

Personal Research Project: Processing Images for an Impressionistic Effect with Imaging Lingo & Sharp Image Export Xtra

Ching Kei Jacqueline Tong

Interactive Multimedia Program in Sheridan College, 2004

Introduction

The primary topic for the research project is Imaging Lingo and the secondary topic is the Sharp Image Export Xtra. Imaging Lingo allows developers to transform a given image to other interesting images. Some of these effects produced are similar to using a filter that is provided by popular image editing software such as Adobe Photoshop. However, there are usually a lot of limits with commercial filters. They might not be able to achieve the effect you are looking for. With Imaging Lingo, you can custom made you own filter for the effect that you desire. This research project also uses the Sharp Image Export Xtra. This Xtra allows user to save images that are created by Imaging Lingo as JPEG, PNG, and BMP files.



In the seventeenth century, Claude Monet created paintings that attempted to catch the impression of sunlight on objects. He achieved that effect by applying unmixed primary colors and small strokes on the canvas to simulate actual reflected light. Impressionist paintings provide the inspiration for the personal research project. Through this project, I can integrate my programming skills and my artistic talent to create digital art work, and I have also gain a better understand of how filters are implemented in popular commercial imaging editing software.

Technology

What is Imaging Lingo? Basically, it refers to functions that allow you to directly interact with and edit image data in cast members, the stage, and image objects. You can think of Imaging Lingo as Lingo that allows you to create imaging edition applications like Photoshop. It allows you to edit photos, create a drawing program, or even perform interesting effects on sprites and cast members. The following is some examples of applications that can be created using Imaging Lingo.

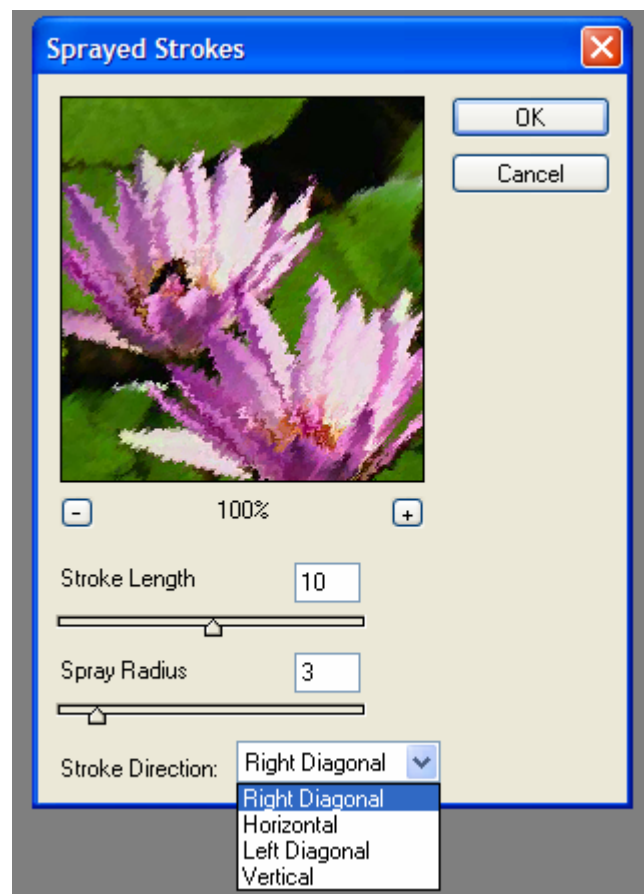
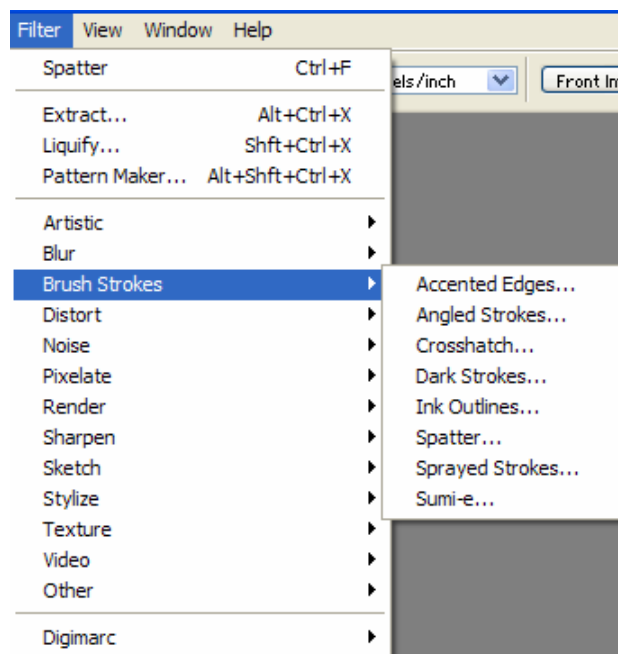
- Real-time jigsaw puzzle engine
- Create a paint program
- Create filters for images, for example, ripple effects on images, real-time motion blur
- Create "morphs" between pictures
- Perform color correction on images
- Add video to 3D texture maps in Shockwave 3D
- Create multi-tiered pop-up menus
- Create custom transitions between frames
- Create tile based game engines
- Generate real time graph using parameters given by users
- Real time game board creation

