DIGITAL COMPOSITING WITH TRADITIONAL ARTWORK

A Thesis

by

MICHAEL STANLEY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2005

Major Subject: Visualization Sciences

DIGITAL COMPOSITING WITH TRADITIONAL ARTWORK

A Thesis

by

MICHAEL STANLEY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Ergun Akleman
Committee Members,	Donald House
	Ricardo Gutierrez-Osuna
Head of Department,	Phillip Tabb

August 2005

Major Subject: Visualization Sciences

ABSTRACT

Digital Compositing with Traditional Artwork. (August 2005)

Michael Stanley, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Ergun Akleman

This thesis presents a general method and guidelines for compositing digital characters into traditional artwork by matching a character to the perspective, lighting, style, and complexity of the particular work of art. The primary goal of this integration is to make the resulting image believable, but not necessarily to create an exact match. As a result, the approach used here is not limited to a single rendering style or medium, but can be used to create a very close match for almost any artistic image. To develop and test this method and set of guidelines I created composites using a variety of styles and mediums.

TABLE OF CONTENTS

CHAPTER Page

I INTRODUCTION 1

I.1. Motivation 1
I.2. Overview 1

II RELATED WORK 3

II.1. Publications 3
II.2. Film and Animation 5

III METHODOLOGY 6

III.1. Matching the "Camera" 6
III.2. Matching the Lighting 8
III.3. Matching the Style 8
III.4. Matching the Complexity 10
III.5. Creating a Consistent Look 11
III.6. Introduction of Error 13
III.7. Nonphotorealism in Motion 14

IV IMPLEMENTATION AND RESULTS 16

IV.1. Software 17
IV.2. Shader Development 17
IV.3. Artistic Images 23
IV.4. Compositing Process 25
IV.5. Completed Images 30

V CONCLUSION AND FUTURE WORK 31

V.1. Evaluation 31
V.2. Future Work 31

REFERENCES 33

VITA 38

LIST OF FIGURES

FIGURE		Page
1	Main steps for matching artwork.	12
2	Software used in each step of the matching process.	18
3	Original shader with up to four separate colors.	20
4	Original shader combined with displacement shader.	20
5	Smooth shading with line, displacement, and occlusion.	21
6	Extended illumination with eight separate colors.	22
7	Final version with smooth shading, highlights, and shadows.	23
8	Additional examples.	23
9	Photographic still backgrounds.	24
10	Photographic animation backgrounds.	25
11	Perspective matching with limited information.	26
12	Light matching with a directional light.	27
13	Creating a color palette.	28
14	Adjusting the shadow.	29
15	Compositing process.	29
16	Composites with photography.	30

CHAPTER I

INTRODUCTION

I.1. Motivation

There has been a great deal of recent interest in developing techniques that allow computer-generated imagery to be rendered in a non-photorealistic style. In the early 1990s researchers began to present methods for imitating artistic styles with computer renders, and by the early 2000s a huge number of papers had been published on improving these methods and developing new ways of rendering in every artistic style available. However, very few of these papers address actually combining the images rendered by the computer with traditional artwork done in the style they are based on. So rather than attempting to develop a new rendering technique or improve on an old one, this paper will focus on combining computer-generated imagery with artistic images. The goal of this thesis is to provide a general method and set of guidelines that will be useful for integrating digital characters with traditional artwork in any style.

I.2. Overview

Computer graphics artists have invested considerable effort in producing rendering techniques that mimic the look of film and photographs. These techniques have advanced to the point that, if used properly with compositing, computer-generated characters and objects can be seemlessly incorporated into live-action footage, giving the impression that they were actually present when the film was shot. Naturally, the

---

The journal model is *IEEE Transactions on Visualization and Computer Graphics*.

movie industry has been the driving force behind improvement of these techniques, as they provide the director with the creative freedom needed to produce the film he or she has envisioned.

As photorealistic rendering techniques have advanced, a number of researchers have turned to non-photorealistic rendering. Technically, this can be any method of rendering that does not attempt to match the look of a photograph, but is usually inspired by the techniques artists have used in the past. As a result, researchers have developed software that gives them the ability to create images that look as if they were done with pen and ink, charcoal, watercolor, and paint, as well as images that resemble cartoons or technical illustrations.

It is interesting to note that the motivation for development of non-photorealistic rendering techniques is not the same as for photorealistic techniques. Interest in non-photorealistic rendering seems to stem from an appreciation of the techniques used by artists or a realization that it is often easier to communicate information with an illustration than a photograph. Researchers in this area focus more on developing methods for achieving a specific look than combining the rendered images with the style they are attempting to duplicate. Very little has been published to present techniques for compositing computer-generated elements into non-photorealistic scenes. That is not to say there is no work being done in this area, as there are quite a few films and animated series that combine computer graphics with traditional art and animation. It's just that the people who are actually combining the two are focused on producing a film or episode in a series, and have very little motivation to publish the techniques they are using. So this thesis will attempt to help fill in this gap between publication and practice by providing a general method and set of guidelines that will be useful for integrating digital characters with traditional artwork in any style.

CHAPTER II

RELATED WORK

II.1. Publications

Many researchers have published work in the area of non-photorealism. In 1994, Winkenback and Salesin developed a method for rendering in a pen-and-ink style in which the stroke type was automatically chosen based on the resolution of the target image [1]. In 1996, Meier presented a method for creating painterly rendering for animation in which the brush strokes stick to the model rather than the view plane, and do not change randomly from frame to frame [2]. Also in 1996, Winkenback and Salesin presented techniques for rendering parametric surfaces in pen-and-ink [3]. In 1997 Curtis et al. automatically simulated the artistic effects of watercolor to use as part of an interactive paint system, a method for automatic image "water-colorization," and as a non-photorealistic way of rendering three-dimensional scenes [4]. Wood et al. described an approach to simulating apparent camera motion based on techniques used in traditional animation [5]. In 1998 Correa et al. presented a method for applying complex textures to hand-drawn characters in cel animation in order to combine the strengths of computer-graphics with the expressiveness of traditional animation [6]. Hertzmann presented a method for creating an image with a hand-painted appearance from a photograph [7]. They simulated brush strokes of multiple sizes and provided a way of specifying a painting style to their algorithm.

