Pixel-perfect ellipses in Krita

Goals

Currently, Krita can't draw pixel-perfect ellipses, the objective of this proposal is to achieve this feature. To be exact, the following points:

1. Make it possible for Krita to draw pixel-perfect ellipses(and other bézier based shapes).
2. Optionally floating-point precision when making pixel arts.

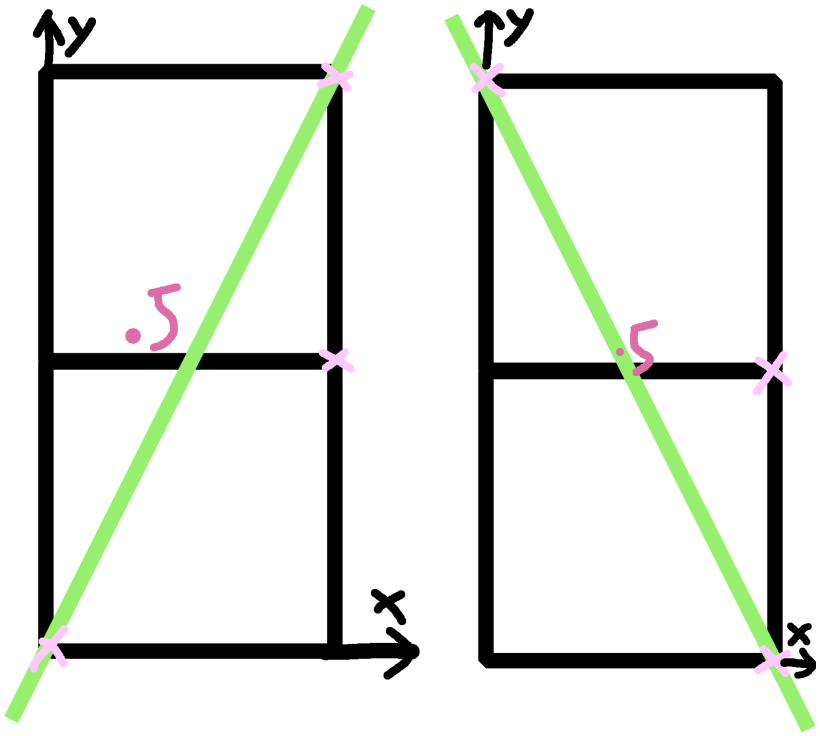
Implementation

Sub Task 1: Improve Bézier Rasterisation

Currently, Krita draws an ellipse by drawing four bézier curves, and each of these bézier curves is then divided into small pieces which can then be approximated with straight lines, and the asymmetric of the ellipse is partly caused by our algorithm for drawing straight lines.

To explain that, imagine we are drawing a line from the origin to (1,2) and a line from (0,2) to (1,0).

At y=1, they both intersect at x=0.5 and thus dotting at (1,1).



(A picture demonstrating this, the pink cross is the actual pixel generated for the line.)

As we can see, rounding 0.5 to the "right" led to the asymmetric, which partly caused the imperfect ellipse.

To eliminate this, we can take round(0.5) to zero when the line is going in certain directions.

Sub Task 2: Disable Float-point Precision In Pixel Arts.

When drawing, Krita saves the ellipse's dimensions in floating-point numbers, but when making pixel arts, this precision could be unnecessary and we can disable this precision by checking if the "Sharpness" -> "Soften edge" attribute equals 0(Which means we are doing pixel arts).

In order to disable the extra precision, we can round coordinates like `m_dragStart` and `m_dragEnd` before we call the action handler.

Optionally, we can add one modifier key to temporary disable floating-point precision. We will ask users to see if they want this feature. If it's not popular, the modifier key will not be introduced.

If "Sharpness" -> "Soften edge" attribute does not equal 0, we can just use the old behaviour, since users are unlikely to care about every single pixel when drawing with a huge sized brush with detailed texture.

Sub Task 3: Make Specialised Algorithms For Common Shapes

Rather than approximating using bézier curves, using a specialised algorithm for drawing common shapes like ellipses will achieve a better result.

The plan is to modify things like the ellipse tools to make them issue an ellipse-painting job instead of issuing a bézier-path-painting job.

Also, the existing `KisPainter::paintEllipse` method has to be modified to use the specialised algorithm. The algorithm I'm planning to use is the midpoint ellipse drawing algorithm with shearing to make it compatible with rotated ellipses, it's specially designed for drawing ellipses thus it's better than the straight-line approximation.

Out of my own interest, I have also designed the following algorithm for drawing ellipses myself. It seems to work with my pen and paper. I'll implement this algorithm and compare it with the midpoint alogrithm with shearing:

1. In rotated coordinates, a vertex of ellipse can be easily obtained by setting y=0. We call it `p`.
2. Get the derivative of the ellipse at point `p` (in the oringinal coordinates)
3. Evaluate the adjacent points of `p` which are on the direction of the derivative, chose the one nearest to the ellipse, dot at that point, make the point `pd`
4. Map pd back to rotated coordinates, extend the connection line of mapped `pd` and origin, find the intersect point on the ellipse, make it `p`, go back to step 2.
5. If we have dotted on the same point for the third time, we must have finished, return.

Timeline

This is my proposed timeline. I'm going to allocate each sub task with 4 weeks of time.

- Week 1-4

  Implement Sub Task 1

- Week 5-8

  Implement Sub Task 2.

- Week 9-12

  Implement Sub Task 3.

In detail, the four weeks of each sub task is made up by:

- Week 1-2 Coding for features
- Week 3 Writing documents and tests
- Week 4 Merging the code and doing fixes if needed

I'll have a final exam in the first week of July, and I have to prepare for it in advance, so I may slow down during these times.

The rest of the time is my summer vacation, so I'll have plenty of time to do this project.

About Me

I'm Xu Che. I'm studying at Fu Zhou University in China. I'm an undergraduate and I'm in my 1st year.

You can contact me via `chrisxuche[at]gmail.com` or Telegram id `Shidao_X`.

I don't have much experience in programming a big project, as I do competitive programming contests most of the time.

I'm familiar with C++(used for competitive programming contests), Rust(used for some of my side projects) and some other PLs.

I'm not familiar with building systems like CMake, I have only used some simple features like `add_executable`, but after reading the document of CMake, I think it won't be hard.

I'm daily driving ArchLinux w/ KDE, and I wanted to contribute to the projects I use every day, that's why I chose this idea from the KDE idealist.

I haven't contributed to the Krita project with any code yet.

My native language is not English, but I think it won't be difficult for me to communicate with mentors.

I'm not submitting another proposal.