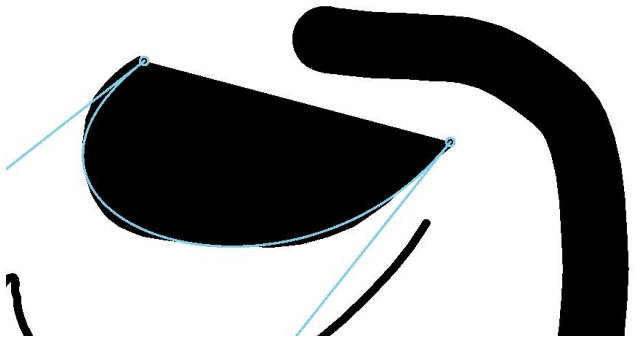


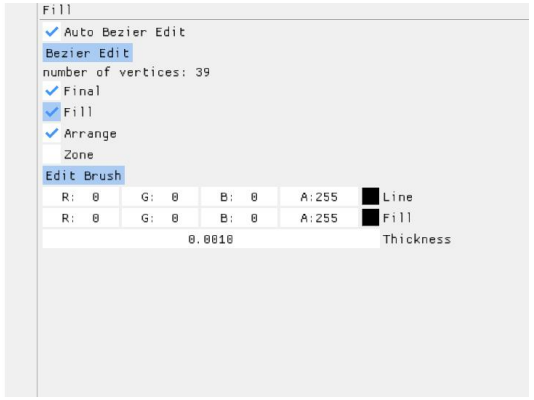

Paint
Edit
Fill
Auto Bezier Edit
Bezier Edit
number of vertices: 53
Final
Fill
Arrange
Zone
Edit Brush
R: 0 G: 0 B: 0 A:255
Line
R: 0 G: 0 B: 0 A:255
Fill
0.0010
Thickness