In 1999 Kowalski et al. studied the work of artists and illustrators to develop a method for rendering fur, grass, and trees that suggests their complexity without explicitly rendering it [8]. In 2000 Deussen and Strothotte developed a method for automatically rendering pen-and-ink illustrations of trees in different styles and with

different levels of abstraction [9]. Hertzmann and Perlin presented a new method for painterly video processing in which they successively painted over frames of a video only in areas where the video is changing [10]. Lake et al. developed a system for emulating cartoon styles in real time [11]. Markosian et al. further advanced their art-based rendering of fur and grass using tufts and graftals [12]. Northrup and Markosian developed a method for rendering stylized silhouettes of objects [13]. Treavett and Chen presented methods for rendering volumes with a pen-and-ink look [14]. In 2001, Praun et al. presented a method for rendering hatching-strokes over arbitrary surfaces [15], and Raskar used polygon information at the hardware level to allow them to render contour lines for special features on an object [16].

In 2002, SnakeToonz was developed by Agarwala to allow children and others untrained in animation to create cartoons from video or image sequences by combining the constraints of the cartooning medium with user input [17]. Chenney et al. developed a system for automatically simulating cartoon style animation [18]. The interest in this area stemmed from a realization that traditional hand animation was often superior at conveying information due to the artist's ability to abstract motion and play to human perception. Decarlo and Santella described a computational approach to stylizing and abstracting photographs with the goal of clarifying the meaningful information and structure [19]. By recording human eye movements they created a model of human perception to aid their system in perserving and highlighting the most important parts of the image. Durand presented a discussion of the general problem of depiction and how it relates to both photorealistic and non-photorealistic images generated by a computer [20]. Freudenberg et al. developed a method for real-time non-photorealistic shading using halftoning [21]. This enabled them to create a variety of rendering styles, from light-dependant engraving to pen-and-ink style drawings. Halper et al. approached the design of non-photorealistic

images by mimicking the process an artist would use to create an image rather than mimicking a visual effect already produced [22]. Hertzmann developed a technique for simulating the physical appearance of paint strokes under lighting [23]. Jodoin et al. presented a system that learns by example how to use hatch lines to shade an image [24]. Johnston described a method of approximating lighting on two-dimensional drawings for use when compositing hand-drawn animation into live-action footage [25]. Kalnins et al. created a system that allows a designer to draw strokes directly onto a three-dimensional model [26]. Lum et al. presented a system for the interactive non-photorealistic visualization of volume data [27]. Lastly in 2002, Majumder and Gopi developed a system to simulate charcoal rendering in real time [28].

In 2003 Decarlo et al. presented a non-photorealistic rendering system that conveyed shape using contour lines [29]. They advanced this style of rendering by introducing the suggestive contour, which anticipates and extends true contour lines as they would be present as contours with a slight view change. Kalnins et al. described a way of rendering stylized silhouettes for 3D models with temporal coherence [30]. Kirsanov et al. presented algorithms for rendering simpler silhouettes for complex surfaces [31].

II.2. Film and Animation

Computer-generated imagery is being used not only in live-action footage, but also in many traditionally animated films. It is increasignly becoming more prominent not only in the feature films Disney and other animation studios produce, but is also playing a role in a number of the animated series made for television not only in the United States, but also in Japan and other countries around the world.

CHAPTER III

METHODOLOGY

When compositing for film or animation our goal is to combine the images in such a way that they do not distract the viewer from what we want them to see. When we want to add a computer-generated character to our film we carefully record camera positions and settings, take notes on the lighting, and measure the objects in the scene when the film is being shot. Then we use that information to match the scene as closely as possible in the computer. We create detailed texture information for our character, match the light direction, color, intensity, and contrast, and create a shadow for the character so that when the character is rendered it looks as if it were present when the scene was shot. We do all of this because we want the audience to believe that the character is actually part of the scene. We want them to perceive the scene as real, as characters and objects interacting in an environment, rather than as the collection of composited images that it really is.

Our approach to compositing computer-generated elements into non-photorealistic scenes should be much the same as it is for photorealistic scenes. Our goal is still to composite the images in a way that they will not distract the viewer from the experience of the film. To achieve this we will need to match the "camera" and lighting, and the style as well as the complexity of the scene.

III.1. Matching the "Camera"

The term "camera" as used here is not quite the same as the camera used in shooting film. There we have an actual camera for recording the live-action footage and the concept of a camera used in the computer when matching the real one. The camera

used in the computer attempts to duplicate the settings of an actual camera so that it is easier to match settings such as aperature, shutter speed, and focal length, as well as camera movement. When working on footage from a real camera we can use the camera settings and positions recorded while shooting to give us a starting point for matching the representation of the camera in the computer. When working on a non-photorealistic scene, however, all we have to go on is what the artist has given us. The framing, layout, and perspective of the image were all determined by the artist who created it. In essense, the artist's eye is the "camera" that needs to be matched, and we only have the information present in the image to help us match it.

Since the scene we have to work with was created by an artist, matching it should begin with determining how the artist created it. If the artist has been kind enough to give us a horizon line or a couple of straight lines to help us determine a vanishing point, we can use that information to match the perspective in the scene. We can't always count on these of course, as it is certainly possible that the horizon line is not visible and there is no definitive way to determine where the vanishing points are. It is even possible that the artist has made a mistake, intentional or otherwise, and the perspective is not quite as realistic as we might expect. These factors make it difficult to come up with a single way of matching the "camera" that will work in all cases. Fortunately, the audience has the same information as we do when it comes to figuring out the perspective in the scene. So in the end, what is really important is that the computer-generated elements give the impression that they are moving through the scene. Our goal is to trick the viewer's eye into believing that a character is actually in the scene rather than composited over it.

III.2. Matching the Lighting

Lighting for film and photography is determined by the light sources used when shooting the film, and the surfaces available to scatter the light. Typically the lighting is carefully controlled to get the look that the director wants for each scene, so it is not too difficult to record the positions, colors, and relative intensities for use as a starting point when matching the lights in the computer. Additionally, light in the real world obeys known laws that can be approximated and programmed into the rendering software, making it easier to automate the light matching process. The lighting in a non-photorealistic scene, however, is determined entirely by the artist. It can be anywhere from extremely realistic to disturbingly unrealistic, depending on the artist's skill and intent when creating the image.

To match the lighting in a non-photorealistic scene, once again we must examine how the artist created the image. It is likely that even in a scene with unrealistic lighting the artist has used some rules to ground the image in reality so the viewer will be able to recognize what it is. The artist has an idea of how the light should reveal the form of the objects in the scene, and uses that idea to determine the color he or she uses for a particular part of the image. It is this ability to choose a color based on the idea of lighting that we must match for non-photorealistic scenes.

