Digital Air Brush – a 10 DOF bimanual tool

Paulo Pacheco[2,1] **George Fitzmaurice**[1,2] **Ian Ameline**[1] **William Buxton**[1,2]

[1]Alias|wavefront
Toronto, Ontario
Canada

[2]Department of Computer Science
University of Toronto
Toronto, Ontario, Canada

ABSTRACT

There is an ongoing tension in user interface design between using new tools in new ways, which requires the acquisition of new skills, *vs* tailoring computer-based tools around the way things are done in traditional media, thereby exploiting existing skills. The former might be called *innovation* and the latter *emulation.* We explore the relationship between the two within the context of a computer paint program, in particular, an airbrush tool. In the process, we also explore human interaction with computers using a high (10) degrees of freedom (DOF) input device, the Wacom *Intuos* tablet. The investigation sheds new light on not only future paint programs, but graphical interaction in general.

Keywords

Bimanual, two-handed, DOF, airbrush, paintbrush, stencil, cursors, feedback.

1. Introduction

The use of computers has had a strong impact on how we think about problems and perform specific tasks. In many, if not most cases, there is little relationship between how work was traditionally performed, and how it is done with computers. Modeling 3D forms with clay is fundamentally different than doing so with 3D computer graphics. Few, if any, of the skills of the clay sculptor transfer easily to sculpting with a computer. The (generally tacit) assumption has been that the price paid in discarding old skills and learning new ones was made up for in other benefits accruing from the new tools.

In some areas, computer paint programs, for example, the computer techniques have tried to follow much more closely the traditional techniques. At the basic level, at least, if you could draw and paint with traditional media, your could rapidly capitalize on these skills using the computer, and thereby get the best of both media, while paying a very low price in terms of new skills required.

The problem with this latter, *emulation* approach, is that at a certain point (generally sooner than later) the correspondence between the traditional and the new break down. A consequence is that while expectations generated by the familiar pointed one way, the reality of the computer tool went in another. A good example of this is with the air brush. While the computer version is superficially similar to the real thing, this breaks down almost immediately upon probing even slightly beyond the surface.

In what follows, we discuss our practice and experience in developing a prototype paint program, focussing on the air brush tool. Our purposes are manifold. First, we want to see how far we can go in terms of maximising skill transfer from the traditional media. In order to do so, we need to push well beyond the number of degrees of freedom typically used in computer air brush tools. In this process, we purposely constrained ourselves to doing so using widely available technology rather than exotic "one off" devices.

What we found is that this deep and narrow study gives rise to some interesting insights that have, in our opinion, significant implications to the design of future paint systems, specifically, and computer graphics applications, in general.

2. Traditional Air Brush

A traditional airbrush has multiple degrees of freedom. The size of the ink pattern is largely determined by the distance of the brush from the surface being painted. The shape of the pattern is a conic section determined by the angle of tilt of the airbrush, and the orientation of the conic section is determined by the direction of the tilt. The amount of ink that is emitted per unit time is controlled by a spring loaded trigger, mounted on the side of the device.

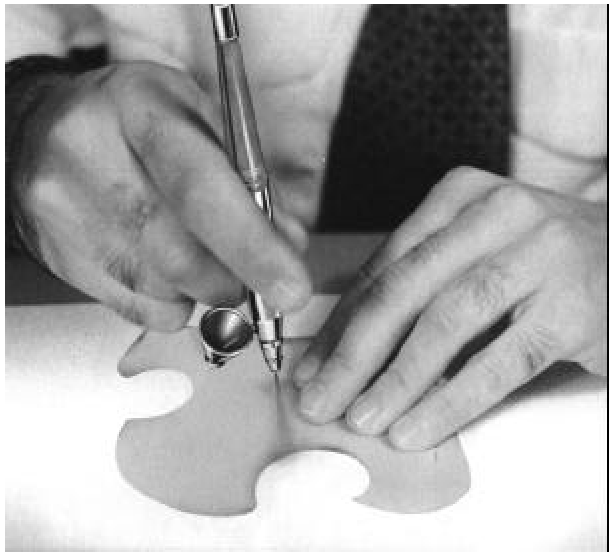

Figure 1: A traditional airbrush drawing process.