There are limitations to Imaging Lingo. For example, once you have drawn shapes on the bitmap image, the RGB value of the pixels in the image will be altered. Therefore, you can no longer transform the shapes using sprite properties such as locH, locV and rotation. It is because the shape itself is not actually a sprite. Another limitation to Imaging Lingo is that it does not provide you with functions to save the images into files. After you have created some amazing graphics, for sure you want to save them for future references. This problem leads to the second topic for the research project – the Sharp Image Export Xtra.

The Sharp Image Export Xtra is a free Xtra developed by Sharp Software. You can download the Xtra and read the documentation on the Xtra in <http://www.sharp-software.com>. This Xtra is supported by both the windows platform and Macintosh. It allows developer to save bitmap castmembers as JPEG, PNG, and BMP/PICT files. Please note that BMP files are only supported on the Windows platform and PICT files are only supported on Macintosh. There are limitations to the Sharp Image Export Xtra. For example, it does not support file extensions such as gif. It is also not Shockwave safe and it is only compatible with Director 7 and upwards.

Research Application

With the sprayed strokes filter in Adobe Photoshop, you can generate a brush strokes effect. However, the application only allows you to change the stroke length, spray radius, and some degree of stroke direction. If you want to increase the width of the stroke, to change the density of the strokes, or to specify the exact orientation of the stroke, you would be out of luck.



In order to address the deficiency of these filters, this research project focuses on creating an imaging tool that allows its users to load an image to the application, then transform the image by adjusting various parameters, such as, stroke orientation, stroke orientation variation, spacing, length of the stroke, and width of the stroke.

In order to create this application, we need to understand the general algorithm for creating an impressionist effect. This algorithm is inspired by the definition of an impressionistic painting. Impressionistic paintings can be defined as the following:

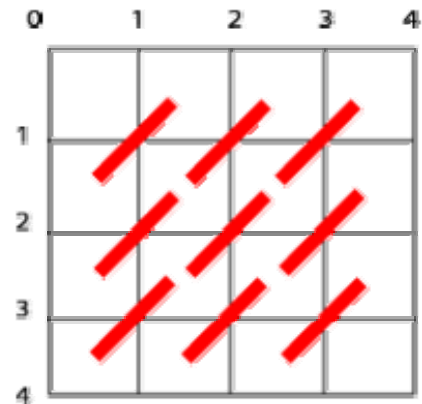
“The impressionist style of painting is characterized chiefly by concentration on the general impression produced by a scene or object and the use of unmixed primary colors and small strokes to simulate actual reflected light.”¹

According to the definition, we can sketch out a general algorithm:

```

for some pixels in the image
  find the color of the pixel from the original image
  draw a small stroke with the appropriate width and length
  center the stroke according to the location of the pixel
  rotate the stroke to the orientation specified by the user
end
  
```

Note that we are not performing the operation on all the pixels in the image because we want to have to ability to adjust the density of the brush strokes.



Task #1: Find the colour of the pixel from the original image

With Imaging Lingo, you can find the color of a specific pixel in an image easily. For example, if we want to find the colour of the pixel in location (100, 200) of the image called gPicture. We can do the following:

```
pixelColour = gPicture.getPixel(100, 200)
```

Task #2: Calculate the coordinates of the endpoints of each stroke

As mentioned by the Technology section of this paper, we cannot transform the bitmap shapes once they are being drawn into another image because they are not separate sprites. In order to solve this problem, we need to find the coordinates of the endpoints of each stroke and draw the strokes according to the coordinate of those points.