III.3. Matching the Style

Every artist has a different style. A number of artists could render the same object from the same position with the same lighting and the same media, and every one of them would create a unique image. This is not simply because they made different choices in framing and coloring the object, but is also a result of how they saw the scene, interpreted it, and translated that interpretation to the image they created.

If we are to be successful in matching a computer-generated character with a non-photorealistic image, it is necessary to render that character so that it looks as if it was originally part of the image when the artist created it. This means that the character needs to look as if it was rendered with the same media that the artist used to create the scene, whether it was pen and ink, paint, or watercolor. In addition, to truly match the style the character needs to look as if it was rendered by the specific artist who created the scene. This begins to look very difficult, as it would require an individual and very specific matching of each non-photorealistic image. Even with software designed for such a purpose, it would be extremely difficult to match every style an artist could devise in even one medium.

Fortunately, our goal is merely to composite a computer-generated element into a scene in such a way that it is not distracting. As it turns out, the human eye is much more forgiving when it comes to traditional animation than it is for film and photography. In traditional animation it is common practice to have characters and backgrounds rendered in completely different styles. Perhaps the best example of this is cell animation composited on top of a painting. Such a discrepancy in style would undoubtedly distract viewers in live-action footage, yet the audience accepts it readily in traditional animation. Clearly this should tell us something about human perception. The closer something comes to resembling reality, the more difficult it is to make the eye believe that it is real. Film and photography are a very close, but not exact, representation of what we see with our eyes. As a result, any additional elements we wish to composite into such films have to come as close as the film does to approximating reality. In addition, the eye becomes more sensitive to error as we more closely approximate the reality it expects to see.

Given this information, it may be reasonable to say that what we really want to match is the level to which the non-photorealistic image approximates reality.

Non-photorealistic images are typically, but not always, simplified depictions of some reality. As such, they are interpreted by the eye as simply a representation of reality rather than as reality itself, and as a result the viewer will be much less sensitive to error. For our purposes, this means that it may be good enough to render computer-generated elements so that they appear to have been created by an artist rather than a computer, and in a medium reasonably close to the one used to create the scene we are attempting to match.

III.4. Matching the Complexity

Style and complexity are heavily related issues, and both need to work together to produce the results we want. The complexity in a non-photorealistic scene is dependant on the level of detail that the artist used to depict the scene as well as the medium used to render it. When an artist creates an image, he or she simplifies reality to some degree. Even a seemingly realistic image has been abstracted as the artist has chosen to enhance the interesting elements of the scene, while placing less emphasis on the more mundane details. This is very much the same way that we look at reality. The real world is very complicated, so it is not possible for us to focus on every aspect of what we see in a scene without prolonged study. Instead we can get the majority of information we need about a scene at a glance by noticing only the most vivid and interesting parts of the scene, the elements that catch our eye. Perhaps this is what gives non-photorealistic images such appeal. They allow us to focus only on what is important or exciting about a scene.

To match a non-photorealistic image, we must take into account this ability to abstract and simplify reality. When creating computer-generated elements for film, skilled artists create detailed texuture maps to add complexity, and thus believability,

to already complicated models. This is because film is a close approximation of reality, so matching it requires an equally complicated image from the computer. As non-photorealistic images are simplifications of reality, it would not make sense to create models and textures as complicated as the ones used in photorealistic scenes for use in non-photorealistic scenes. Our eyes are sensitive to differences in detail, so if a character is much more or less detailed than his or her surroundings it will be noticable and potentially distracting.

It is interesting to note that even while we are simplifying and abstracting the models and textures for non-photorealistic scenes, it may be necessary to add complexity during or after rendering. This is because complexity is also dependant on the process the artist used to create the image, as well as the artist's style. If the image was painted, there will be visible brush strokes on the image plane. If the image was done in pen and ink, the artist probably used hatch lines to shade it. This adds another level of complexity to the image that is not present in photorealistic images, and it might necessitate image processing after the render has completed to match it.

Figure 1 shows the four main steps in the process of matching an artistic image. Note that it may be necessary to match style and complexity at the same time as they are heavily related.

III.5. Creating a Consistent Look

Given the success of traditional animation in combining different styles for characters and background scenes, it is difficult to overlook the possibility that this could work when combining computer-generated elements with non-photorealistic images. It seems feasable that if the character matches the scene in terms of camera angle, lighting, color, and complexity, it should be possible to render the character in a

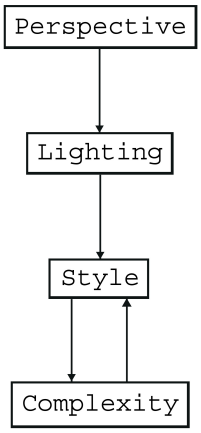

Fig. 1. Main steps for matching artwork.

different style from the original image and still create a composited image that is appealing rather than distracting to the audience.

This concept could be useful in a number of ways. For example, if we were interesed in creating an animation over several scenes using background images created by different artists in a similar media, we might find that if we closely matched the style of each artist the look of our character changed so much during the animation that this in itself was distracting. It is even possible that the audience would have trouble recognizing the character when the scenes changed. Clearly this is not what we want. It would be more useful to decide on a single style and level of complexity for the character that worked reasonably well with all scenes, and only concern ourselves with matching the camera, lighting, and color. Then our character would have a consistent look over all scenes, and be easily recognizable for the audience. This difference in styles for the characters and background could even help to emphasize

them in the same way that rim lights used in film help to draw attention to an actor and pull them out of the background.

III.6. Introduction of Error

Another important concept for non-photorealistic rendering is the introduction of error in the rendered images. It is almost impossible for an artist to create an image that is completely free of error. The perspective may be slightly off, the line quality may not be consistent thoughout the image, the color and shading may not exactly reflect the lighting, or the shadows may be slightly different in terms of length and direction. Even the medium used to create the image can result in mistakes. It simply may not be possible for an artist to work at a certain level of detail because of brush or paper size. These factors all add up to an image that is not quite "perfect" even when the artist has attempted to re-create a real scene. These mistakes are not always detrimental, and in many cases add interest to the final image.