What we have described thus far is the airbrush itself. This is something that can be (but isn't) emulated reasonably well with existing technology. (More on that later.)

If we move from a focus on the tool to the work, intent, and method of use, we see that even if we copied a traditional airbrush perfectly with a computer, the tool would still be inadequate in comparison with the real thing. The reason for this can be gained by anyone who carefully reads any book on airbrush technique, or simply watches a professional artist at work. What quickly should become clear is that airbrushes are almost never used alone. In almost all cases, they are used in combination with a "frisket" or stencil, which is typically held in the other hand. In fact, the key feature of the vocabulary of an airbrush is a sharp edge on one side, made possible by the frisket, and a soft feathered edge on the other, non-masked side.

Figure 1 shows a traditional airbrush artist also using a piece of cardboard cut to a desired shape as a mask.

3. Conventional Digital Airbrushes

Digital paint programs have a fairly long history, dating from the late '70s and early '80s. [12]. In many ways they grew in sophistication to the point where they have become standard tools of the graphic artist. However, even today, there is little resemblance between the richness of the digital tool compared to the traditional one, when we look at airbrushes.

In fairness, until recently, this was partly due to the hardware technology available. Affordable generally available pressure sensitive styli are less than 10 years old. Early styli were tethered, affecting usability, and the graphics displays were not capable of handling the large brushes that were associated with airbrush art.

Hence, the fact that digital airbrushes did not much resemble real ones is understandable. Typically they used stylus pressure to control the size and/or the amount of ink, and the ink pattern was some gaussian type distribution. Examples of this generation of digital airbrush are Corel *PhotoPaint* [14] and Adobe *Photoshop* [13].

More recently, tablets have become available that support styli with pressure sensitive tips, wheels on the side that emulate the trigger on a conventional airbrush. They are also able to sense the tilt of the stylus, and even enable the user to control another widget, such as a digital frisket, by simultaneously sensing the position and orientation of a puck on the tablet [11].

Painter6 from MetaCreations [15] is the first product to attempt to come closer to mimicking traditional technique of airbrushing. They use a conic section for their airbrush, and the side trigger to control inkflow. Nevertheless, the feel is still not smooth, and the tool is still not capable of taking full advantage of the skills of a trained traditional airbrush artist. Foremost, in this regard, is the ability to dynamically change the size of the pattern, and use the airbrush in combination with a frisket or mask.

In this regard, the only product that has supported bimanual airbrushes with movable masks is Alias|Wavefront's *StudioPaint*. However, this product does not take advantage of tilt or conic sections.

None of the tools yet do the job that needs to be done. Yet, one of our arguments is that with the technology available today, this need not be the case.

While providing a brush with real-time responsiveness where that brush is sub-pixel positioned, has variable intensity, size and shape combined with a stencil mask presents a formidable challenge, it can now be done. Several tens of millions of different shapes and intensities of brush images may have to be stored. Even for cursor feedback, several million shapes must be computed. Providing all this matching real-time requirements, was not an easy task for a personal computer even as recently as some months ago. [Note: This could be accomplished on an SGI Reality Engine as far back as 1993] We are just entering a very particular moment that allows us to put everything together in a reasonably priced package.

4. A Hybrid Digital Airbrush Tool

In our approach (Figure 2), we propose a combination of two tools: an airbrush and a paintbrush. The idea is to capture as closely as possible the skills and technique of a traditional airbrush, and at the same time, integrate this with other functionality commonly used in combination with it, but not practical to integrate with traditional media.

Without pressure on the tip, we have an airbrush tool, with tilt for the shape, distance being controlled by the puck wheel (size), and ink by the stylus wheel. The airbrush is in effect only when the stylus is used off of the tablet surface, as in the case of traditional media.

However, when the stylus comes into contact with the tablet, its functionality switches to a pressure sensitive "marker" tool, rather than an airbrush. With this, size is controlled by the pressure. Table 1 summarises the functionality of the tool.

