Artful Abstraction

(Mini-Project – Stylized Rendering)

Omkar Saiswaroop Varma Chekuri*

University of Oklahoma

ABSTRACT

Most of the computer Graphics research has focused on photo-realistic rendering and not much on artistic rendering.

Artistic rendering can be achieved by controlling the color, shape, size of the brush strokes. In this paper I tried to replicate the ideas about abstract image representations from [Haeberli 1990].

Although I did not get the exact same results as [Haeberli 1990], I followed and implemented some of the techniques described in the paper and achieved reasonable results with artistic elements.

Keywords: Computer Graphics, Fragment Shaders, Artistic rendering.

SUMMARIES OF TECHNIQUES

1. Paint by Numbers: Abstract Image Representations

This paper was written by "Paul Haeberli" and published at proceedings of Siggraph 90. The goal of the paper was to communicate surface color, surface curvature, centre of focus, and location of edges.

Whenever the user moves the cursor over the picture in a direction, point sampling is done to calculate the resultant color at that location. By doing this the author is avoiding to calculate the overhead associated with selecting a color that would result in limited color selection for paintings.

The user selects the size, direction and shape of the cursor interactively. Whenever he clicks a mouse, the image is repainted with the resulting cursor at that location. Brush strokes of the repainting is dependent upon the speed of the cursor. The user uses geometry rather than bitmaps to facilitate scaling, rotating efficiently. The user has a way of storing the brush strokes so that he can apply the same to different images. The author also discusses advanced painting techniques such as "pushing edges" whose utility is to accentuate boundaries. The author discusses geometry using ray tracing whose utility was to display information about shadows effectively. The author also discusses Iterative relaxation techniques that helps in building interesting paintings.

2. Painterly rendering for animation

This paper was written by "Barbara J.Meier" and published in proceedings of Siggraph 1996. The main idea behind the technique is this method renders surfaces as 3D particles which are later rendered as 2D paint brush strokes in screen space. Geometry and lighting properties of the scene can be used to control brush strokes and produce variations in the painting. This method wants to use small brush strokes for farther surfaces and long brush strokes for closer surfaces. This method also uses color as a way to distinguish between two objects at their intersecting surface and curves to represent the shape of the object.

* email address: swaroopvarma.chekuri@ou.edu

The surface is first tessellated into triangles and we compute its surface area and determine the number of particles that go into it.This method generates desired number of particles using parametric surfaces each with different brush strokes.

When rendering each frame in an animation, create reference pictures and transform the particles based on the reference pictures, like distance from view point , orientation from screen space. Surface normal shaders are used to encode orientation. Animation effects on brush strokes can be achieved by animating characteristics of the reference pictures slowly and rendering the painting along with it. A fast change may result in noisy and unappealing picture so, the reference picture must be changed smoothly. After the transformation determine thee other properties of the brush stroke like color, shape , size and orientation are determined by stored information related to particles.

We can also modify the particle position by using some functions. For each particle we apply color, shape, size of the brush stroke from reference pictures.

A reference particle or object is used as a base object and randomness is applied to it within a defined range to produce an image that does not look mechanical. The various parameters that we can apply randomness are orientation, color, scale and texture. This also gives the image more hand crafted look.

The same procedure is applied to all the frames in the animation to produce the desired animation.

2. Artful abstraction: Implementation and analysis

An interactive program was developed using "HTML", "CSS" and "Javascript" using "WebGL 1.0" package.

The application is web based. The application has three canvas windows, one showing the original image to the far right, the brush strokes to the far left and the combined image to the center. Whenever the user clicks one canvas to the far left and drags the mouse on the left canvas where the user wants to draw the strokes, the path of the mouse is recorded by the canvas and an array is populated with the positions the mouse was dragged in and the canvas is rendered with the images of the brush strokes. The user can control the size of the strokes (in this case the height of each stroke by pressing the buttons provided. The direction of strokes can be controlled by the mouse. Here I only implemented line strokes. The distance between the strokes can be controlled by the speed of them mouse. If the mouse is dragged fast, the distance between the strokes is high, if the mouse is dragged slow , the distance between the strokes is less.

The user always has the option to clear the canvas and redraw the brush strokes if he thinks so. As soon as he completes drawing the brush strokes, the user needs to save the canvas as a image by right clicking on the image and selecting "Save As" option. The user needs to save the image in the same location as the script file. If he then clicks the Draw result button, A blurred version of the image shown on the right side is created