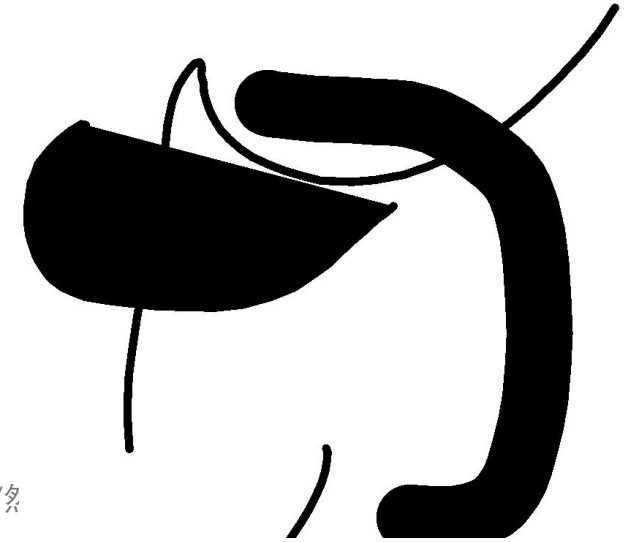
Experiment



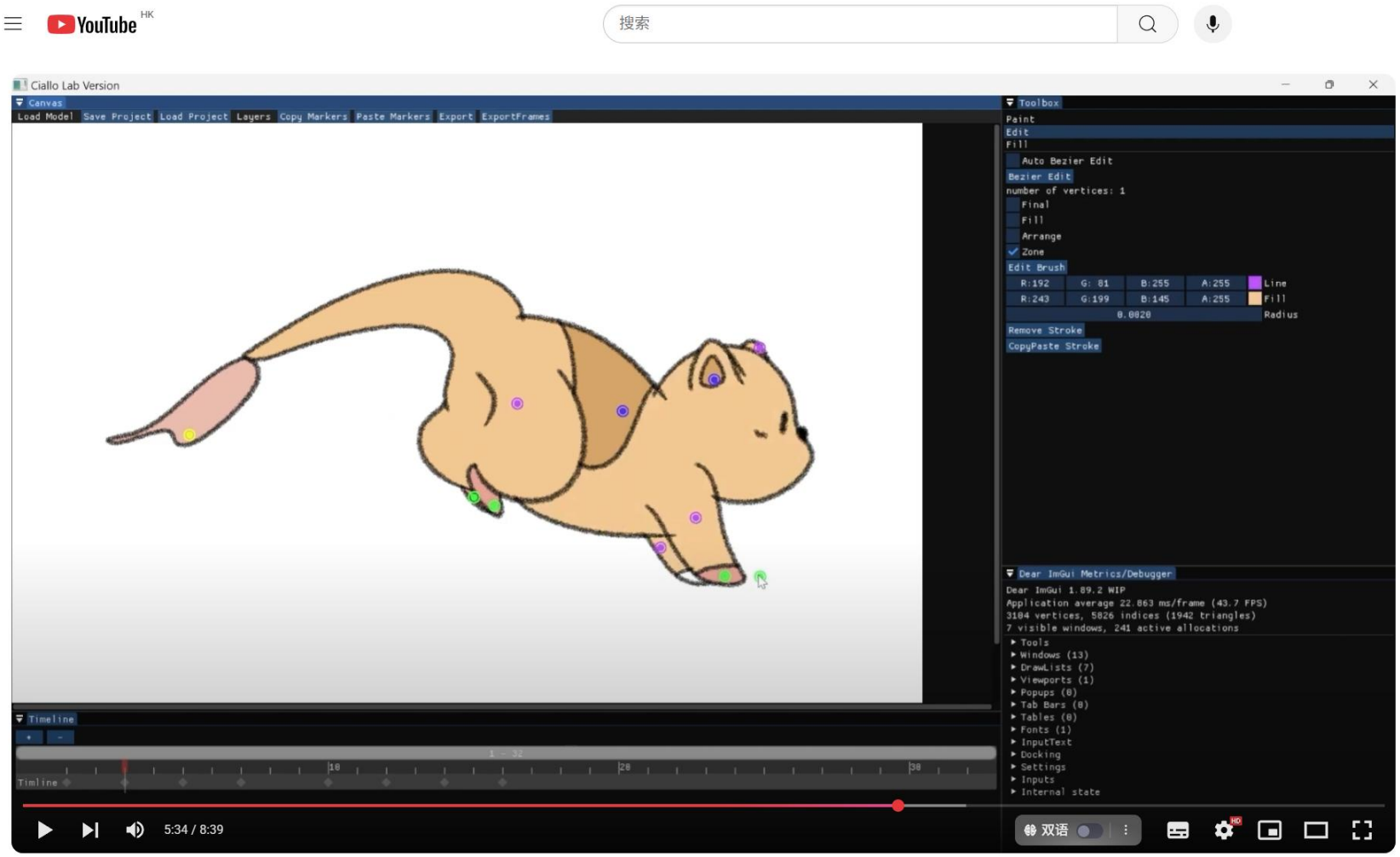

Bezier Edit
number of vertices: 40
Final
Fill
Arrange
Zone
Edit Brush
R: 0 G: 0 B: 0 A:255 Line
R: 0 G: 0 B: 0 A:255 Fill
0.0080
Thickne