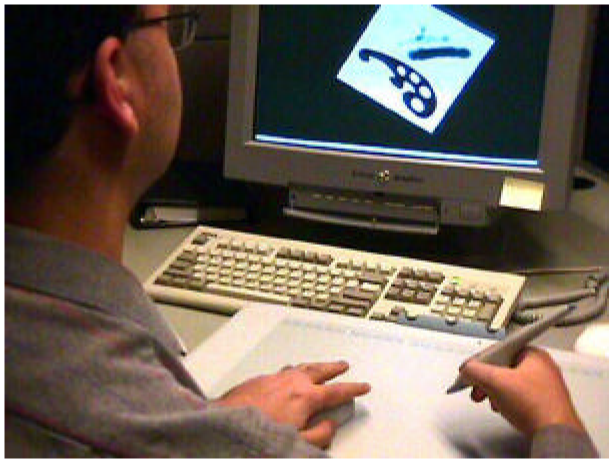

Figure 2: Our proposal. Note that the Stylus is above the Tablet.

Tool	Ink	Size	Shape
Real	Trigger	Distance	Tilt
Traditional	Tip	Tip	Tilt
Mimic	Stylus Wheel	Tip	Tilt
New (Tip = 0)	Stylus Wheel	Puck Wheel	Tilt
New (Tip > 0)	Stylus Wheel	Tip	Tilt

Table 1: Comparison among several airbrush tool approaches, our purpose and the real airbrush

Table 2 shows the mapping of the interchangeable analogue input attributes with the tool.

Tip State	Attribute	Puck Wheel	Stylus Wheel	Stylus Tip	Puck Stylus Offset	Stylus Button
P = 0	Size	✓		5		
	Ink		✓	5		
	Zoom	✓		5		✓
P > 0	Size			✓		
	Ink		✓			
	Zoom	✓				

Table 2: Analogue inputs mapping

5. Degrees of Freedom

Several works on evaluation of human-computer interaction are available. Since a deeper analysis would be beyond the scope of this paper, we summarize some interesting results obtained by Shumin Zhai [4][p101]:

“The physical property of a 6 DOF input device should provide rich proprioceptive feedback so that the user can easily feel her control actions so as to learn the task quickly.

The controller transfer function used in any interaction technique should be compatible with the characteristics of the physical device.

Fine, small muscle groups and joints (i. e. fingers) should be included in the operation of input devices when possible.

Visual display of user actions in relation to target objects should be designed to allow immediate extereoceptive feedback, and the inclusion of semi-transparency well serves this purpose by revealing the depth relationship between a cursor and target objects.”

DOF	Device	Action / Control
2	Puck Translation (x / y)	Translates Mask / Paper
1	Puck Rotation	Rotates Mask / Paper
0	Any Puck Button	Toggles Mask / Paper action
1	Wheel	Zoom
1	Pressure	Airbrush distance (shape size)
1	Wheel	Ink flow
2	Tilt (x / y)	Airbrush tilt
2	Stylus Translation (x / y)	Airbrush x / y position

Table 3: DOF implemented in our application

In this work, we are extending these concepts to a 10-DOF system (Table 3). As a matter of fact, we are still far from the at least 56-DOF the human body has (considering only bones movements).

The limitation, however, is the connection human - machine, keeping ALL of those DOF’s in a conscience level at the same time.

6. User Interface

In the traditional digital airbrush tool, we are only concerned with the position and pressure. In contrast, this tool offers the tilt and the ink flow control. In addition, an actual piece of paper is simulated providing both rotation and translation, as in the real world. A Stencil Mask completes the metaphor.

The graphical interface chosen is made up of the following components:

- Sheet of Paper
- Stencil Mask
- Variable Cursor.

The input control devices are:

- Stylus Pen (Figure 3)
- Six-button Puck (Figure 4)
- Wacom tablet.

Figure 7 shows the basic element output.

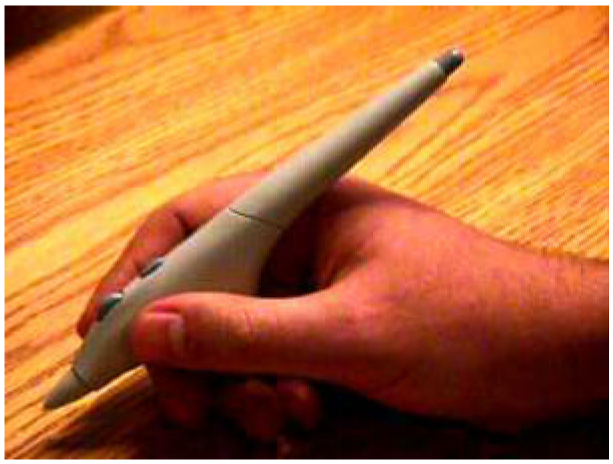

Figure 3: The Stylus Pen and its various elements: the eraser (on top), the tip, on bottom and the wheel, close to the index finger.