The computer, however, does exactly what its programming tells it to. Any errors in the images it creates are a result of the software that tells it how to create the image rather than a mistake during the creation process. As a result, we can create a very crisp, detailed image with smooth color changes based on lighting and accurate shadows. And this image will certainly look as if it were created by a computer rather than an artist. The problem is that it is too "perfect," too clean, for us to believe that someone created it by hand. To give the impression of being created by a human hand, non-photorealistic renderings need to have some level of error in them.

For the purposes of this thesis, this means that small mistakes are certainly acceptable, and maybe even required. The camera position does not have to be exact, the light direction and shadow can be a little off, and the color and texture

matching can be close enough without being exact. It may even be necessary to modify the surface of the objects being rendered so they are not quite smooth. In the end, we want the viewer to believe that the character was rendered by an artist, not a computer.

III.7. Nonphotorealism in Motion

Even when a computer-generated character matches a scene perfectly in a still frame, it may still be obvious that the character was animated by a computer rather than an artist. This can be a result of the actual motion of the character or camera, or a result of how light interacts with the character's surface as it moves. As this is not an animation thesis, we are more concerned with the light interaction. The easiest way to tell that a seemingly hand-painted characted was created by a computer is the way that the highlight moves across its surface during the animation. Most hand-painted characters have very static lighting, and the highlight tends to stay in the same place as the character moves. Naturally it is difficult for an artist to determine exaclty where the highlight should be for each frame, so the change in the lighting itself is simplified. Sometimes an artist will animate the lighting change well and produce a beautiful series of images. Even in this case, though, the animation is done in the way that the artist wants us to see it, and may not accurately reflect reality or the representation of reality programmed into our rendering software.

The computer uses the rules programmed into it calculate a new position for the highlight every frame based on viewing angle, surface direction, and light position. This could allow us to create some very interesting effects that would be difficult with traditional animation, but could also allow us to create some animations that look very computer-based if it is not carefully controlled. If we wish to create a computer

animation that really looks as if it were done by an artist, we have to animate the motion of the character and the lighting in the same way that an artist would.

CHAPTER IV

IMPLEMENTATION AND RESULTS

As the focus of my thesis is primarily development of a method and set of guidelines, it makes sense that the process I have gone through is one of inspection and invention. I began by closely examining a number of artistic images to get an idea of the functionality I would require from my software in order to create computer-generated elements that would fit into each scene. It quickly became apparent that I would need to match each scene's perspective, lighting, and color palette to render a character from the computer that was anywhere close to a reasonable match for the original image. I also realized that the software would need to have the ability to render the character in a way that gives the impression that the character was rendered by an artist. That is, since I wanted to provide a method and set of guidelines for any artistic style, I would need some way of approximating the style and complexity of the image without having to be too specific about the look of a particular medium or the rendering technique of the original artist. I have intentionally allowed myself a certain amount of freedom from the restraints of producing an exact match for an image in the interest of developing a method and guidelines that can be applied to all types of non-photorealistic images. A good deal of previous research has already been done to develop software that mimics the specific look of almost every medium available, and an attempt to duplicate or improve on this work is beyond the scope of this thesis.

IV.1. Software

To further develop and test my ideas, I had to choose which software packages to implement my method with. I decided on a combination of Alias/Wavefront's Maya, Pixar's Renderman, and Adobe's Photoshop. This choice was based on the functionality present in the software, as well as how readily available it is to people in the industry as well as to myself. As a production-level modeling and animation tool, Maya provided me with the ability to create and animate a character that I would later composite into an artistic image. I also used Maya to control camera placement, perspective, and general light direction. While Maya is capable of producing rendered images of its own and has an interface for creating shaders, I decided that Renderman would give me more control by allowing me to write my shader much as I would a program. As it is the shader that tells the software the specifics of how to render a scene, I knew that this is where I would need the most control and functionality. While Renderman can perform limited compositing operations with two images, I found that I needed Photoshop to create final images that were believable. Figure 2 shows the software I used during each step of the matching process.

IV.2. Shader Development

Development of a set of shaders was an important step towards completing this thesis. I needed the ability to mimic a number of artistic styles so that I could create a close match for almost any image. This would allow me to test my ideas on a variety of visual styles to ensure that the method and guidelines I was developing worked in all cases. I decided early on that I would like to be able to give my shader a limited number of colors to use when rendering an object, in much the same way as an artist would select the color palette to use when creating an image. This is slightly different

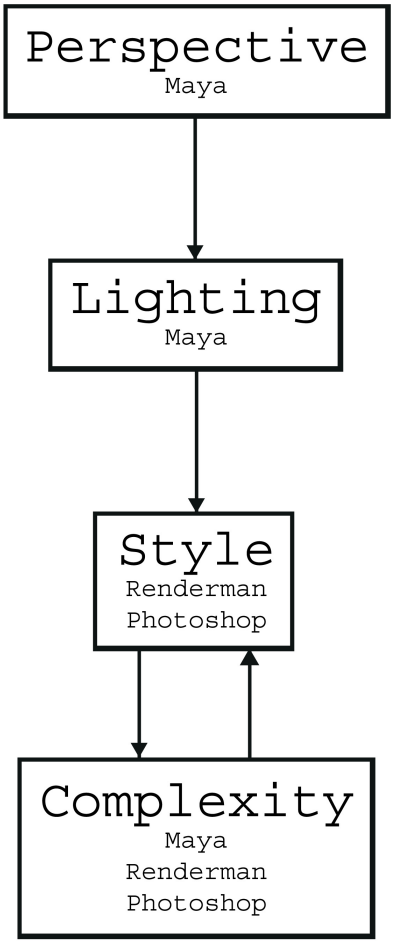

Fig. 2. Software used in each step of the matching process.

than the approach traditionally used when compositing for live-action footage, where the final color of a character is dependant on both the colors given in its textures as well as color from the light sources. These two color sources are multiplied together to compute the color that is rendered to any given pixel in the output image. This approach is very useful for film because it allows artists to give the colors of a character under white light as a texture map, and then match the character with the different lighting in each scene by changing the color of the light sources represented in the computer. This also gives us the ability to match a scene with multiple light sources of different colors. Unfortunately, this approach results in a bit of guess-work for the lighter responsible for matching the scene, as it can be somewhat difficult to determine exactly which colors to use for the lights so that the character will look right when composited into the film.

