Application, Rendering and Display of Automotive Paint

by

Jonathan Konieczny

Dr Gary W. Meyer
Advisor

Presented to the Department of Computer
and Information Science
of the University of Minnesota
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy, Computer Science.
November 2009.

Acknowledgements

My thanks to my Professor Gary Meyer, for his support, guidance and excellent suggestions for the projects. His expertise and experience were vital in the completion of this work. I would also like to thank Clement Shimizu for his numerous contributions to this work, along with my other collaborators: Seth Berrier, D'nardo Colucci, John Heckman, Mark Manyen, and Marty Rabens.

Several industrial partners were also instrumental in the work presented in this dissertation. DuPont Paints was kind enough to allow us to visit their own paint testing facility and track their spray painters using real spray guns. The data gathered from that visit was essential to creating the spray paint training system, as well as the later airbrush system. Lehmans Garage also allowed us to visit their facility and photograph their spray booths for use in the virtual environment. Lehmans Garage also donated their spray painter's time for testing the system. Finally, Heat-n-Glo industries helped test early versions of the spray paint system.

The airbrush system was successful largely due to the expert skills of artists who tested it. Leah Gall gave great input throughout the creation of the virtual airbrush, and produced wonderful pieces of art for us using the system. Artists Gustavo Lira and Marc Dahlin also tested the final product and created great examples of artwork with it. I would also like to thank Dr. Mark Cook and Dr. Ken Roberts for their input on the visualization system, and Vital Images for providing us with their medical data sets. The frame for the flexible screen was constructed by Stephen Shimizu.

This research was done as a part of the Digital Technology Center at the University of Minnesota. It was funded by the National Science Foundation under grant number IIP-0438693. Finally, I would like to thank the members of my thesis review and examination committee: Victoria Interrante, Daniel Kersten, Joseph Konstan and Gary Meyer.

Dedication

In loving memory of my father, Martin P. Konieczny.

Abstract

New computer graphics tools are developed for use in the automotive paint design and paint application industries. These tools are designed to aid in every aspect of automotive painting from initial paint design all the way to viewing a final spray paint job. This thesis also provides new computer graphics techniques by leveraging industrial appearance standards and measurement instruments to yield new interaction, rendering and display algorithms.

First, a system is introduced for the simulation of spray painting. Head mounted display goggles are combined with a tracking system to allow users to paint a virtual surface with a spray gun. Ray tracing is used to simulate droplets landing on the surface of the object, allowing arbitrary shapes and spray gun patterns to be used. This system is combined with previous research on spray gun characteristics to provide a realistic simulation of the spray paint including the effects of viscosity, air pressure, and paint pressure. Experiments were performed to validate the system for use as a training tool.

Next, a virtual airbrush tool is introduced. The basic particle simulation used in the spray paint system is modified to include the finer control needed for airbrushing. Paint mixing between colors applied to the surface is modeled using Kubelka-Munk theory. Computerized stencils, including semi-permeable stencils, can be manually positioned by the artist or projected onto the object's surface. The resulting airbrush system can be used by airbrush artists to both practice their craft as well as to decorate virtual models.

The dissertation then utilizes an industrial measurement instrument to simulate the surface roughness in automotive paint finishes. This simulation is integrated with a design interface to enable industry professionals to devise new paints that have detailed surface effects. The industrial measurement device can also be used to rapidly measure and render real world materials without the need for elaborate full BRDF acquisition tools. Finally,

the surface model developed in this research can be used to study human detection of small scale surface roughness.

Lastly, new projection systems are introduced to display the paints generated by the previous algorithms. A desired paint is projected onto the surface of an object, making that object appear to be painted with that material. This allows painters to evaluate how the final painted surface will look in a very natural way. A novel projection system is also described in which the user can hold and evaluate a flexible sample of virtual paint. This provides a more compelling display of the paint than a standard 2D monitor can generate. These display methods can also be used to display materials other than paint, potentially benefiting a large range of industries.

The combination of these tools is intended to allow spray painters to design paints, paint them onto an automobile or other object, and verify exactly what the result will look like before ever having to manufacture the paint itself. The use of these tools could therefore reduce wasted paint, speed up training times, and allow for precise design of automobile appearance: all before ever manufacturing any real paint. In addition to aiding paint industries, these tools enhance the current state of the art in computer graphics. The airbrush tool provides a new texture creation system that existing airbrush artists can use with little to no training; the surface roughness simulation provides enhanced rendering of automotive paint that could be used in movies and games; and the projection display system improves the state of the art in augmented reality systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computer graphics has become increasingly pervasive in recent years. While its primary applications are still in the entertainment industry, it is becoming commonplace in other fields as well. For instance, the car industry utilizes computer graphics in order to model the structure of cars before they are built, and volume visualization is regularly used in the medical field to analyze MRI and CT data. This dissertation introduces computer graphics tools into an industrial field which has previously used very little computer technology in its process: automotive paint design and application.

Spray paint application is used throughout the automotive industry, as new cars are painted, damaged cars are repainted, and car owners have custom paint and decal work performed. Currently, in order to view how a particular paint will look on a surface, that paint must be physically applied. If the resulting appearance isn't what was desired, the job must be redone. This means that getting a precise look can be very costly in both time spent and paint used. In addition to typical application, training also currently uses real paint. Spray painters will paint test panels until they are able to achieve an acceptable level of quality. Since many new paints require a high level of skill in order to be applied

Figure 1.1: A spray painter painting a test sample. Note the respirator and protective gear. By giving tools to train and test paints virtually, the time spent spraying real paint could be reduced, saving paint as well as having environmental and health benefits.

properly, automotive painters frequently have to retrain themselves when new paints become available. This leads to a lot of wasted paint that is never applied to a finished product. Not only does this wasted paint cost money, but there are also environmental concerns. Many parts of the paint are environmentally harmful, in particular the solvents when the paint initially dries. It is therefore of interest to save as much paint as possible as well as to design new paints that are environmentally friendly.

Many other industries such as construction, ship building, and interior design utilize spray paint in a very similar manner to automotive spray painting. Therefore, tools that are developed for automotive paint can be easily used in these areas with little to no

Figure 1.2: An image of a motorcycle gas tank airbrushed with green flames. The virtual airbrush system can allow airbrush artists to create virtual versions of this design, both allowing them to practice as well as to decorate 3D models for use in games and movies.

modification. The addition of virtual spray painting and training software could have significant benefits to these industries, potentially saving paint, lowering training times, defect rates, and allowing more precise design of desired paint finishes.

In order to provide a suite of tools to aid in automotive paint training and design, this thesis covers a number of relevant areas in automotive paint application, rendering, and display. To aid in spray paint training, a virtual spray system is developed. This system gives users the ability to paint with a virtual spray gun and view the resulting paint distribution on the surface they are painting. Training virtually rather than with real paint allows painters to be trained without wasting any paint, potentially providing a significant

Figure 1.3: An image of orange peel. This is a very common paint defect in the automotive industry. Proper simulation of this imperfection allows for industry professionals to properly assess virtual paints jobs as well as allowing them to design tolerances into paint application standards.

savings in paint costs.