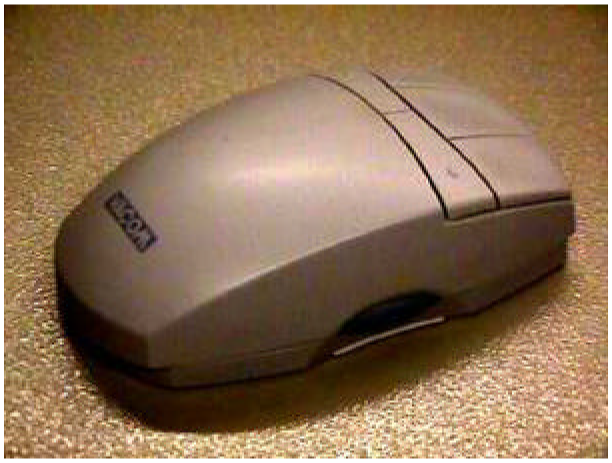

Figure 4: The Puck - The puck wheel appears in foreground, close to the table where it lies.

The Paper is the region where the user draws. It has the translation and rotation properties, which are controlled by the Puck when any of its buttons is pressed.

The Stencil Mask can be any bitmap. As a consequence, it can be any irregular shape, concave or convex, continuous or not. It is the default object controlled by the Puck, when no button is pressed. The puck wheel controls the amount of zoom for both Paper and Stencil Mask.

The Variable Cursor is an important element in the application, since it is responsible for providing the feedback for the dominant-hand (DH). It is basically an ellipse with variable shape and position, which is modeled according to the Stylus Pen position, tilt and pressure.

7. Algorithms

To implement the various elements mentioned in the previous item, we had to deal with several optimizations.

7.1 Paper

For the implementation of the real-time translating and rotating paper, two coordinate systems were used.

One fixed in the origin, without rotation or translation, being used only in back buffer, to record information in memory, only. This information is used only when the paper is either rotated or translated.

The other coordinate system is aligned to the actual paper the user sees on the screen. It is used basically to avoid slow memory operations, using as much as possible hardware capabilities, such as the OpenGL display lists.

7.2 Mask

Since the mask is basically stored in a display list, not being modified during the entire process, it is the fastest component to be displayed. For the stencil operations, only OpenGL blending is being used, instead of its slower stencil buffer.

7.3 Strokes and Variable Cursors

The elliptical strokes for the brushes used for the output are stored previously in OpenGL display lists, being calculated as it is shown in the next section. The same occurs for the cursors, but only the shape is calculated, without ink issues.

It is also interesting to mention that in order not to have a flickering cursor, some copy and draw pixel operations are performed in the front and back buffer.

Some mathematical derivations are presented in Appendix 1.

8. Conclusions

Despite their long history (in computer terms), digital airbrushes have not yet reached their potential. In order to do so, and in so doing deliver artists tools worthy of their skill, we have to move beyond the individual tools and focus on workflow and skill, not technology. The underlying technology is currently there to deliver worthy tools. What we as computer professionals need to do is apply the same concern with design to our products, as the artists that we ask to use them apply to their output.

9. Future Work

There are some obvious and practical extensions of this work. First, we would like to increase blending to at least 12 bits per channel to offer an even softer brush edge. In general, we would also like to continue to improve the overall performance of the system. The user interface can also be enhanced in a number of ways such as adding a more sophisticated set of brush-related widgets such as a color palette and allowing the mask and menus to be transparent to reduce interference between the user and the artwork. Lastly, we would like to explore how our techniques would translate to painting in 3D.

10. ACKNOWLEDGMENTS

The authors would like to thank Azam Khan, Jade Rubick, Gordon Kurtenbach, Ravin Balakrishnan, Stephen Spenceley and Jill Jacob at Alias|wavefront for their comments and assistance to this work. Lastly, we would like to thank Wacom for their encouragements and supplying us with Intuos tablets, pucks and air brushes.