After examining a few non-photorealistic images, I noticed that in most cases the artist has chosen a general direction in which the light is stronger in order to help bring out the form of the scene with lighted and shadowed areas. Based on this light direction, the artist renders the objects in the scene with a limited number of colors. Clearly the artist is free to choose colors for the lighted and shadowed areas that are as close to or far from reality as he or she desires, so it makes sense that I would need this kind of freedom when selecting colors for my shader. I also decided at this point that it would be best to specify only a single light direction, and specify the colors I wanted the shader to render with based on that direction. Results from the first version of my shader can be seen in Figure 3. Here I chose to use shades of gray to render the images, but these could easily be any set of colors. This version of my shader allowed me to select up to four colors to render a character with, and included a simple algorithm for drawing an outline around an object.

At this point I knew that it would not be too difficult to mimic most comic book

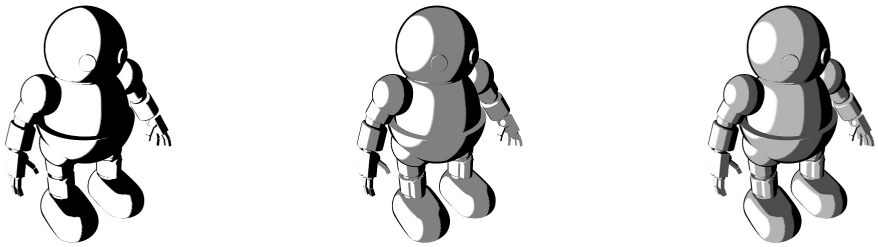

Fig. 3. Original shader with up to four separate colors.

and cartoon styles with my shader. I decided to write a displacement shader to work with it so that I could easily add some interest to the surface as will as a degree of error to give the impression that the rendering was done by an artist. The results from this combination can be seen in Figure 4, which is the same as above with the displacement shader added.

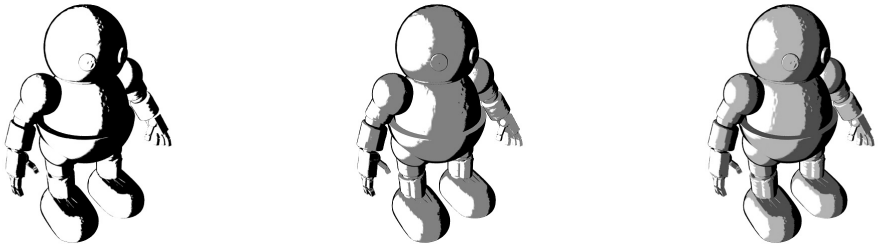

Fig. 4. Original shader combined with displacement shader.

While separating the colors used in the shader would be useful for matching comic and cartoon styles, clearly I would need to transition smoothly between the colors to mimic other artistic styles, such as painting or charcoal. Also, I decided that I would need to be able to render in a fairly realistic style if I were to have any hope of matching some of the more richly detailed images, so I added an ambient occlusion calculation to my shader. This lowers the intensity of the light in the creases and grooves of an object, and helps to bring out the form. Figure 5 shows a comparison

of smooth shading with line, displacement, and ambient occlusion.

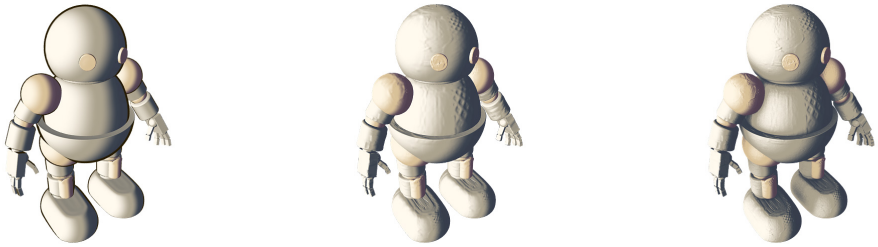

Fig. 5. Smooth shading with line, displacement, and occlusion.

I was fairly pleased with this version of my shader, but after attempting a few compositing tests I noticed that it had a fundamental flaw. As I stated earlier, my design for this shader was to give it a limited number of colors and have it render an object with those colors based on a single light direction. This was not only a close approximation of how an artist would render a scene, but would also simplify the guess-work required to match a scene by eliminating the multiplication of light and object color used in film. As a trade-off, this would make matching a scene with multiple main light sources nearly impossible, but I knew that I would not need this functionality for most artistic images. Unfortunately, this design conflicted somewhat with how the lighting calculations were being done in the rendering software. Traditionally, light is represented in the computer as an approximation of reality. If there is no light on an object, that object is rendered as black, or in the case of my shader with whatever color I specified to use when there is no light. If there is full white light on an object, the object is rendered with the colors specified by its shader or texture map. As the light becomes less direct or intense, the object is rendered with a gradient.

The flaw in my shader resulted from the fact that the light intensity calculated for a single light on a surface goes from 0 where the light direction is perpendicular to

the normal of the surface, to 1 where the the light direction is in the opposite direction from the normal of the surface. This means that for a single light there is a smooth gradient of light intensity on the half of a sphere facing the light, while the other half of the sphere is in complete darkness. Using this representation, it is not possible to light an object with a single light in a way that will look good from all directions. For this reason, most of the digital characters you see composited into film have anywhere from as little as three to hundreds of lights illuminating them depending on the scene. And every one of these lights adds to the time required to render an image. So in the interest of preserving my shader's ease of use and keeping render times as low as possible, I extended the lighting calculations used in my shader to wrap the light from a single source around the entire surface of an object. The result of this extension can be seen in Figure 6. In both images the character is completely illuminated based on a single light direction, and I have separated the colors to illustrate how they are rendered on the surface.

Fig. 6. Extended illumination with eight separate colors.

Figure 7 shows results from the final version of my shader. I added self-shadows and highlights to allow for increased realism, and the shader works with both displacement and texture maps to add detail and to introduce error.

I have included additional examples in Figure 8 to demonstrate less realistic uses

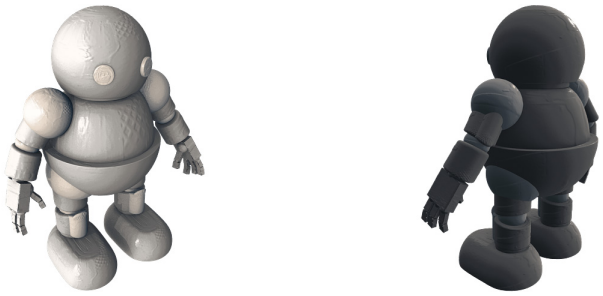

Fig. 7. Final version with smooth shading, highlights, and shadows.

of my shader.