The virtual spray paint system also provides the basis for a virtual airbrush system. Airbrushing is used in the automotive industry to decal automobiles and motorcycles. In addition, it has significant uses outside the automotive industry, such as pottery glazing, artistic canvas painting, body painting, taxidermy and object restoration. The airbrush system developed in this dissertation provides airbrush artists with everything they need to take their skills directly from real airbrushes into the virtual domain, including: full 3D stencils, realistic paint mixing, a tracked electronic airbrush that looks and feels exactly like a real airbrush, and color selection tools.

Once the paint has been applied using the above spray paint and airbrush tools, it is important to render and display the result back in a convincing manner. This ensures

that the users are able to properly evaluate the virtual paint job and judge if a trainee or designed paint is performing as expected. Therefore, this research also includes new rendering and display methods to supplement current computer graphics techniques in displaying paint and surface finishes.

First, an orange peel measurement and rendering method is introduced. This allows for proper simulation of a common spray paint defect known as orange peel. This effect is a visible bumpiness of the paint surface, so named for its resemblance to the skin of an orange. Proper simulation of orange peel can help paint designers evaluate tolerances for application of particular paints onto automobiles, as well as help train painters. In addition this dissertation develops an algorithm to determine if orange peel is visible given varying surface material and lighting parameters.

Beyond proper rendering of paint, it is also important to display it back to the user convincingly. There are some known problems with current 2D displays in providing a compelling simulation. First, users can see that the display itself is only two dimensional. This leads to the image presented appearing as if it is a picture, rather than a real object. Second, you can't hold the display in your hand and view how the lighting plays off the surface as the sample is viewed from varying angles. Therefore, in order to properly show the final rendered paint to the user in a convincing manner, this thesis also presents a novel projection display system. This system projects a desired paint onto the surface of an object, making that object appear to be painted with that material. This allows painters to evaluate how the final painted surface will look in a very natural way. A system is also described in which the user can hold a flexible sample in their hands, and the desired paint is projected onto that sample as the user views and flexes that sample in their hands. This provides a more compelling display of the paint than a standard 2D monitor can generate.

In addition to providing significant benefits to the paint design and application in-

dustry, this work also enhances the state of the art in computer graphics. The airbrush system described above can not only be used to simulate automotive decaling, but can also be used to paint any virtual object. This allows airbrush artists to use their skills to decorate 3D models and create virtual artwork, providing a WYSIWYG texture creation system that could be used to supplement commercial tools such as Adobe Photoshop™, Pixologic ZBrush™, and Maxon BodyPaint3D™.

Proper simulation of orange peel is not only important to industrial spray painters, but can also be used to give better renderings of automobiles for movies and computer games. Not only that, but the measurement and rendering of surface roughness provided by the research could potentially be applied to other complex surfaces such as vinyl, leather, and certain fabrics. This software also provides the foundation for studies on human detection of surface roughness with varying factors such as object surface lightness, bump wavelength and amplitude, and even surface color.

Finally, the display technologies developed in this dissertation can be used to display any material, not just automotive paint. This allows anyone who wishes to design materials virtually to view the results of their work in a realistic manner. Also, the interactivity of the projector display presented in this research allows for the visualization of different data types, such as slices of 3D volumes. Thus, although the research contained in this thesis focuses on the application and display of automotive spray paint, a much wider variety of material industries and computer graphics can benefit from it.

The remainder of this dissertation provides details on research performed in order to create a convincing simulation of automotive paint appearance and application. Chapter 2 discusses research on the creation of a virtual spray paint tool. This tool is designed as a high-fidelity virtual version of a real spray gun, to both drive later rendering of the paint as well as to give the user a natural method of designing and training with spray

paints. Chapter 3 presents the virtual airbrush system, which extends the work presented in Chapter 2 to allow the addition of finer details such as automotive decals. It also shows how airbrush artists can use the system to create virtual artwork and decorate 3D models. Chapter 4 presents research done on simulation of orange peel and surface roughness using the Byk Gardner Wave-Scan. The Wave-Scan is used to measure painted surfaces and gives information back on surface roughness. These readings can then be used to drive design and rendering of new paints. In addition it can be employed to properly render existing paints, allowing the graphics community to benefit from research already done in the automotive industry. Chapter 5 provides details on research performed on projection display devices. These systems are used to display the above designed spray paints, giving the user a completely natural and realistic display of the final rendering. Chapter 6 gives closing thoughts and possible future directions this research could follow.

Chapter 2

VR Spray Painting

Training spray painters to apply modern paints can be an expensive process. Paints can vary significantly in how they must be applied to a surface, forcing painters to vary spray gun settings, speed of spray gun movement, and distance of the gun from the surface of the object. Therefore, training new painters can be costly in both time spent training the painter and in the amount of paint used. Even experienced painters may need to re-train for newly formulated paints that require careful application techniques to achieve proper results. When working with real paints, this training must be performed in an expensive spray booth with both the instructor and the pupil wearing protective clothing and bulky respirator masks.

The system introduced in this chapter is designed to provide a realistic spray paint training experience. The aim of this system is to be useful for both spray painters and paint designers. Spray painters can be cheaply and effectively trained on the simulation without the cost of using real paint, while paint designers can design new paints and test out how easy they are to spray, as well as how they will look on the surface. This work also lays the foundation for the airbrush system described in the next chapter.

Portions of this chapter have appeared in ISVC 08: International Symposium on Visual Computing [42]

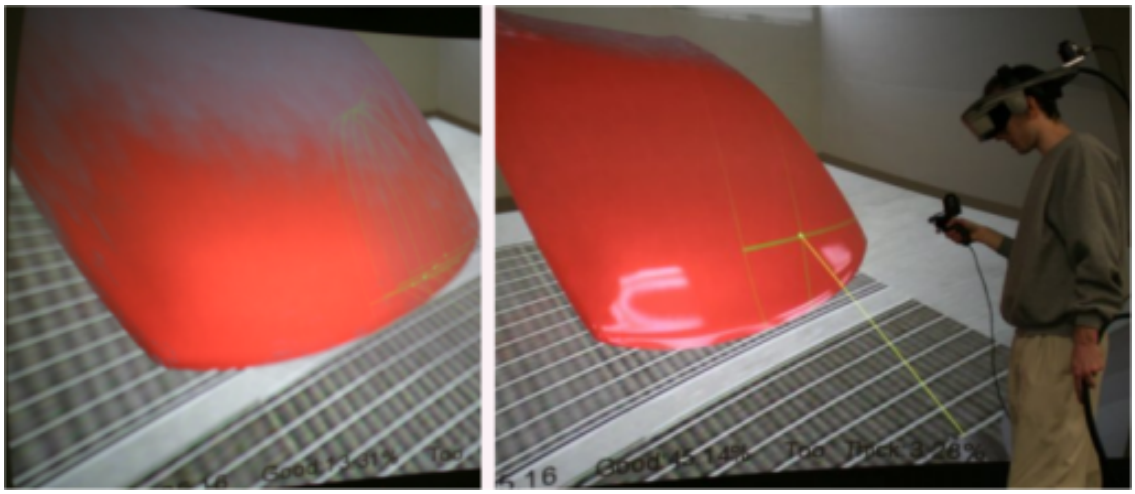

Figure 2.1: Left: Photorealistic rendering (as seen in headset) of directionally diffuse paint on a car hood. Note that some minor artifacts can be seen from too few rays being used to simulate the gun. Right: The final result of the painted car hood after a gloss coat has been applied.