11. REFERENCES

[1] Moore, D. S. and McCabe, G. P. Introduction to the Practice of Statistics, Freeman, 1993.

[2] Foley, J. D., van Dam, A., Feiner, S. K., Hughes, J. F. Computer Graphics: Principles and Practice. Addison Wesley, Reading, MA. 1990.

[3] Porter, T. and Duff, T. Compositing Digital Images. Proceedings of SIGGRAPH 84, volume 18, pages 253-259, July 1984.

[4] Zhai, S. Human Performance In Six Degree Of Freedom Input Control. Ph.D. Thesis. University of Toronto. Department of Industrial Engineering. 1995

[5] Wright, R S. and Sweet, M. OpenGL Superbible. Waite Group Press. 1996

[6] Silicon Graphics. OpenGL Reference Manual. Addison Wesley. 1992.

[7] Neider, J., Davis T., Woo M., OpenGL Programming Guide. Addison Wesley. 1993.

[8] Kurtenbach G., Fitzmaurice, G., Baudel, T., Buxton, B. The design and evaluation of a GUI Paradigm based on Tablets, Two-hands and Transparency.

[9] Smith, A.R.. Paint, Tutorial: Computer Graphics, pp. 501-515 (1982). IEEE Computer Society. Edited by K. S. Booth.

[10] Maurello, R. S. The Complete Airbrush Book. Wm. Penn Publishing, New York. 1955.

[11] Wacom: http://www.wacom.com

[12] Smith, A.R. (1982). Paint. In K. S. Booth (Ed.). *Tutorial: Computer Graphics*. IEEE Computer Society, 501-515.

[13] Adobe Photoshop: http://www.adobe.com/products/photoshop/main.html

[14] Corel Photopaint: http://www.corel.com/paint9/index.htm

[15] MetaCreations Painter6: http://www.metacreations.com/products/painter6/

[16] Alias|Wavefront StudioPaint: http://www.aw.sgi.com

Appendix 1: Stroke derivations

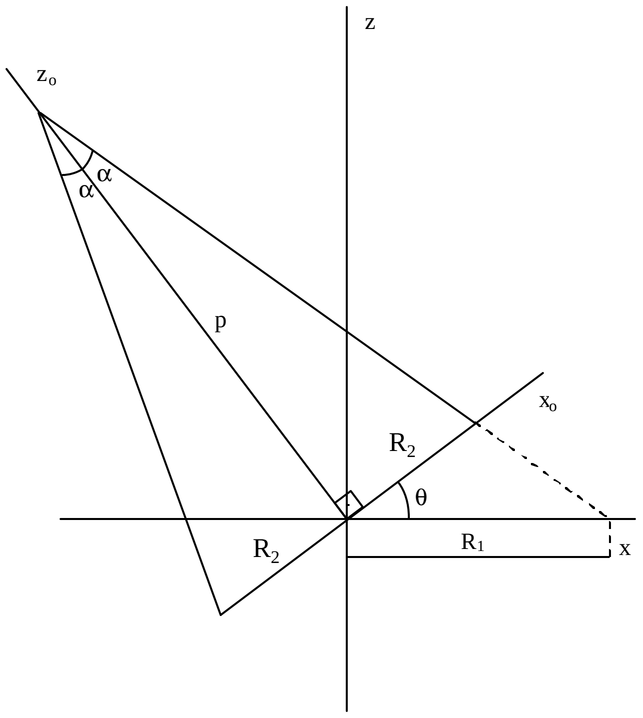

Figure 5: Schematic cone, intercepting the plane z=0

In this section we present some mathematical derivations, part of which was used to implement the brush shapes and feedback cursor.

A.1. Geometry

Let a cone (Figure 5),

$$x_0^2 + y_0^2 = [(p - z_0)\tan\alpha]^2 \tag{1}$$

where α is the cone angle, and *p* is its axle length

and a plane,

$$z_0 = 0$$

in the (x_0, y_0, z_0) coordinate system.