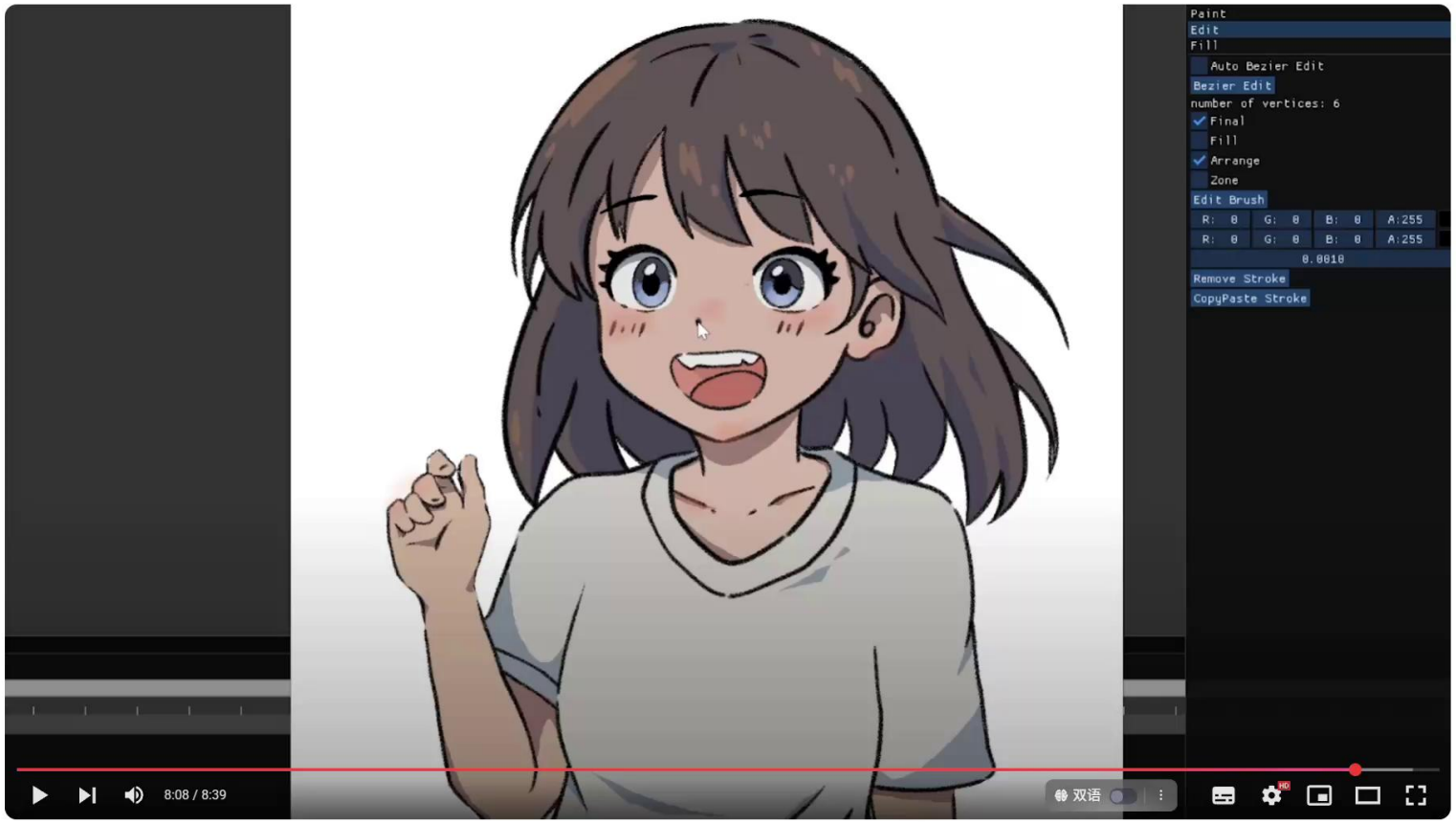

Fill
Auto Bezier Edit
Bezier Edit
number of vertices: 39
Final
Fill
Arrange
Zone
Edit Brush
R: 0 G: 0 B: 0 A:255 Line
R: 0 G: 0 B: 0 A:255 Fill
0.0010
Thickness



Disscussion



Disscussion



Disscussion

CialloVulkan Public archive

Watch 1

master 1 Branch 0 Tags Go to file t Code

ShenCiao Update README.md 539c440 · 2 years ago 83 Commits

Ciallo	Update README.md	3 years ago
articles	Redesigning drawing system	3 years ago
.gitignore	render test triangle with dynamic rendering	3 years ago
Ciallo.sln	initial setup for rendering a triangle	3 years ago
LICENSE	Create LICENSE	3 years ago
README.md	Update README.md	2 years ago
vcpkg.json	Update README	3 years ago

README AGPL-3.0 license

Ciallo

Ciallo～(∠・ω<)⌒★! Anime/Cartoon/2D computer graphics.