The basic system setup includes a tracked head mounted display along with a tracked spray gun. When the user presses the trigger on the spray gun, a simulation of spraying paint from that gun is performed. This allows a user to use the gun they are holding to paint virtual objects inside the simulation. This painting can be represented on the object in one of two ways. First, it can be shown as a paint density map on the object, informing the user if they have sprayed the correct thickness of paint on the object. Second, a realistic simulation of the paint can be shown which displays what that paint would actually look like on the object.

In addition to the basic functionality of painting, there are several other factors that are essential to the painting process:

First, the environment in which spray painters work is very important. The lighting in the environment in which a painter works can either help or hinder a painter's effort to evaluate the quality of his/her own work. For instance, having approximately even lighting

across the surface of the object prevents a paint job from looking artificially uneven. At the same time, having some pattern on the light can be necessary to spot such defects as orange peel. Auto body painters, therefore, work in paint booths which have very carefully chosen lighting conditions. The simulation given in this chapter takes this into account, allowing the user to select what environment they wish to work in, including a paint booth environment that was carefully captured on site at an actual auto body facility.

Second, the properties of both the spray gun and paint being used can have a significant effect on how that paint sprays onto the surface. For example, some paints may spray faster, forcing the user to move the spray gun more quickly to achieve an even coating. The system described in this chapter uses previous research performed on spray paint guns to add variables such as paint viscosity, paint pressure, air pressure, and distance to achieve a realistic simulation of various paints that a spray painter may want to use.

Finally, reproducibility is important in spray painting. When a car is manufactured, the parts may be painted separately before the final assembly. All the parts must be painted consistently, or they may not match when they are put together. The first way this simulation helps aid consistency is by showing the paint thickness to the user. The user knows both how much paint he/she has applied, along with the "target" thickness that should be placed on the surface. In this way, they can train until they can paint the entire object to within some acceptable deviation from the perfect paint thickness. In addition, a playback feature is included in the simulation. This could allow an expert to paint the object, with his movements recorded so that trainees anywhere can view exactly how he painted the object. This could also be used to "train" mechanical painting arms for car parts that don't require human painters.

In order to validate the system as both a reasonable simulation of spray painting and as a useful training tool, several user experiments were performed. The performance of

the system has been evaluated by both expert and amateur spray painters, and spray jobs performed by painters with real paints are compared to jobs done by the virtual system.

2.1 Relevant Work

This section covers relevant previous research on paint simulation, with both brush and spray paints. First, simulation of paint appearance, including both artistic paints in addition to automotive paints is described. Then, relevant work on paint application methods is given. This includes both analytical models that only simulate the final appearance of paint, as well as physical simulations of paint brushes. This lays the foundation for the new research given in both this chapter as well as the next chapter on computerized airbrush simulation.

2.1.1 Simulation of Paint Appearance

There have been numerous papers on simulation of paint appearance. For simplicity, this chapter will first explore artistic paints (brush, calligraphy, and the like), then metallic/industrial paints.

Artistic Paint

The most commonly used method for the reproduction of artistic paints, which is generally considered to give convincing results is the Kubelka-Munk (KM) model [30]. This model has been used for the rendering of many paints, including watercolors [13], oil and acrylics [4], and even wax crayons [67].

The KM model simulates three properties of each paint layer: transmittance, scattering and interference [67]. “At any location in the paint, light from the surface is moving

deeper into the material and light that has reflected from the substrate is traveling back toward the top of the film. A certain fraction, K, of the light traveling in each direction will be absorbed by the material, and another portion, S, will be scattered." [30]. See Figure 2.2 for a visual example of K and S according to wavelength for sample materials.

This can be done for each desired wavelength in the simulation. The number of wavelengths used varies from 3 (RGB) used by Rudolf [67] and Curtis [13], to a full spectral rendering used by Johnson and Fairchild [38]. The most complicated method used is probably [4], which begins with 101 K and S samples for all paints and lights, then uses Gaussian quadrature to find 8 representative wavelengths (and weights for these wavelengths) in order to reduce the problem to a size that can be solved in real time.

While most research on paint praises KM as being quite accurate, there are some important limitations:

1. KM gives the diffuse reflectance of the paint. In order to fully represent the reflectance, a specular component must be added. Most papers on the subject simply disregard specularity [67][13], or add a simple Phong based specular highlight [4].

2. It assumes all colorant layers are in mediums of the same refractive index, which is always violated by having an "air to pigment layer."

3. Pigment particles are oriented randomly. This is violated by most metallic paints, but kept for many other paints (such as watercolor).

4. The KM equations apply to only a single wavelength at a time. While this is acceptable for most paints, it can be violated by fluorescent paints.

5. Particle size is uniform (or at least relatively so). In reality, this assumption is violated in many paints, although the violation of this property appears to not cause grossly inaccurate results in most cases [13].

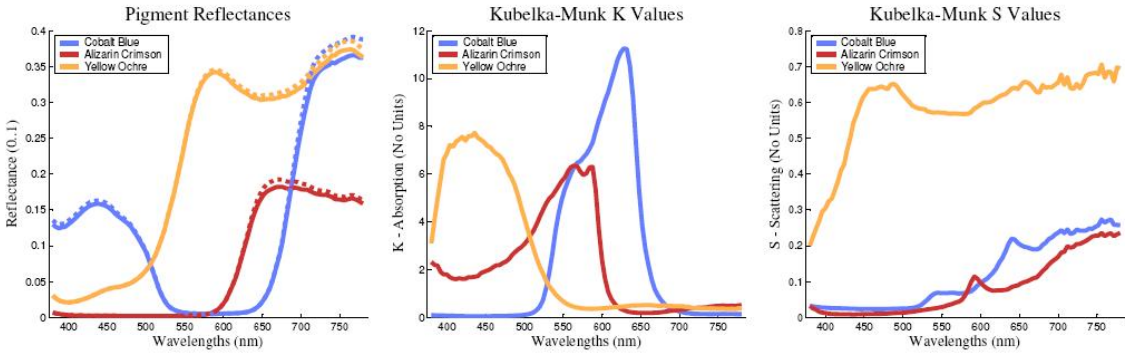

Figure 2.2: Sample Kubelka-Munk Values. The left graph show measured reflectances, and Baxter et al.'s computed recflectances after solving for the K and S values. The right two graphs show the KM absorption(K) and sattering (S) coefficients computed from measured reflectance data. Taken from [4].

KM appears to be the most complicated and realistic method currently used to calculate artist paint renderings. Other methods used are a simple Phong model [34], and alpha map gray scale [9].

Metallic Paint

Metallic paints grossly violate one of the basic assumptions stated above: the paint is diffuse only. Generally, metallic paint has a "directionally diffuse" component that is not easily simulated by simply adding a single Phong lobe onto a diffuse model. As mentioned above, metallic paint generally consists of 3 layers: gloss, binder, and substrate layers. Generally, these layers are separated into two parts: the gloss layer and the pigment layer (the directionally diffuse portion of the calculation).