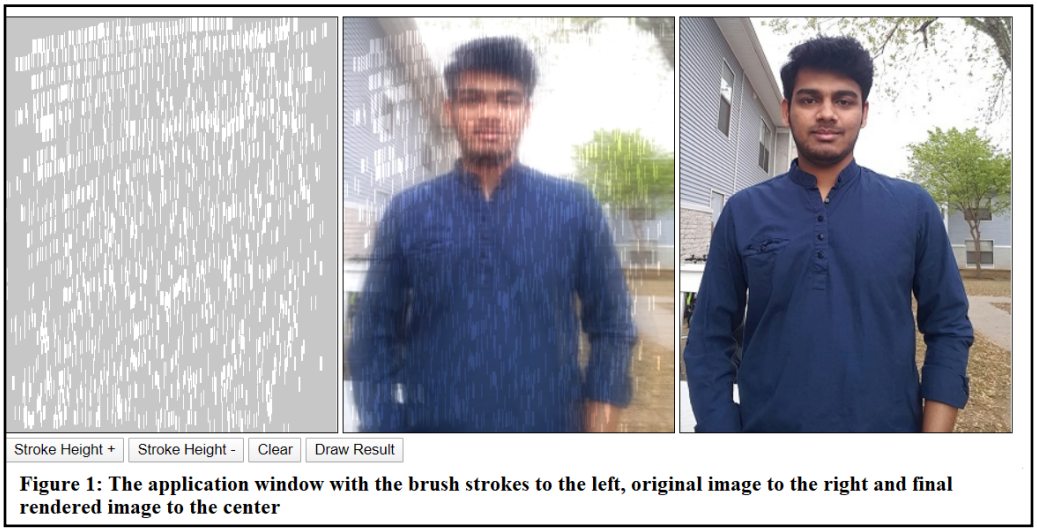

Figure 1: The application window with the brush strokes to the left, original image to the right and final rendered image to the center

in the shader . The blurred image is created to display the painting effect.

The shader then multiplies the color of each pixel of the blurred image with corresponding pixel of the image containing brush strokes producing the resulting image shown in the center. The color of the pixels is multiplied because the brush strokes are drawn with white color on a grey background. So, when we multiply both pixels the pixels that are multiplied with white color produces a different a different color than the other parts of the image.

Other variation of the implementation was using the white brush strokes on a dark grey background. If we use this variation, multiplying the person image with the brush stroke image produces grayish color on places where there was no brush strokes and uncovers the image in a slight variation on the original color where there are brush strokes giving a mush variation in the background and foreground resulting in hiding of unnecessary details.

Algorithm:

The program used two vertex shaders and two fragment shaders. As soon as the browser detects an input from the mouse the first set of shaders renders the image to the left, the second set of shaders multiplies the left image and the right image to produce the center image. Convolution is used to add the color of the neighboring pixels and adding a randomness to the pixels produces this jittery image. I did not implement any mechanism to control the distance between the strokes. The nature of mouse location sampling sampling coupled with the speed at which the user mouse is able to take care of it by producing the desired results.

By using the above method, I was able to hide most of the details and was able to produce abstract representation of the image which was my original goal. The second goal that i accomplished was by using the interactive controls, I can adjust and control the brush strokes. The current implementation was only using one shape i.e. lines. By implementing other shaper variation in the abstraction can be produced. I was not very successful in producing the colors that are representative of the artistic rendering instead i chose to use convolution and some randomness in selecting the pixel colors.

3. Utility and Applicability

3.1. Utility

Artistic rendering or Non-photorealistic rendering has been widely used in painting, drawing, technical illustrations and cartoons. This technique is used when the purpose of the visualization is not show accurately but when the designer wants to highlight some of the features that are not obvious in photorealistic rendering. It can also be used when the designer choses to abstract the content of the images and show only a part of it.

Different purposes of the images needs different abstraction levels. A person who is viewing an image as a part of cartoon is not interested in photorealistic rendering, instead he is interested in accentuating features of the original image that depicts a cartoon. An image that is displayed as a message needs a different level of abstraction. The ability to store the abstraction separately from the art that this is applied on makes this technique usable and reproducible easily. This technique is helpful in providing surface orientation, boundaries and can influence the viewers perception on the subject.

The drawbacks for this technique are there is no scientific thought process for this technique. It mainly depends on the creativity of the designer. So, the ideas of the designer are variable and the tools may not suit his needs and specifications. Though the brush strokes can be saved separately from the original file and applied to other file, this may not be optimal considering the slight differences between the images that the strokes was originally intended for.

4. Conclusion

There is a fast development in the field of artistic rendering due to its heavy use in films and media. A number of automatic tools were developed and were more accessible than ever. The availability of various categories of techniques makes it very easy for the user nowadays to influence the audience and effectively convey his ideas.

By developing this I was able to learn how to understand various techniques of abstraction that could be implemented and the abstraction levels that can be obtained by varying the technique. By developing this tool, i got better insight into how

shaders work in programmable pipeline in computer graphics and how to do shader programming.

References:

[1] [haelberli-1990a] Paul Haeberli, "Paint by Numbers: Abstract Image Representations", Proceedings of SIGGRAPH 90, pp 207-214, 1990

[2] [meier-1996a] Barbara J. Meier, "Painterly Rendering for Animation", Proceedings of SIGGRAPH 96, pp 477-484, 1996.