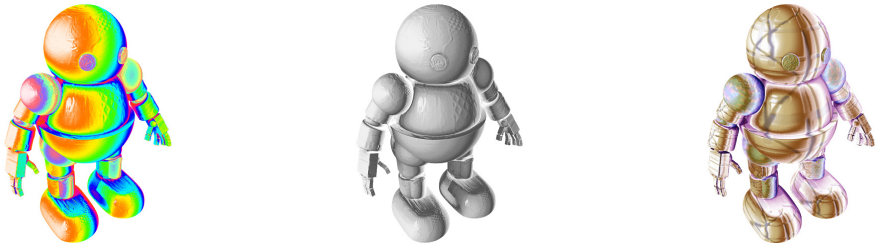

Fig. 8. Additional examples.

IV.3. Artistic Images

In order to really test my shader and my ideas, I needed to composite some digital characters into non-photorealistic scenes. Technically these scenes could be any image that does not look like a photograph, but I prefered to use scenes that were realistic enough that I could clearly distinguish form, ground, perspective, and lighting. It was also helpful to pick scenes with enough open space in them that I could add a digital character of substantial size. A number of the images I used for backgrounds are based on my own photographs, which I spent some time working on in Photoshop until they began to look more artistic than photographic. Only after finishing working

on a scene did I attempt to match it and composite a digital character into it. I wanted to ensure that the approach I was developing would be the same for any given artistic image, so I did not record any camera or lighting information to help me match the scene.

Figure 9 shows some of the background images I chose to test my shader with. These images are based on my own photography, and are the ones I will use to demonstrate the compositing process.

Fig. 9. Photographic still backgrounds.

In addition to testing how well I could match a still image, I wanted to see how well a digital character in motion fit in with a scene. Figure 10 shows the background images I created to help with this test. They are all of the same scene rendered in several different styles. This is helpful because I can change just the rendering style to see which images my shader works well for and which it has difficulty matching.

Fig. 10. Photographic animation backgrounds.

IV.4. Compositing Process

Once I had selected the background images I would use, I could document the process I went through to match each image. I began the compositing process by first matching the perspective and viewing angle as closely as possible. If the original image included a horizon line and vanishing points, these helped to make the matching process a bit simpler. Using this information, it is possible to make a few primitive shapes in Maya, such as planes and rectangles, and adjust the camera settings until the perspective in the Maya scene is the same as in the background image.

If there is no definitive way to find the horizon line or vanishing points in the background image, it is necessary to do a bit of guess-work. In this case, it helps to move a digital character across the ground plane in the Maya scene while adjusting the camera until the character gives the appearance of moving through the scene as shown in Figure 11. Keep in mind that everyone will have exactly the same perspective information when viewing the final composite, so the matching needs to

be believable, but not necessarily exact.

Fig. 11. Perspective matching with limited information.

After completing the perspective matching for the scene, the next step is to match the light direction. I used a directional light in Maya to represent the direction of the strongest lighting. I did this not only because the directional light requires less lighting calculations, and thus less render time, than any other light type, but also because it dramatically simplifies light placement when matching background images. Directional lights in Maya give a direction and intensity independant of where they are placed in the scene. I can use this to my advantage by placing the light on the ground plane in Maya, which theoretically should correspond to the ground plane in the background image. Then I can move the light along the ground plane so that it appears to be at the base of one of the objects in the original scene that is casting a shadow. At this point I can rotate the light around the normal to the ground plane so that it is parallel with the object's shadow, and then rotate the light to point down until it appears to connect a point on the top of the object with the corresponding point on the shadow.

This method is illustrated in Figure 12, and gives a very close approximation of the light direction provided that the original image has a shadow-casting object and a relatively flat ground plane to base it on. For other images, it is still possible to create a fairly close match for the light direction by rendering a few test images

and correcting the directional light as necessary. I have found that a small amount of error in matching both the perspective and light direction does not significantly reduce the believability of the final image, and may even add to the impression that the character was rendered by an artist.

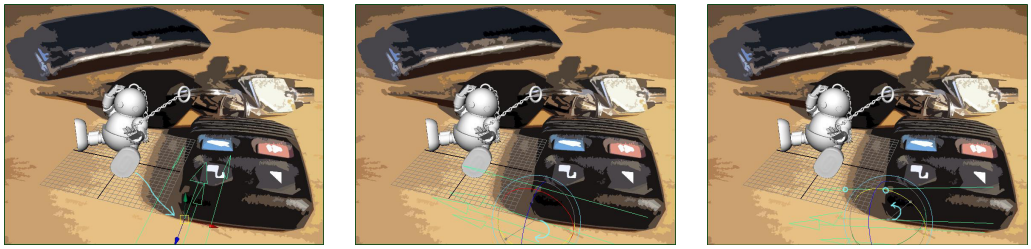

Fig. 12. Light matching with a directional light.

Once the perspective and lighting have been matched it is time to place the character in the scene and create a color palette for it that matches, or is based on, the one the artist used in the background image. I used the shader that I had written earlier to help me with this process. After examing the background image, I selected colors from it to use as a starting point for creating the character's color scheme as shown in Figure 13. It is best to select these colors all from the object in the scene that most closely resembles how the character should look in the scene. Since the characters I added to the scenes are metallic, I based their colors on a metallic object in the original scene if one was available. Also, it is important to sample the full range of color used in both the lighted and shadowed areas of the object to ensure that the digital character will be rendered with the same level of contrast as the background.

After creating a color palette, I adjusted my shader to match the style of the scene as closely as possible. I found that I was able to match the style of each scene reasonably well without needing to perform any image processing on the rendered images. In some cases I used Photoshop to add noise or blur the rendered images

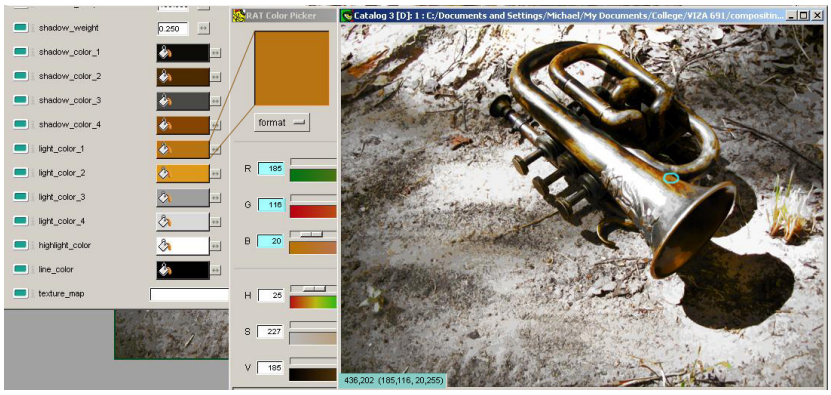

Fig. 13. Creating a color palette.

slightly so they would appear to be closer to the quality of the original image. This technique is often used for matching film, so it seemed reasonable to use it here as well.