The gloss layer of paint is the first layer that incoming light strikes, and Westlund et al. [83] notes that even a small change in the gloss can have a large effect on our perception of the surface. Westlund creates a virtual gloss meter to equate BRDF parameters to real paint measurement instruments. Then this correspondence is used to apply actual

measurements of the paint to create a metallic paint simulation using a ray tracer.

While this is likely the most accurate method of representing gloss, more interactive methods are used by other authors. Ershov et al. [22][23] use a single calculation to estimate the thickness of the specular lobe created by the gloss layer. Dumont-Becle [20] and Shimizu [73] both use environment map textures to represent the gloss. Dumont-Becle uses a single environment map combined with a bump map to simulate the glossy reflection, while Shimizu used one or more pre-convolved environment maps to estimate the method used by Westlund. All of these methods gain higher interactivity, at the cost of some of the accuracy introduced by Westlund.

After light has passed through the gloss layer, it interacts with the subsurface paint. As with the gloss layer, there are differing ways of both measuring and simulating it. The most complex method used to reproduce the appearance of the subsurface is Ershov et al. [22]. Ershov represents the subsurface as a series of thin paint layers that interact with each other. The layers themselves are assumed to be thin enough such that no scattering effects take place within a single layer. This method creates a correspondence between the actual structure of the paint and its final appearance. Therefore, if an accurate model of the paint structure is input into the system, the final result should look realistic. This method is borderline interactive, with frame rates ranging from 8hz to 0.1hz, depending on how many layers and extra effects (such as gloss and sparkle) are used.

Rather than using the microstructure of the paint itself to drive the simulation, the rest of the authors in this literature review instead use data gathered from measurement instruments. Dumont-Becle uses measurements from a light meter at many different angles (over 100 measurements total). This is then used to create a color texture map that represents the contribution to the final color from the subsurface. Westlund and Shimizu both note that experts from the paint industry have verified that a metallic color can be ac-

curately represented from just three measurements by a light meter. These measurements are then used to create a second degree polynomial through LAB color space. Since Westlund uses ray tracing, its implementation is non-interactive. Both Dumont-Becle and Shimizu's methods are interactive, with Dumont-Becle ranging from 1- 28hz depending on how many effects are shown (orange peel, sparkle, gloss). Shimizu's method ranges from 5hz to over 30hz, depending on the size of the final generated image.

A final method worth mentioning is Dorsey et al. [19] which simulates metallic patinas. Although not specifically paint, it uses a modified KM model to calculate the metallic appearance, similar to other paint models discussed. It computes a KM computation for each wavelength. Then, in order to model the directionally diffuse lobe, it attenuates the diffuse KM calculation with a specular color. This is done for multiple surface layers to simulate the entire directionally diffuse lobe. These contributions are summed to create the final color. This method is not interactive.

Spatially Varying Metallic Paint Effects

While the overall color of a metallic paint, as seen from a distance, is generated by the gloss and directionally diffuse subsurface, metallic paint has some micro structure effects that can be seen from closer distances (up to several meters) [23]. The primary one of these effects is the appearance of bright sparkles that seem to appear randomly on the surface of the paint. This effect is caused by the mica and aluminum flakes embedded in the binder layer of the basecoat. If the reflection angle from the light source off of an individual flake happens to align with the viewer direction, then the viewer will see a small "hot spot" in the metallic paint, which usually appears white. These sparkles appear to be random along the surface because there are many flakes inside the paint aligned at slightly different angles, and the odds that the light and viewpoint happen to align with

any individual flake is low [23].

In order to properly model this effect, three different approaches are taken. Becle et al. [20] uses a texture map of random white points to estimate the distribution of flakes. As the distance from the user viewpoint and the paint increases, this noise map is multiplied by a decreasing number (ranging from 1 to 0) to simulate the flakes averaging themselves into the final paint color. Texture coordinate computation is done using both vertex location and lighting/viewpoint angles in order to properly render the view dependant properties of the sparkle.

A more physically accurate method of metallic flake simulation is performed by Ershov et al. [22]. In this method, the actual size, density, and material properties of the metallic flakes are taken into account. Since there are far too many flakes to realistically make calculations for each, for any given lighting/viewing angle, a statistical distribution of flakes are chosen to represent those flakes that are aligned with that angle. Each of these flakes is then assigned a size and exact orientation. After that each flake is treated as a mirror surface, and the visible luminance of the flake, assuming that each flake is less than or equal to the size of a single pixel, is calculated based on the lighting, flake size, and refraction of light through the binder and gloss layers. This contribution is then added to the color of the pixel the flake is in. It should also be noted that in addition to flakes on the "top" layer of the paint, there are also flakes deeper into the subsurface. These are accounted for in the same way as the top flakes, but given smaller lighting weights and densities.

The final method of sparkle simulation is given by Durikovic et al [2], which uses geometrically modeled flakes to represent the sparkles. To perform this, a large mesh of small bumps is added to the existing geometry of the object. This allows the normal rendering and lighting calculations to simulate the effects of the metallic flake. Since the

flakes are physically modeled many considerations, such as the flake sparkle blending into the overall metallic color as the view distance becomes large, are automatically taken into account. This also gives very consistent results, as once the flakes have been created there are no random elements in the simulation. Durikovic [21] also uses stereo pairs of the geometry to create binocular rivalry and give more convincing depth effects than the previously discussed methods. While this method has some attractive aspects (consistency, smooth lighting transitions for a moving viewpoint, stereo effects), it requires the initial creation of the flake geometry, and the authors note that the rendering time of this method is slower than either of the previous methods due to the added geometry.

Another micro structure effect to consider for metallic paints is orange peel. This is the effect of having tiny ripples or imperfections in the the paint. Like the flake sparkle, this effect is only visible from a limited distance. The only author in this review that attempts to simulate this effect is Becle et al. [20]. In this method, a bump map with mildly altering normals is added to create the illusion of the small orange peel bumps. Since proper simulation of orange peel are of interest in the automotive industry, further research has been performed on orange peel, and is presented in Chapter 4.

2.1.2 Simulation of Paint Application

The actual application of the paint can have significant effects on the final appearance. In artistic paint rendering, the actual brush stroke is what causes spatial variation in the appearance of the paint. With metallic and industrial paints, the application method can also cause texture effects such as orange peel, and can also determine the amount of paint necessary to achieve an acceptable appearance: this has both cost saving and environmental considerations (less paint means less raw materials, and less toxic chemicals entering the environment). Thus, the actual application of the paint has also been studied and

simulated, and a summary of relevant research is given in this section. As above, the techniques are broken into artistic and metallic paint simulation.

Artistic Paint

The simulation used to recreate the application of artistic paint is generally done one of two ways:

1. A simulation of the brush physics involved. The exact method varies according to the artistic technique involved.

2. Automatically generated painterly renderings that are created by heuristically placing strokes based on the image and desired paint style.

Chu et al. [9] create a geometric 3D brush model. This model is then deformed using constrained energy minimization. When the user "presses" the brush against the canvas surface, the surface creates a resistance against the brush, causing the brush geometry to deform, as well as depositing ink on the surface. In order to create a convincing user interface, a six degree of freedom tracking device is given, along with a force feedback system to give haptic feedback to the user. The deposition of the ink is then stored as an alpha map to represent the thickness of the ink that has been deposited on the surface. This is given a direct grayscale representation for rendering.

