2017-09-06

The idea of a brush is to produce a stroke on a surface with certain characteristics. The ideal brush would produce effects in real time, which I almost succeeded for simple designs.

Testing a graphical application as if you were using it poses some serious problems. Let's not get into the details because it is long and boring and very far from the subject of image production of this blog, but I mention this fact to explain why I create a parallel application to test the specific code of creation of brush strokes: the program will draw a line between two predetermined points which will enable me to observe the values of the variables during the progression of the stroke and find bugs or, better, means to optimize the whole. It is also a way to isolate the code that paints from the code that manages the spatial allocation of points and the moments when the stroke begins and ends and, possibly, the pressure of the pen. The clearer the separations between each functional group of the application, the easier it will be to move forward.

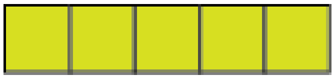
Windows is in charge of giving the application the indications of the location of the mouse or the stylus. These indications occur several times per second and each time my small program is supposed to be able to accomplish all its work by the next indication from Windows. If my program lags, Windows is patient and will drop, if necessary, mouse events. These shortcomings initially meant incomplete lines on the screen.

The main drawing process consists of having a starting and ending point and, in between, a stroke. The starting point is when the mouse button is pressed or when the stylus is placed on the drawing surface. The end point is, of course, when the mouse button is released or the stylus is lifted from the surface.

With the brush, the beginning and the end of the stroke are not identical: quantity of physical factors come into play: type and quantity of material (oil, acrylic, watercolor, etc., brush almost dry or overloaded...) length and resistance of the bristles and so on. All these factors are simulated by different methods in drawing applications. What I am trying to do is not to faithfully reproduce material characteristics in the pixel world but to discover new things (for me anyway), a visual language that satisfies me more.

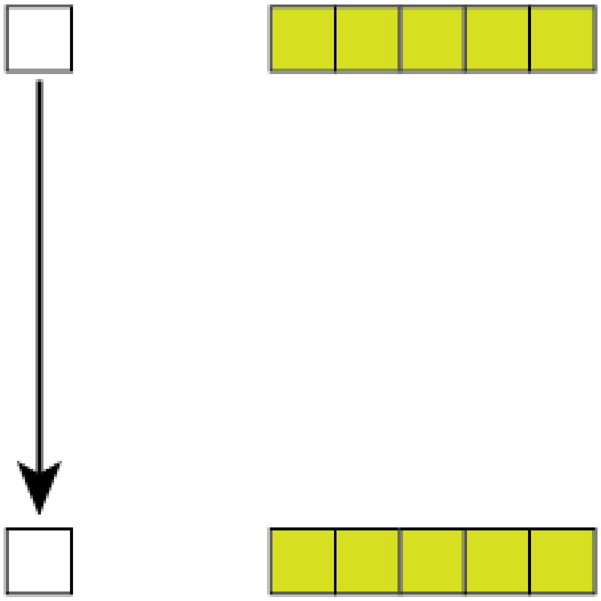
I felt that, by building the application, it would push me to observe in greater details existing images and get a more acute representation in my head of things I’d like to draw. It is becoming a quest for a deeper, richer and more intimate language. In fact, I think it could be a good idea to share my observations in this blog.

At its simplest, a digital brush is a virtual set of pixels that form an image:

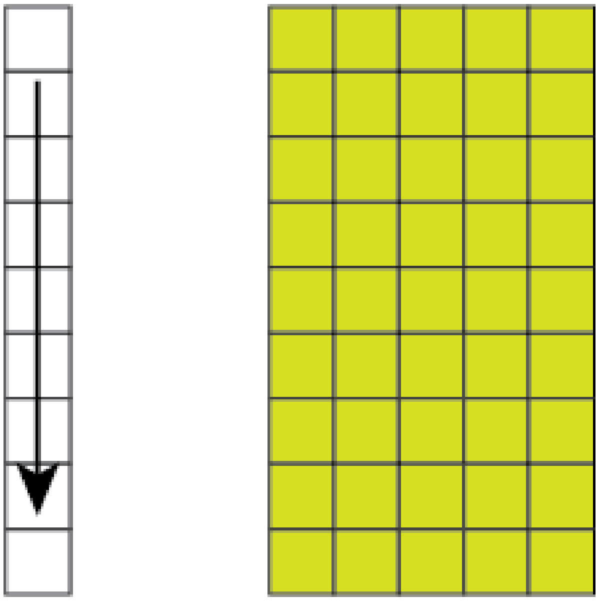


This image will be "plated" on the drawing surface. Repeated several times, it will form a line.