Rotating the plane clockwise by θ degrees and considering a new coordinate system (*x, y, z*), so that the plane equation be

$$z = 0 \tag{2}$$

we have

$$\begin{cases} x = x_0\cos\theta - z_0\sin\theta \\ y = y_0 \\ z = x_0\,\mathrm{sen}\,\theta + z_0\cos\theta \end{cases}$$

and thus,

$$\begin{cases} x_0 = x\cos\theta + z\sin\theta \\ y_0 = y \\ z_0 = -x\sin\theta + z\cos\theta \end{cases} \tag{3}$$

Substituting (2) and (3) in (1),

$$(x\cos\theta + z\sin\theta)^2 + y^2 = [(p + x\sin\theta - z\cos\theta)\tan\alpha]^2 \xrightarrow{z=0}$$

$$x^2(\cos^2\theta - \sin^2\theta\tan^2\alpha) - 2px\sin\theta\tan^2\alpha + y^2 = p^2\tan^2\alpha$$

Analyzing the cases of θ + α, we have three cases:

1) $\theta + \alpha = 90^\circ\,(Parabola)$

$\theta = 90^\circ - \alpha$, then

$$y^2 = 2p\sin\alpha\tan\alpha\left(x + \frac{p}{2\cos\alpha}\right)$$

2) $\theta + \alpha < 90^\circ\,(Ellipse)$

$$\frac{\left(x - \dfrac{p\sin\theta\tan^2\alpha}{\cos^2\theta - \sin^2\theta\tan^2\alpha}\right)^2}{\dfrac{p^2\cos^2\theta\tan^2\alpha}{(\cos^2\theta - \sin^2\theta\tan^2\alpha)^2}} + \frac{y^2}{\dfrac{p^2\cos^2\theta\tan^2\alpha}{\cos^2\theta - \sin^2\theta\tan^2\alpha}} = 1$$

Which is an equation of an ellipse, where

$$R_1 = \frac{p\cos\theta\tan\alpha}{\cos^2\theta - \sin^2\theta\tan^2\alpha} = \frac{p\cos\sec\theta\tan\alpha}{1-\tan^2\theta\tan^2\alpha}$$

$$R_2 = \frac{p\cos\theta\tan\alpha}{\sqrt{\cos^2\theta - \sin^2\theta\tan^2\alpha}} = \frac{p\tan\alpha}{\sqrt{1-\tan^2\theta\tan^2\alpha}}$$

$$X_M = \frac{p\sin\theta\tan^2\alpha}{\cos^2\theta - \sin^2\theta\tan^2\alpha} = \frac{p\tan\theta\cos\sec\theta\tan\alpha}{1-\tan^2\theta\tan^2\alpha}$$

3) $\theta + \alpha > 90^\circ \left(Hyperbola\right)$

$$\frac{\left(x + \frac{p\sin\theta\tan^2\alpha}{\sin^2\theta\tan^2\alpha - \cos^2\theta}\right)^2}{\frac{p^2\cos^2\theta\tan^2\alpha}{\left(\sin^2\theta\tan^2\alpha - \cos^2\theta\right)^2}} - \frac{y^2}{\frac{p^2\cos^2\theta\tan^2\alpha}{\sin^2\theta\tan^2\alpha - \cos^2\theta}} = 1$$

In this implementation, we considered only the second case, leaving the other cases to be considered for future implementations.

A.1.2. Ink

Since we have already our boundaries limited to a known ellipse shape, now we are able to fill the ellipse. To do this, lets consider Figure 6, through which we will derive the amount of ink for each pixel, taking into account two things:

- The intensity decrements in a manner inversely proportional to the square of the distance from the vertex V of the cone, from where the ink is emitted, in our model, and
- The variation of the ink intensity according to the angle β the pixel has in relation to the axle of the cone, which will be modeled as a Gaussian curve.

The distance d from a generic point inside the ellipse curve is:

$$d^2 = \left(p\sin\theta + x_p\right)^2 + \left(p\cos\theta\right)^2 + {y_p}^2$$

And the angle β can be calculated as follows:

$$x_p^2 + y_p^2 = p^2 + d^2 - 2p \cdot d \cdot \cos\beta$$

$$\therefore \cos\beta = \frac{p^2 + d^2 - \left(x_p^2 + y_p^2\right)}{2p \cdot d} \qquad (4)$$

We can, then, consider a Gaussian ink distribution, using the following equation, using the β angle calculated in (4).

$$f(\beta) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{\beta}{\theta}\right)^2} \qquad (5)$$

Then, the final value for the amount of ink, can be given by the following expression:

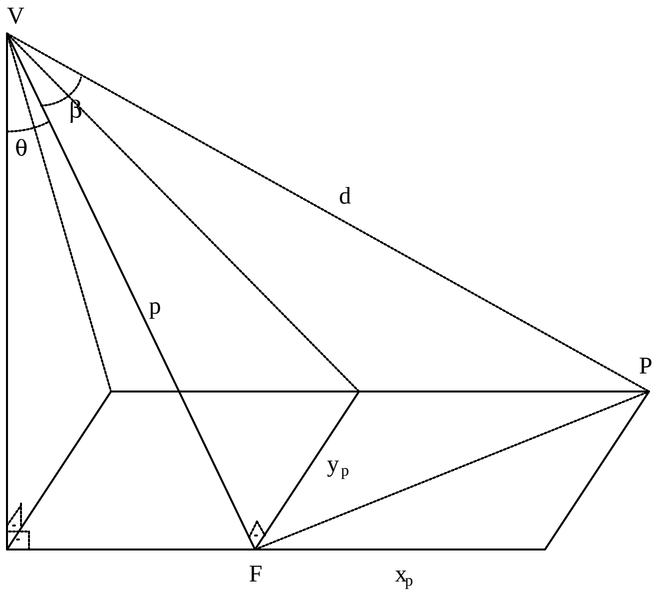

Figure 6: Auxiliary scheme to calculate the pixel intensity where F = ellipse focus, P = pixel, d = distance between the Pixel P and the cone vertex V.

$$ink = K_0 \frac{f(\beta)}{d^2},$$

where K_0 is a constant and $f(\beta)$ is calculated in (5).