Baxter et al. [3] also gives a force feedback interface system to the user and utilizes a geometric model of the desired brush (see Figure 2.3) which interacts with the surface of the canvas. In addition, it gives a slightly more complicated physical interaction model in which the brush can also pick up paint from the canvas surface as well as vice versa. In this way, wet paint can be "smeared" across the surface. The drying of the paint is also modeled to allow multiple layers of paint to be deposited on top of each other. The actual drying process is just given by the user: the user selects when a layer is "dry", which

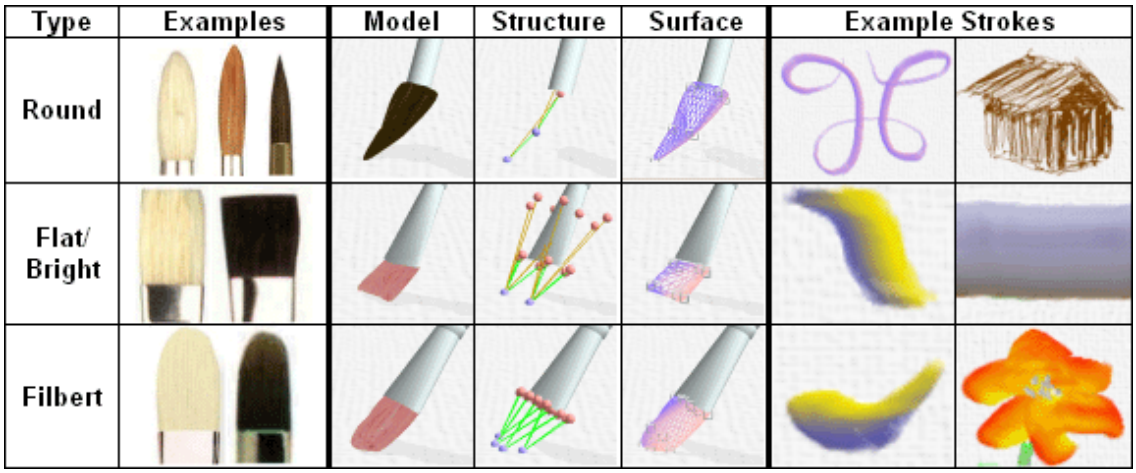

Figure 2.3: Various virtual brushes created using Baxter et al.'s [3] method. A skeleton of the brush is created, and a surface mesh placed over the top of it. When the brush contacts the surface, it deforms, giving differing brush strokes. This method is also used by [9]

prevents that layer from being smeared or deposited back on the brush. This system uses the modified KM method for rendering discussed above. Rudolf et al. [67] uses a similar method to model wax crayons, but uses mouse interaction to move and compress the crayon against the surface.

Curtis et al. [13] performs a full fluid simulation of watercolor paint, with accurate physical interaction between the brush, fluid, and canvas. The water is simulated physically as it runs across the surface, slowly drying and depositing the pigmentation on the canvas. The resulting deposition of paint on the surface is then rendered using the KM method discussed above. This method does not have direct user interaction of the brush strokes, as the physical fluid calculations are too slow. Instead, the user "plans" brush strokes ahead of time on the surface of a pre-existing image, and those brush strokes are used to guide the system in its simulation.

Hertzmann et al. has created several systems that require even less user interaction [34][35][35]. In these programs, the user just inputs "style parameters", which then

Figure 2.4: A completed image generated using Baxter et al.'s method.

guide a fully automatic system that distorts a pre-existing image to make it appear to have been painted (see Figure 2.5). This distortion is done as a series of directional blurs that are made to appear as brush strokes. The allowed parameters include both physically based ones: brush size, stroke length and curvature, and algorithmic ones: approximation threshold (how close the final image must be to the original), color jitter (random color jittering), and blur factor (the size of the blur kernel). This work built off a body of similar work [50][77], and the general techinique is used by many modern photoshop programs such as Corel and Adobe.

Metallic Paint

Examples of metallic paint application are harder to find. Shimizu et al. [73] uses a metallic paint designer to choose the desired color and gloss, but doesn't directly simulate

Figure 2.5: An example of automatic painterly rendering. A series of filters are used over an existing image (top left) in order to simulate the effects of brush strokes. The final result is the bottom-right image. Taken from [36].

the application of the paint. Ershov, Becle, and Durikovic all use statistical methods to simulate the deposition of metallic pigment and flake particles. These simply use a random number generator that gives paint particles aligned in an acceptable distribution. Kim et al. [41] and Yang et al. [87] give simulations for spray painting, which could simulate the deposition of metallic paint, although the algorithms given aren't designed specifically for this purpose.

2.2 Setup and Spray Paint Simulation

Figure 2.1 shows the critical components of the virtual spray paint system. A tracked head mounted display allows the user to navigate around a virtual environment. The user holds a tracked "spray gun" that is used to spray paint objects placed in the environment. The system currently works with any head mounted display and either a magnetic or optical tracking device. An nVis 1280x1024 resolution head visor along with a HiBall optical tracker was used for testing the system.

There are a number of important considerations when designing the framework for the spray paint simulation. First, the simulation must be real time. Any drop below acceptable frame rates could result in incorrect spray distributions on the object, or an artificially reduced spray rate. Either of these could end with the user being improperly trained, ruining the purpose of the system. Second, the system must mimic actual spray painting closely enough that using the virtual system will train the user in using real spray equipment. This means the paint must flow convincingly onto the object in a realistic pattern and rate. Also, the system must paint onto curved surfaces, as many objects of interest to paint are curved.

Finally, the system must display meaningful information back to the user. One possible method of display is to show the user exactly how the spray paint looks when sprayed on the surface. A realistic simulation of how a certain paint looks when sprayed can give the user a good idea of how well he/she is spraying the paint onto the object, and this method obviously has the closest correlation to real spray painting. However, the downside to a photorealistic paint simulation is that it may be impractical to simulate every possible paint one might want to use. An alternative display method is a non-photorealistic representation of paint thickness. This can give meaningful training

information, without requiring the visual characteristics of the exact paint being used.

The following sections describe a solution to the simulation of paint particles being sprayed onto a virtual surface, as well as both photorealistic and non-photorealistic algorithms for the display of the resulting paint job.

2.2.1 Simulation of Paint Particles

A natural method to simulate spray painting is ray casting, because casting a ray toward a virtual surface is very similar to a paint particle striking a real surface. However, since any drop below a real time frame rate could result in improper training, calculating a ray cast for every paint particle is computationally infeasible.

Fortunately, a good compromise between realism and rendering speed can be accomplished by firing fewer rays, and having each ray that strikes the surface spread paint over a reasonable area. Thus, each particle fired from the spray gun is intersected with the virtual object using a ray cast calculation, and "splats" onto the surface, much as a real particle of paint would do. Varying the size of the splat allows more or fewer rays to be used, allowing a balance between realism and rendering speed.

The first step in the spray simulation is to sample the mesh at load time to determine *uv* density: the area of the *uv* triangle (determined by the texture coordinates and texture size) divided by the area of the 3D triangle (determined by the 3D position coordinates). The *uv* density of each triangle is then stored for later use in paint "splatting."