To finish adding a character to a scene, I rendered a shadow for the character in Maya. When necessary, I rendered a contact shadow as well using my Renderman shader to help give the appearance that the character was actually touching the ground. Depending on the complexity of the original image, the shadow rendered by Maya was often too smooth or too solid to really fit in the scene. In a number of cases I had to adjust the shadow using some image processing in Photoshop to get the look I needed. An example of this is shown in Figure 14.

Once I had rendered everything I needed to add the digital character to the scene, I combined all of the images in Photoshop to create the final composite. Figure 15 illustrates the entire compositing process for one of these images.

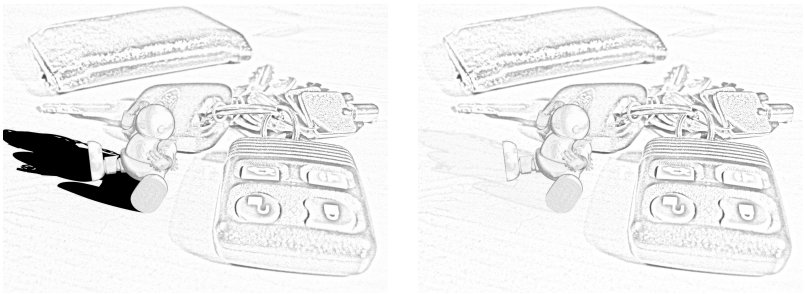

Fig. 14. Adjusting the shadow.

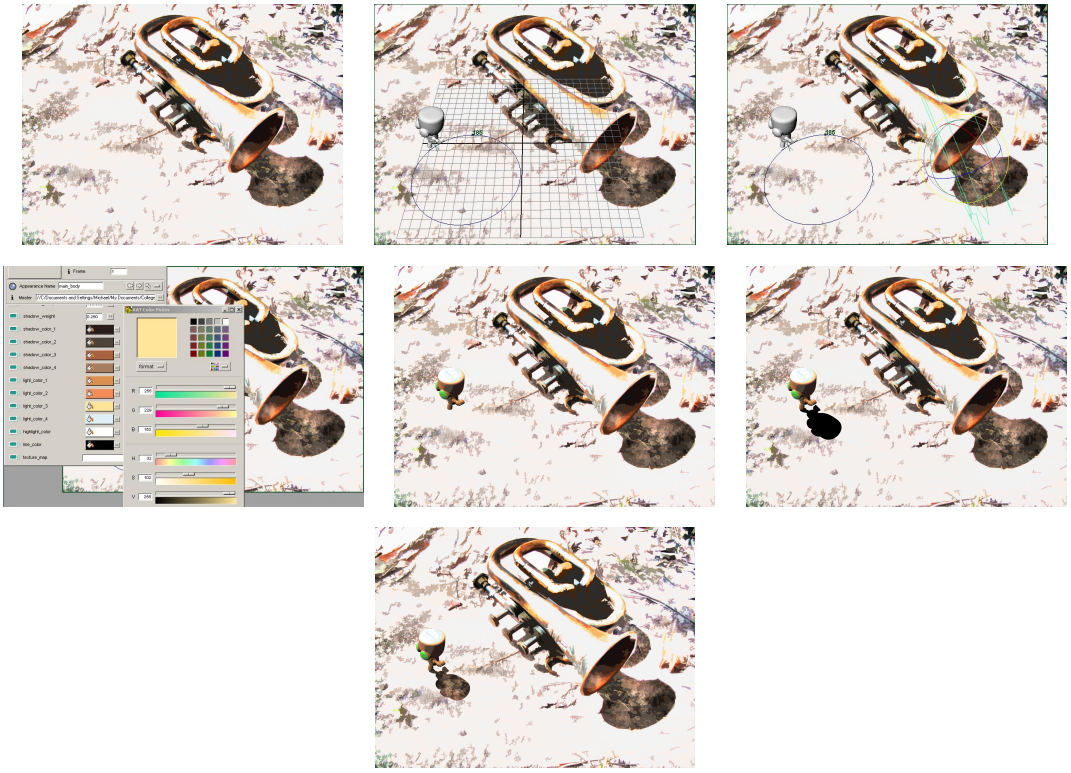

Fig. 15. Compositing process.

IV.5. Completed Images

Figure 16 shows some of the final composites I created using the methods and guidelines I have developed, including a still frame from each animation. I also created many composites with traditional artwork from other artists to add strength to my work, but I do not have permission to reprint them here.

Fig. 16. Composites with photography.

CHAPTER V

CONCLUSION AND FUTURE WORK

V.1. Evaluation

Having completed a number of successfully composited images, I am satisfied with the method and guidelines that I have come up with. I have been able to create a very close match for every background image I have worked with using the process described above. Of course, it should certainly be possible for an artist to create an image that would be very difficult or even impossible to match with the current implementation of my shader and the software I am using, but even in this case the basic guidelines and method would still be useful. The main criticism I have of my results is that while the characters match very well in a still frame, sometimes it may become clear that they were rendered by a computer rather than an artist as soon as they begin to move. This flaw results not from my methodology, but from the underlying algorithms present in the rendering software. The light interaction on the surface of the character is not simplified or exagerated in the way that an artist would render it, but is recomputed every frame using the approximation of lighting in the rendering algorithm. Naturally this can result in a different look than we are used to seeing in a hand-drawn animation, so in some cases it may be somewhat difficult for us to believe that the character was rendered by an artist because of this.

V.2. Future Work

There is a good deal of future work that could be done in this area. One possibility is implementing the algorithms for simulating paint and watercolor effects, or any

other medium, presented by previous researchers and adding them to custom rendering software specifically designed for matching artistic images. In fact, most of the concepts and algorithms introduced by previous researchers can be further developed and integrated with the ideas presented in this thesis to create an even closer match for traditional artwork in almost any style. It might also be possible to automate a good deal of the matching process by allowing someone to give an image, or a region of an image, as input and then use that information to adjust how the digital character is rendered. This would probably involve the development of an algorithm that could examine a set of images to learn the style of the artist or medium, and then use what it has learned to mimic that style.