The line drawing is determined by at least two pixels:



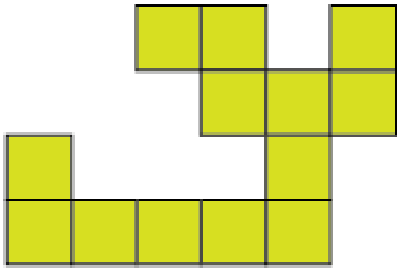
If the distance between the two is too great, calculations can determine all the points to be drawn:



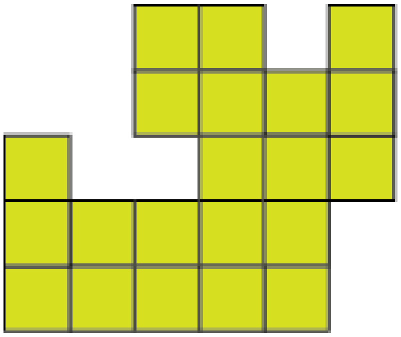
It's very nice and very basic but, already, it takes a lot of code to work because you have to tell the computer everything, including the tiniest detail, for it to perform a task.

But, already, from this basic work one can imagine variations. One can think of moving brush pixels and varying the color. Here again, the rules of the cellular automata which dictate behaviors according to environment variables (surrounding pixels, colors of the surface to be repainted, distance, speed, etc.) may be made to effective use.

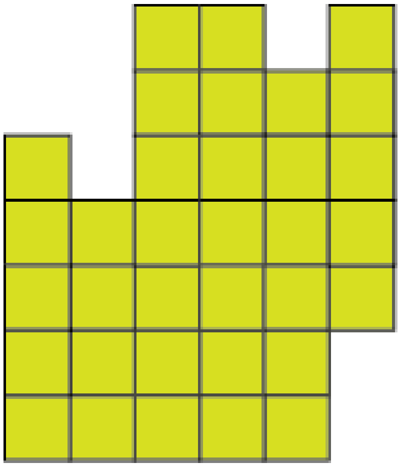
In the case of more complex brush images, the complexity disappears as the image is stamped to follow a path:



2 almost overlapping stamps:

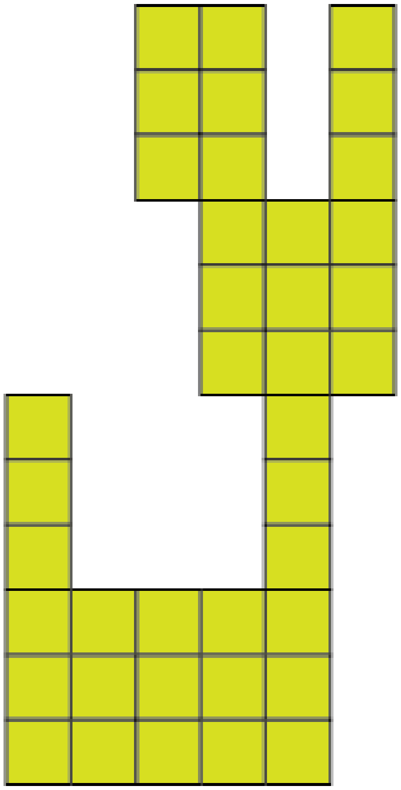


5 stamps:



It may be preferable to stretch the image of the brush. The stretching is done gradually, which means that each time the brush “moves on the surface” by one pixel on the screen it is necessary to redraw everything from the beginning of the line, which obviously causes work overloads. The effect is however a little more realistic.

Brush shape stretched by a factor of 3:



In the cases presented here, it only gives something akin to a sloppy pattern, without much soul or vitality. I am looking for something more complex, say organic, without being confined to a reproduction process of actual brush work.

To do this, I must test a lot of variables, variations, values and avoid being constrained by the rest of the application. The most successful combinations can be integrated into Paintual as functionalities or attributes.

En parallèle, je construis un système de collecte, stockage et classification d’images. Je tiens à ce que mon programme puisse tout seul trouver des images similaires à ce que je suis en train de dessiner et les compléter automatiquement.

Donc du code, du code, du code et pas beaucoup de dessins à montrer pour le moment.

In parallel, I build a system for collecting, storing and classify images. I want my program to be able to find images that are like what I'm drawing and automatically complete them.

So it’s code, code, code time and not much new drawings to show for now.