When the spray gun's trigger is pressed, a number of rays are generated, each with an origin of the spray gun's nozzle tip, and a direction chosen within the shape of the spray cone. Currently, the ray directions are randomly distributed within the spray cone, which represents the ideal spray gun's distribution. However, other distributions could

be used if desired. Each ray is tested with the virtual object to determine the intersection location, both in 3D space as well as *uv* texture space using barycentric coordinates [66]:

$$u_{Dir} = uv_1 - uv_0 \tag{2.1}$$

$$v_{Dir} = uv_2 - uv_0 \tag{2.2}$$

where uv_0, uv_1 and uv_2 are the *uv* coordinates of the triangle at each of its vertices. Then:

$$uv_{Tex} = (uv_0 + u_{Dir} * uvBar.u + v_{Dir} * uvBar.v) * TexDensity \tag{2.3}$$

where uv_{Tex} is the final texture coordinate of the ray intersection, *uvBar* is the barycentric coordinates of the ray intersection with the triangle, and $TexDensity$ is the texture density of the triangle, calculated as above. The barycentric coordinates are determined from colliding the rays with the polygon mesh triangles. A precise simulation of gravity on each particle is not necessary as in any normal painting situation high particle speed and close distance of the spray gun to the painted object makes the effects of gravity negligible.

After the precise intersection point has been determined, the paint density on the affected portions of the object must be updated. The paint density is stored as a texture map across the surface of the object. In addition to the precise texture coordinate that each ray hits, a "splat" is performed to affect nearby texels in a circular shape, based on the pre-computed texture density performed above. Splat size is based on a world coordinate area, then translated to a size in texels based on the pre-computed *uv* density (rounded to the nearest texel).

Once it has been determined which texels in the density map should be updated, the precise amount to increase the value of each texel must be calculated. This quantity is the amount of paint represented by the ray multiplied by the percentage of the total splat area that the texel represents. The amount of paint each ray represents is based on many factors, including the total rays being cast, the characteristics of the gun being used, and

the distance from the gun to the object (paint is lost as particles travel from the gun to the object). See Section 2.3 for details on the effects of distance and gun settings on the amount of paint that reaches the object surface.

In the current system, rays are cast 30 times/second and both splat size and the number of rays to be cast are user set parameters. If too few rays are cast and/or the splat size is too small, the visual effect of the spray going onto the surface of the object can become "splotchy." This can be seen in Figure 2.1. The exact number of rays and splat sizes required to prevent this appearance varies with the size of the area the gun is spraying at any given moment, which is a function of the spray gun settings and the distance of the gun from the surface.

The number of rays that can be cast per frame while maintaining 30 frames per second varies with the number of polygons in the paintable object. In practice, an object with 10,000 polygons can be run at about 500 rays per frame on a Pentium 4 2.8 ghz single core processor, while a 10 polygon flat panel model will run at about 2000 rays per frame. The splat size is then scaled appropriately to generate an even appearance on the surface. Generally, the splat size can be kept to just one neighboring texel with acceptable visual results. However, larger splats may be necessary for high polygon count models (high cost ray casting) or large textures (high *uv* density).

Using the above approach, a density map is built up on the virtual object representing the thickness of the paint at each point on that object. This can then be used to give output back to the user on how well the painting has been performed.

2.2.2 Non-Photorealistic Display Algorithm

The first method of user feedback is a non-photorealistic (NPR) display. This method takes the thickness data from the texture map and attempts to visualize it in a manner

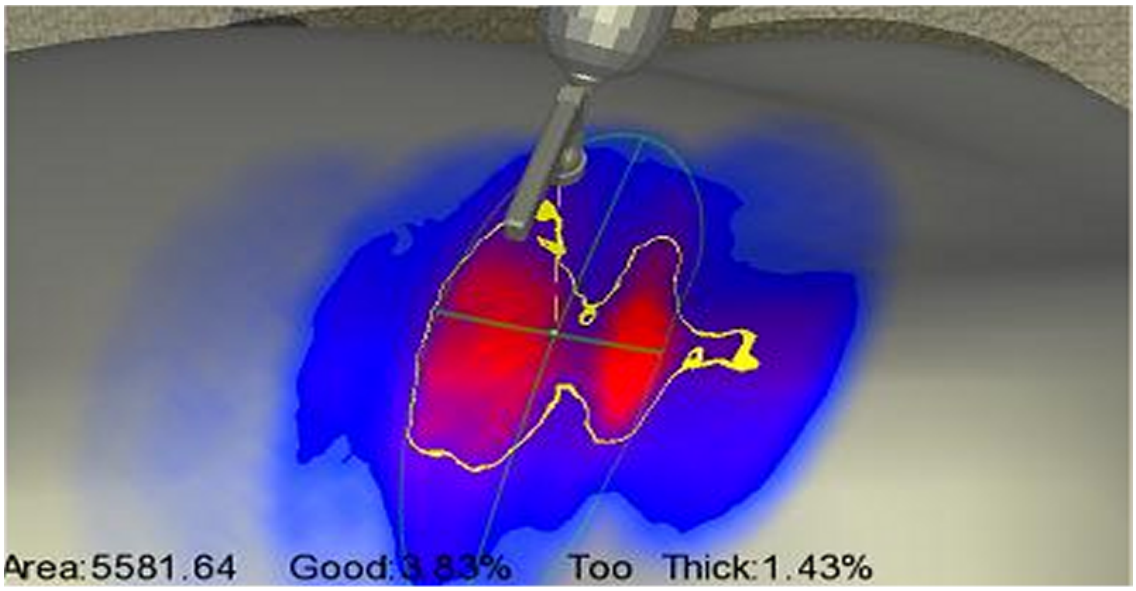

Figure 2.6: An NPR rendering with a shape that has been painted using a range of thicknesses. A cold-hot color visualization scheme has been used to show the user the thickness of the paint.

that allows the user to immediately judge exactly how thick and uniformly the paint has been applied. This is an excellent way to provide training information to new painters, or discover defects in an existing paint procedure.

The algorithm for performing this is relatively simple. A 1D texture of colors is created and passed into the shader. The thickness data that is stored from the paint particle simulator described in Section 2.2.1 is used to index into the 1D texture and retrieve the proper color for that texture pixel. The 1D texture that is currently used in the system is a common cold-hot color scale, ranging from blue to yellow to red as the paint becomes thicker. Areas that are light blue have too little paint, deep blue the correct amount, yellow warns that the paint is about to become too thick, and finally red indicates that too much paint has been applied. The rate at which the thickness moves through these colors can be controlled by a script, allowing users to easily set the proper paint thickness for a particular paint being simulated. See Figure 2.6 for an example of the NPR display

algorithm at work.

2.2.3 Photorealistic Display Algorithm

In addition to an NPR algorithm, a photorealistic rendering algorithm was implemented. This algorithm is a modification of a metallic car paint simulator described in [65]. The metallic paint simulator allows a user to design a paint color, and then displays it on a model using environment map based lighting.

For the spray painting simulation, a couple modifications were made: first, the gloss portion of the simulation is separated from the directionally diffuse color. This allows a user of the virtual system to realistically spray the diffuse color of the metallic paint before applying the final gloss coat, just as a real spray painter does. Second, the simulation was modified to permit the paint to become "lighter" or "darker" based on the thickness of the paint (Beer's law). This allows the paint to appear more realistic as it being applied in real time. A similar effect can be done with the gloss, allowing the surface to appear more or less glossy based on how much gloss has been applied. Figure 2.1 shows this rendering with both a partially complete diffuse coating as well as a fully painted object.