We need to use some basic trigonometry to accomplish this task. Recall the following:

$$\sin \theta = \frac{\text{opposite}}{\text{hypotenuse}}$$

$$\cos \theta = \frac{\text{adjacent}}{\text{hypotenuse}}$$

¹ Nicolas Pioch, “WebMuseum, Paris: Impressionism,” <http://www.ibiblio.org/wm/paint/glo/impressionism/>, February 2004.

Apply trigonometry to our specific problem, we get the following equations:

$$\cos \theta = \frac{x1}{\left(\frac{\text{StrokeLength}}{2}\right)} \text{----- equation (1)}$$

$$\sin \theta = \frac{y1}{\left(\frac{\text{StrokeLength}}{2}\right)} \text{----- equation (2)}$$

Rearrange equation (1) and (2), we get the formula that describes the coordinate of the endpoints of the stroke that is centered at (0, 0).

$$x1 = \cos \theta \cdot \left(\frac{\text{StrokeLength}}{2}\right)$$

$$y1 = \sin \theta \cdot \left(\frac{\text{StrokeLength}}{2}\right)$$

$$(x1, y1) = \left(\cos \theta \cdot \left(\frac{\text{StrokeLength}}{2}\right), \sin \theta \cdot \left(\frac{\text{StrokeLength}}{2}\right) \right)$$

We need a general formula where the stroke is centered in an arbitrary location (i, j).

$$(x1, y1) = \left(\cos \theta \cdot \left(\frac{\text{StrokeLength}}{2}\right) + i, \sin \theta \cdot \left(\frac{\text{StrokeLength}}{2}\right) + j \right)$$

The steps for generating the formula for (x2, y2) are trivial.

$$(x2, y2) = \left(\cos (\theta + \pi) \cdot \left(\frac{\text{StrokeLength}}{2}\right) + i, \sin (\theta + \pi) \cdot \left(\frac{\text{StrokeLength}}{2}\right) + j \right)$$

Please note that Director's trigonometry functions such as sin and cos work with gradients instead of degrees. In order to change from degrees to gradients, we have to use the following formulars:

$$\text{gradient} = \text{degree} \cdot \left(\frac{\pi}{180}\right)$$

Task #3: Draw the strokes on the image

The syntax for drawing a line in an image is as follows:

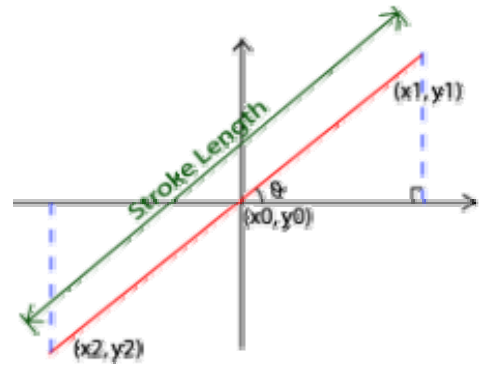
`imageObject.draw(point(x, y), point(x, y), colorObjectOrParameterList)`
 In order to draw a line with the color of 'pixelColour', stroke width of 'thickness' and endpoints (x1, y1) and (x2, y2) on an image called 'gProcessedImage', the Lingo code would look like the following:

```
gProcessedImage.draw(x1, y1, x2, y2, [#color: pixelColour,
#lineSize: thickness])
```

Task #4: Saving images

In this application, we are using the Mui Xtra to allow the user choose the name of the file that he wants to save the image into. The following is the Lingo code for creating the save image pop up window:

```
gMui = xtra("mui").new()
newFileName = FileSave(gMUI, "", "Save Image")
```



Sharp Image Export Xtra to save images to either JPG, PNG, or BMP/PICT format. First, we need to create an xtra instance.

```
xtraInst = new (xtra "SharpExport")
```

To save an image as JPG: (it is only recommended if the castmember is 8-bit or higher) Note that the first parameter is the bitmap member reference, the second parameter is the pathname to save to, and the third parameter is the jpeg compression level. The higher the level, the better the quality.

```
result = xtraInst.exportJPG (member "bitmap", "c:\filename.jpg", 70)
```

To save an image as PNG:

```
result = xtraInst.exportPNG (member "bitmap", "c:\filename.png")
```

To save an image as BMP (Windows only):

```
result = xtraInst.exportBMP (member "bitmap", "c:\filename.bmp")
```

To save an image as PICT (Macintosh only):

```
result = xtraInst.exportPICT (member "bitmap", "MacHDName:filename.pct")
```

Task #5: Putting everything together

Now, we are ready to put everything together. The source code for the application can be found on the CD that is attached to this report.