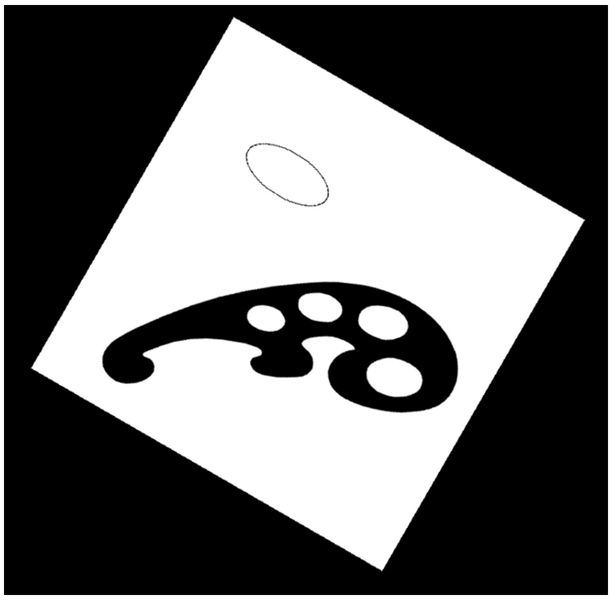

Figure 7: The basic elements - Paper, Mask and Feedback Cursor

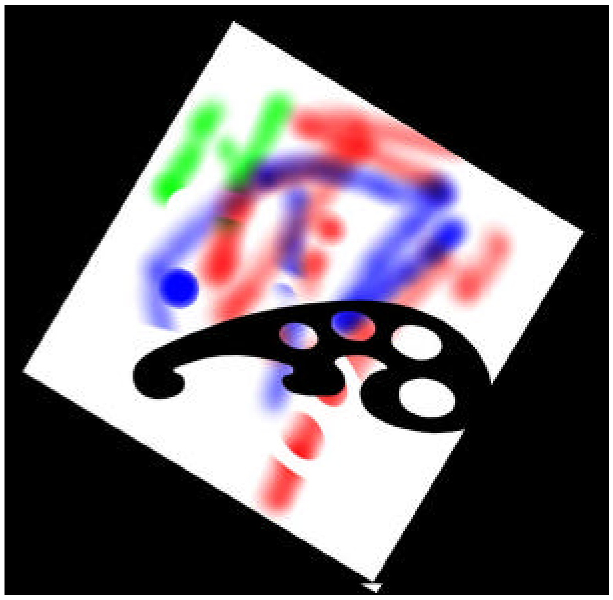

Figure 8: Some colour drawing

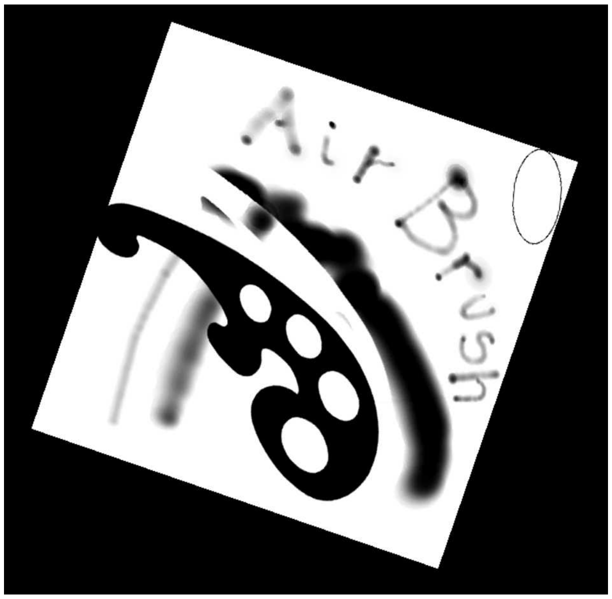

Figure 9: Some random drawing exploring our application capabilities

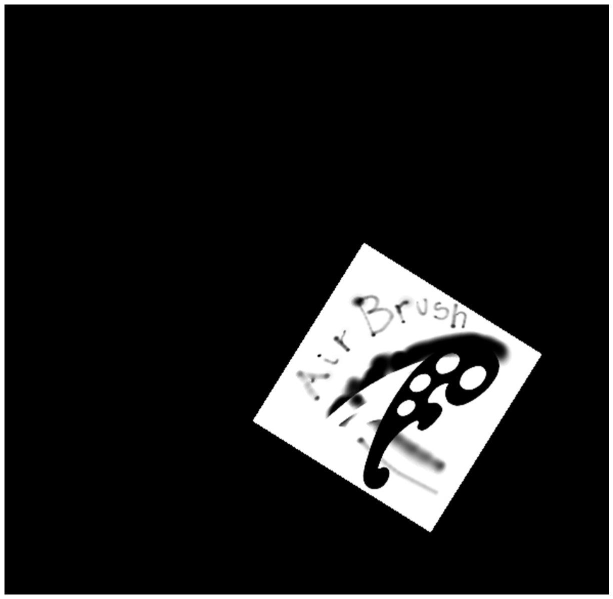

Figure 10: The same drawing as in Figure 9, with the paper zoomed, rotated and translated.

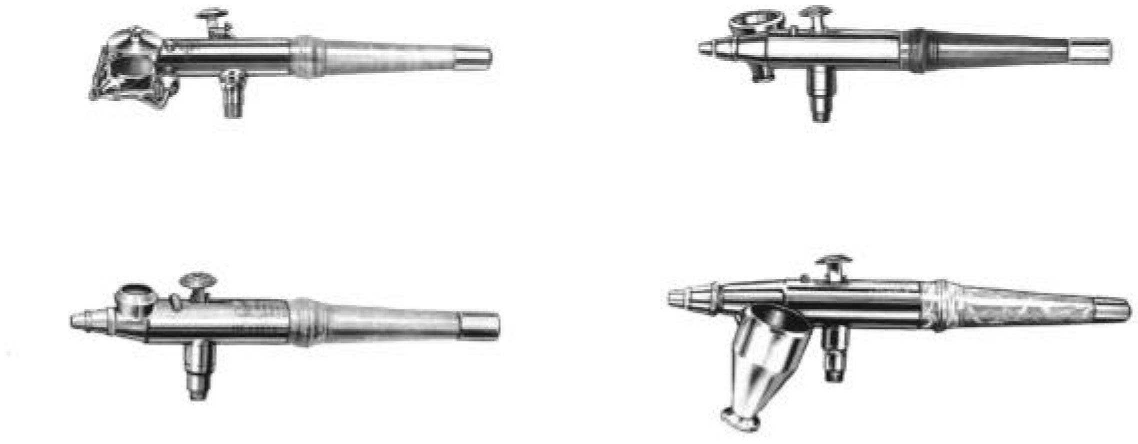

Figure 11: Some traditional (real) airbrush tools