Another important aspect of displaying a realistic simulation to a painter is to place them in a familiar environment. Modern paint shops have paint booths designed to give the painter a lighting setup best suited to showing any defects in the paint job. Therefore, a real paint booth environment has been captured using high dynamic range photographs, which is used as the environment in which to paint (see Figure 2.7). The painted object is also properly lit using this environment [25]. The system also allows users to input their own lighting environments to use if they wish.

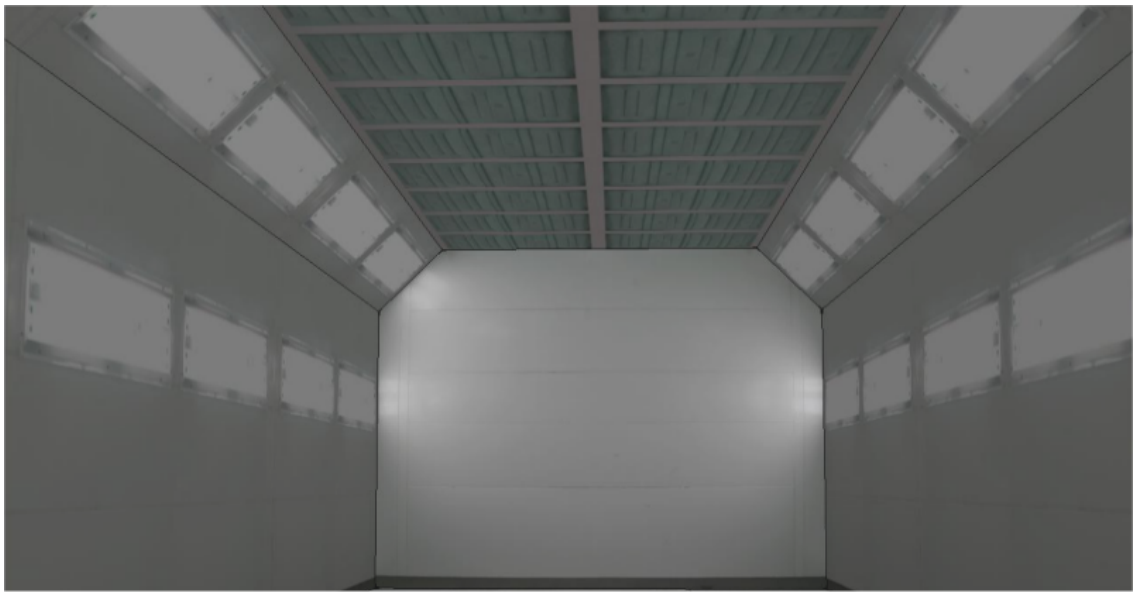

Figure 2.7: The model of the spray booth. Both a model and environment map have been constructed. This allows a painter to use the virtual system in an environment that is familiar to him/her.

2.3 Spray Paint Parameters

Simply allowing a user to spray a surface with a virtual spray gun and observe the resulting paint thickness leaves out a critical fact: not all paints spray onto a surface in the same way. Changing the spray gun can also have a dramatic effect on the application of the paint. Factors such as air pressure, paint pressure, and viscosity of the paint must be taken into account when determining the final appearance of the painted object. For instance, paints with more solid particles (higher viscosity) tend to travel further, resulting in more paint landing on the target compared to a paint with lower viscosity. However, a much higher paint pressure must be applied to achieve the same flow rate with the higher viscosity paint.

The simulation presented in this chapter makes use of research performed by [26]. Kwok did trials of spray painting using differing paint spray gun characteristics. The re-

sulting distribution of paint on a target surface was carefully measured for a variety of variables. By using the results of this study, variables have been added to the simulation: viscosity, A/P ratio (the ratio of air pressure to paint pressure), target distance, paint flow rate, and spray gun cone size can all be controlled by the user with realistic results. For instance, the amount of overspray (paint that misses the target, either due to hitting something else or to evaporation) varies approximately linearly based on the distance of the gun to the target. Table 2.1 shows the effects of a few of the more important variables that are used in the simulation. Figure 2.8 shows how varying the parameters of the spray paint can affect the final visual appearance of the painted object.

One interesting result of the data shown in the table is that paint deposition on the target varies approximately linearly with all three variables. This is beneficial to the simulation, since alterations to any combination of these variables can be calculated in constant time, allowing the user to alter the variables in real time as the simulation continues. The table also shows that all three of these variables have rather strong effects on the amount of paint that lands on the surface.

The use of these variables allows the simulation to be tailored to a particular paint with little effort. A new paint's characteristics can simply be input to the simulation, and

Variable	Value	Paint Deposition (gm)	Overspray (%)
A/P Ratio	0.92	4.14	22.32
A/P Ratio	1.49	3.54	31.69
A/P Ratio	2.18	3.19	39.69
Viscosity(cstk)	57	3.54	31.69
Viscosity(cstk)	106	4.32	25.44
Distance(inches)	7.00	4.59	21.93
Distance(inches)	10.00	3.54	31.69
Distance(inches)	14.00	2.75	45.99

Table 2.1: An excerpt from [Kwok 1991] showing some of the variables that alter the amount of spray deposition that lands on the target. In all cases, the paint flow rate was kept constant at approximately 275cc/min.