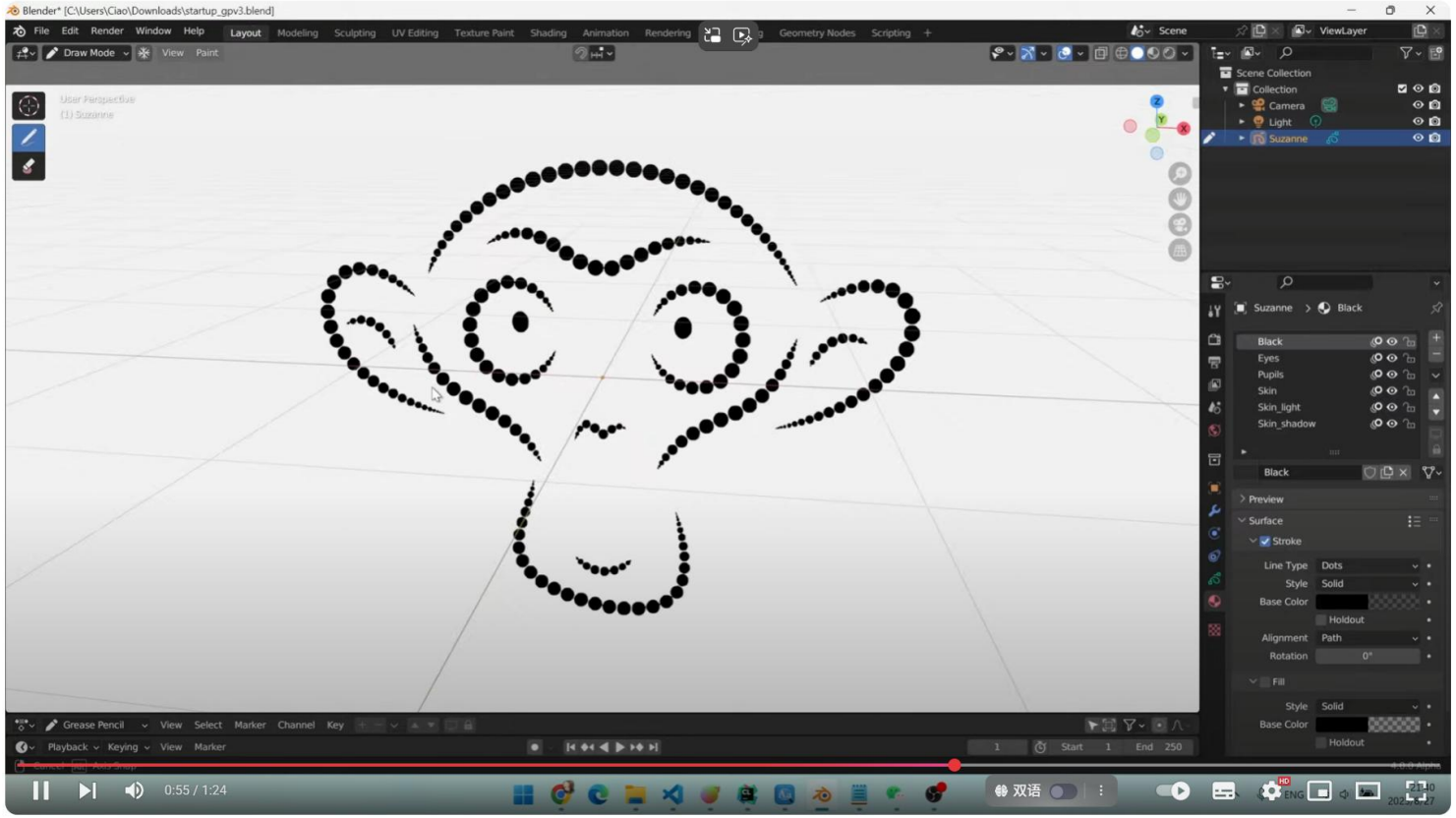
Deprecated

The repository is deprecated since using Vulkan. Check out https://github.com/ShenCiao/Ciallo for OpenGL implementation.

For myself, it's a nice practice about using Vulkan. But I'm pretty glad to switch back to OpenGL, it's late but much better than never.

The lesson I've learned is that **never use raw Vulkan (or DX12) for your research project! (unless necessary)**.

The images or videos below are rendered in Vulkan.



Blender Grease Pencil Stamp Brush Demo

The End