There is also some work that could be done to improve characters in motion to truly give the impression that the animation was rendered by an artist. The animations I have created have the same problem as many other films and series that incorporate digital elements with traditional animation. Even when the characters match the perspective, lighting, color, and style of the scene, the way that they interact with light as they move gives away the fact that they were rendered by a computer rather than an artist. When the light interactions of the characters rendered by artists are highly simplified, this discrepancy becomes obvious and may even be distracting. However, when the artists have focused on creating more detailed or realistic lighting animations it is considerably more difficult to see. In this case, it may even become challenging to render the digital character so that its lighting looks as real as the rest of the scene. The trick is to get the computer and the artist to work with a similar interpretation of light.

REFERENCES

[1] G. Winkenback and D.H. Salesin, "Computer-Generated Pen-and-Ink Illustration," *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pp. 91-100, 1994.

[2] B.J. Meier, "Painterly Rendering for Animation," *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 477-484, 1996.

[3] G. Winkenback and D.H. Salesin, "Rendering Parametric Surfaces in Pen and Ink," *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pp. 469-476, 1996.

[4] C.J. Curtis, S.E. Anderson, J.E. Seims, K.W. Fleischer, and D.H. Salesin, "Computer-Generated Watercolor," *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 421-430, 1997.

[5] D.N. Wood, A. Finkelstein, J.F. Hughes, C.E. Thayer, and D.H. Salesin, "Multiperspective Panoramas for Cel Animation," *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 243-250, 1997.

[6] W.T. Correa, R.J. Jensen, C.E. Thayer, and A. Finkelstein, "Texture Mapping for Cel Animation," *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 435-446, 1998.

[7] A. Hertzmann, "Painterly Rendering with Curved Brush Strokes of Multiple Sizes," *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 453-460, 1998.

[8] M.A. Kowalski, L. Markosian, J.D. Northrup, L. Bourdev, R. Barzel, L.S. Holden, and J.F. Hughes, "Art-Based Rendering of Fur, Grass, and Trees," *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 433-438, 1999.

[9] O. Deussen and T. Strothotte, "Computer-Generated Pen-and-Ink Illustration of Trees," *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 13-18, 2000.

[10] A. Hertzmann and K. Perlin, "Painterly Rendering for Video and Interaction," *Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering*, pp. 7-12, 2000.

[11] A. Lake, C. Marshall, M. Harris, and M. Blackstein, "Stylized Rendering Techniques for Scalable Real-Time 3D Animation," *Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering*, pp. 13-22, 2000.

[12] L. Markosian, B.J. Meier, M.A. Kowalski, L.S. Holden, J.D. Northrup, and J.F. Hughes, "Art-Based Rendering with Continuous Levels of Detail," *Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering*, pp. 59-66, 2000.

[13] J.D. Northrup and L. Markosian, "Artistic Silhouettes: A Hybrid Approach," *Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering*, pp. 31-37, 2000.

[14] S.M.F. Treavett and M. Chen, "Pen-and-Ink Rendering in Volume Visualization," *Proceedings of the Conference on Visualization '00*, pp. 203-211, 2000.

[15] E. Praun, H. Hoppe, M. Webb, and A. Finkelstein, "Real-Time Hatching," *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 581-586, 2001.

[16] R. Raskar, "Hardware Support for Non-Photorealistic Rendering," *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware*, pp. 41-47, 2001.

[17] A. Agarwala, "SnakeToonz: A Semi-Automatic Approach to Creating Cel Animation from Video," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 139-146, 2002.

[18] S. Chenney, M. Pingel, R. Iverson, and M. Szymanski, "Simulating Cartoon Style Animation," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 133-138, 2002.

[19] D. DeCarlo and A. Santella, "Stylization and Abstraction of Photographs," *ACM Transactions on Graphics (TOG), Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, vol. 21, no. 3, pp. 769-776, 2002.

[20] F. Durand, "An Invitation to Discuss Computer Depiction," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 111-124, 2002.

[21] B. Freundenberg, M. Masuch, and T. Strothotte, "Real-Time Halftoning: A Primitive for Non-Photorealistic Shading," *Proceedings of the 13th Eurographics Rendering Workshop*, pp. 227-231, 2002.

[22] N. Halper, S. Schlechtweg, and T. Strothotte, "Creating Non-Photorealistic Images the Designer's Way," *Proceedings of the 2nd International Symposium on*

Non-Photorealistic Animation and Rendering, pp. 97-104, 2002.

[23] A. Hertzmann, "Fast Paint Texture," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 91-96, 2002.

[24] P. Jodoin, E. Epstein, M.G. Piche, and V. Ostromoukhov, "Hatching by Example: A Statistical Approach," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 29-36, 2002.

[25] S.F. Johnston, "Lumo: Illumination for Cel Animation," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 45-52, 2002.

[26] R. Kalnins, L. Markosian, B.J. Meier, M.A. Kowalski, J.C. Lee, P.L. Davidson, M. Webb, J.F. Hughes, and A. Finkelstein, "WYSIWYG NPR: Drawing Strokes Directly on 3D Models," *ACM Transactions on Graphics (TOG), Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, vol. 21, no. 3, pp. 755-762, 2002.

[27] E.B. Lum and K. Ma, "Hardware-Accelerated Parallel Non-Photorealistic Volume Rendering," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 67-74, 2002.

[28] A. Majumder and M. Gopi, "Hardware Accelerated Real Time Charcoal Rendering," *Proceedings of the 2nd International Symposium on Non-Photorealistic Animation and Rendering*, pp. 59-66, 2002.

[29] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, "Suggestive Contours for Conveying Shape," *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 848-855, 2003.

[30] R.D. Kalnins, P.L. Davidson, L. Markosion, and A. Finkelstein, "Coherent Stylized Silhouettes," *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 856-861, 2003.

[31] D. Kirsanov, P.V. Sander, and S.J. Gortler, "Simple Silhouettes for Complex Surfaces," *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pp. 102-106, 2003.

VITA

Michael Stanley
10316 Lisa Jean Dr.
Crowley, TX 76036
michael@viz.tamu.edu

Education

M.S. in Visualization Sciences Texas A&M University, August 2005
B.S. in Computer Science Texas A&M University, May 2002