The following is examples of images that are produced by the application. Notice that you can generate different effects by adjusting the parameters, such as, spacing, stroke orientation, stroke orientation variation, stroke width, and stroke width.

Original Image



Spacing = 1
Stroke orientation = 0
Stroke orientation variation = 0
Stroke width = 1
Stroke length = 1

Sponge Style



Spacing = 2
Stroke orientation = 0
Stroke orientation variation = 360
Stroke width = 10
Stroke length = 20

Pencil Crayon Style



Spacing = 2
Stroke orientation = 45
Stroke orientation variation = 30
Stroke width = 1
Stroke length = 35

Impressionistic Style



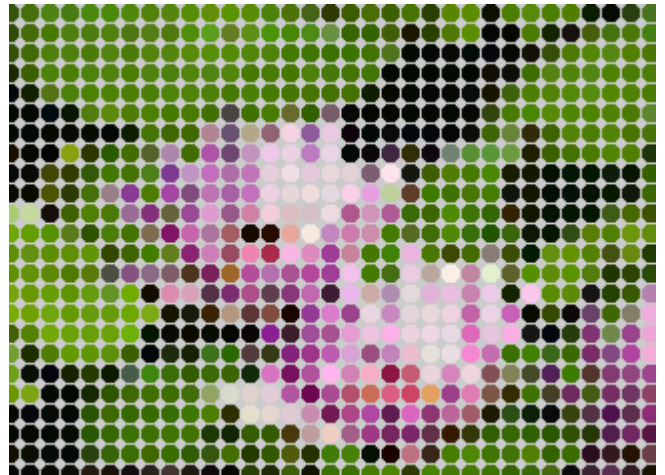
Spacing = 5
Stroke orientation = 30
Stroke orientation variation = 40
Stroke width = 4
Stroke length = 30

Fish Scale Style



Spacing = 8
Stroke orientation = 0
Stroke orientation variation = 0
Stroke width = 8
Stroke length = 35

Bubble Style



Spacing = 10
Stroke orientation = 0
Stroke orientation variation = 0
Stroke width = 10
Stroke length = 10

Blind Style



Spacing = 10
Stroke orientation = 0
Stroke orientation variation = 0
Stroke width = 8
Stroke length = 20

Block Style



Spacing = 10
Stroke orientation = 0
Stroke orientation variation = 0
Stroke width = 35
Stroke length = 20

Conclusion

This research paper explores the way that you can create a tool that allows the user to process images for an impressionist effect and other effects using Imaging Lingo. It also shows how to save images for future reference by using the Sharp Image Export Xtra. By completing the project, it provides me a better understanding of how filters are implemented in popular commercial imaging editing software.

The following functionalities can be included into future versions of this application:

- Allow user to adjust the RGB of the entire image
- Allow user to resize and crop the image to the desired size
- Resize the image automatically if the image size exceed the size of the stage
- Add randomness to the length and width of the strokes
- Allow user to select the color of the background
- Create a blur function that blurs the resulting image. It will help to create a softer image.
- Clip the strokes when it detects an edge. This will help to generate an image with more define edges

References

Litwinowicz, Peter. "Processing Images and Video for An Impressionist Effect." SIGGRAPH Proceedings 1997, pp. 151-158.

Neal, Chuck. "Director Article: Imaging Lingo Basics." http://www.macromedia.com/devnet/mx/director/articles/imaging_lingo.html. February 2004.

Pioch, Nicolas. "WebMuseum, Paris: Impressionism." <http://www.ibiblio.org/wm/paint/glo/impressionism/>. February 2004.

Rosenzweig, Gary. "Using Lingo to Create Images." Special Edition Using Macromedia Director MX. Indiana: Que Publishing, 2003.

Sharp, Werner. "Sharp Image Export Xtra." <http://www.sharp-software.com/products/index.htm>. February 2004.