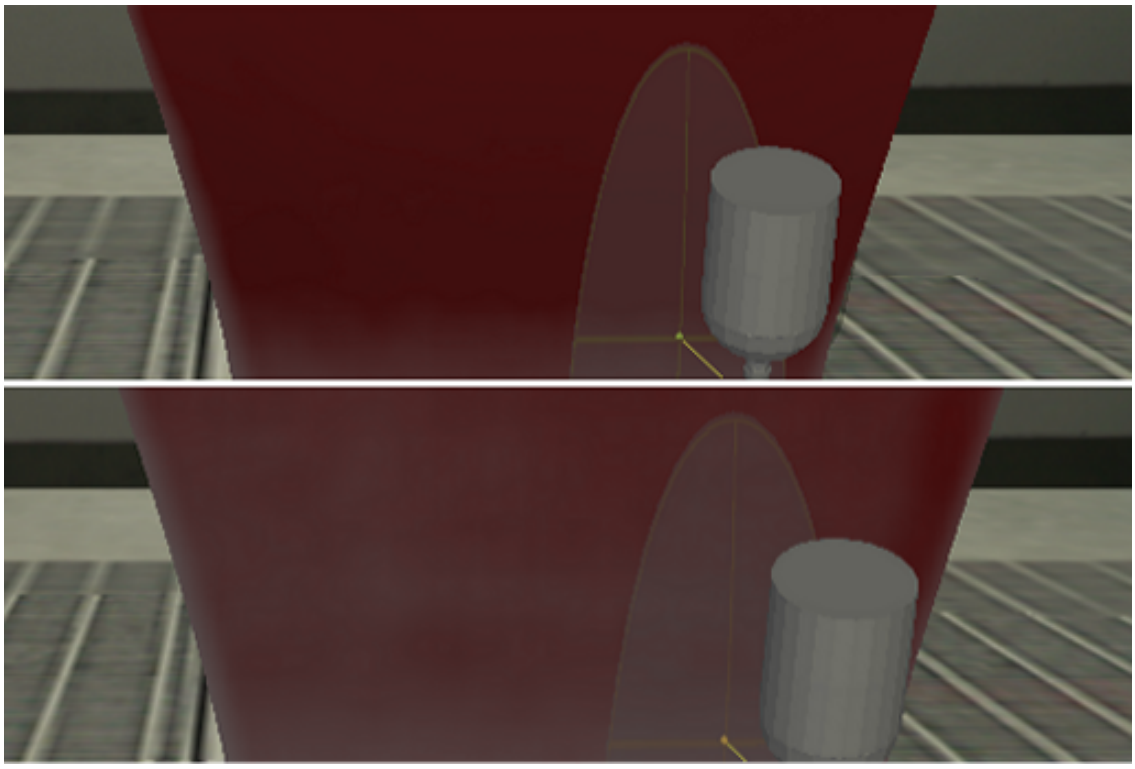

Figure 2.8: Top: An image of a spray object painted with a pre-generated replay. Bottom: The same object painted with the same replay data, but with 50 cstk lower viscosity as well as 0.2 higher A/P ratio. The result is that the object is insufficiently painted due to the higher overspray caused by the parameter changes.

painters can practice painting without wasting large quantities of potentially expensive paint.

One extremely important aspect of spray painting that these variables affect is overspray. Overspray is undesirable for a number of reasons. First, it is a waste of paint, costing the paint company money in materials. Second, stray paint particles can potentially fall on portions of the object or work area that will have to be cleaned later. Finally, overspray lost into the air can become a health and environmental risk.

However, many of the parameters that reduce overspray may also make a paint more difficult to spray. During user testing, when spray painters were asked to make ad-

justments to the spray gun until it sprayed correctly, they tended to adjust the settings in directions that caused greater overspray. A strong advantage to using the virtual training system is that overspray is accurately calculated (utilizing the research performed by Kwok) and displayed back to the user at all times. Therefore, this system provides an effective method for evaluating new spray paints and for reaching a good compromise between ease of use and overspray.

2.4 User Studies

The system has been tested with both controlled experimentation as well as field testing in actual spray paint companies. The controlled experiments consisted of three tests. In all three experiments, each participant was allowed as much time as they wanted to familiarize themselves with the virtual environment, and they were allowed to paint a few test objects before starting the actual experiment. Tests were limited to only a few professional painter participants, as getting enough professional painters for a full statistical study was infeasible. These tests do, however, provide basic verification that the system performs in a similar manner to real spray painting.

Variable	Expert1 (Real Paint)	Expert2 (Virtual Paint)	Novice (Virtual Paint)
Gun Dist (1st Coat)	5in.	6in.	6-10in.
Gun Dist (2nd Coat)	6in.	7in.	N/A
# Passes (1st Coat)	11	10	5
# Passes (2nd Coat)	8	6	0
Time (1st Coat)	33secs	38secs	50secs
Time (2nd Coat)	16secs	13secs	N/A
Correct Coverage (%)	100.0%	97.4%	79.9%

Table 2.2: The first experiment: An expert was tracked spray painting a panel. Then, the same setup was recreated virtually and painted by another (different) expert as well as a novice using the system.

Starting Settings	Painter1 Adjustments	Painter2 Adjustments
60.0% A/P 90.0% Flow Rate	Flow→130.0%	A/P→100% Flow→100%
120.0% A/P 140.0% Flow Rate	A/P→105% Flow→110%	Flow→90.0%
70.0% A/P 110.0% Flow Rate	Flow→125.0% A/P→100.0% Flow→105.0%	Flow→95.0% A/P→100.0%

Table 2.3: The second experiment. Two painters were asked to adjust the settings of the virtual spray gun back to their nominal settings (%100) after they had been altered. The painter adjustments are shown in the order in which the painters made them. Painters generally made adjustments in 10-15% intervals, so any adjustment that ended 85% to 115% was considered to be "close enough" to the original setting. This means all but one of the experiments ended with the painter properly adjusting the gun.

The purpose of the first test was to confirm that an expert spray job on a simple flat panel is similar in both time and technique regardless of whether the painter is using the virtual system or spray painting with real paint. To begin the experiment, an expert spray painter was tracked painting a 25x40 inch panel with real paint at a spray paint facility. Then, expert and novice spray painters were tracked painting the same panel using the system (neither painter was the expert who applied the real paint job to the panel).

Table 2.2 summarizes the results of this test. Both expert painters painted the object in a very similar manner. The virtual spray painter took only slightly longer with a couple less passes, likely due to his gun tip being a bit further from the panel than that of the expert using real paint (which also accounts for fewer passes being made). In addition, both expert spray painters outperformed the novice spray painter.

In the second experiment, the parameters of the virtual spray gun (air/paint paint pressure, and flow rate) were adjusted so that they were different from the nominal settings. Two spray painters (who both had some knowledge of how to correctly set up a spray gun) were asked to adjust the virtual gun back to the original settings, using only

Variable	Expert	Novice
Car Hood % Correct	99.75%	60.0%
Car Hood Time	1min21secs	1min21secs
Car Hood Overspray	52%	40%
Car Hood Gun Dist.	10-11 inches	6-10 inches
Motorcycle % Correct	98.0%	71%
Motorcycle Time	2min8secs	1min58secs
Motorcycle Overspray	44%	57%
Motorcycle Gun Dist.	10-11 inches	12-14 inches (bottom) 4-9 inches (top)

Table 2.4: The third experiment. Two shapes were painted by both an expert painter and a novice, and their performance recorded.

the performance of the virtual gun on the surface of the panel as a guide. The painters made alterations to the gun settings by asking the experimenter to make adjustments to the gun (for instance, "lower flow rate by 15%"). The painters themselves couldn't see the current settings. Each adjustment session ended when the painter felt that the gun was "approximately" the same as its original settings.

Table 2.3 summarizes the results of this test: both painters were quite accurate in diagnosing what parameters had changed in the gun and in adjusting the virtual gun back to its original settings. This demonstrates that the alterations made to the spray gun settings in the virtual simulation were accurate enough to allow spray painters to properly evaluate and adjust the virtual spray gun just as they would a real gun.

In the final experiment, an expert spray painter's performance using the virtual system was compared to that of a novice spray painter. After familiarizing themselves with the virtual spray system, each was asked to paint two objects: a car hood and a motorcycle midsection. Spray time, distance from spray gun to the object, percentage of correct coverage, time, and overspray were all calculated. Each painter was asked to paint the object as completely as possible.

Table 2.4 summarizes the results of this test. As expected, the expert spray painter

performed significantly better than the novice, showing that someone more skilled as a real spray painter performs better using the virtual system as well. Of particular note is that the expert was able to perform a rapid, almost flawless paint job using the virtual system after familiarizing himself for around a half hour with the system.

In addition to controlled testing, the system has been shown to a number of auto refinish and paint companies since its creation. User feedback was positive. A manufacturing company included the system into their training program, with positive results. After the system was introduced to the painters, they have reported reduced error levels and rejection of paint jobs. In addition, use of the system during painter training improved skill acquisition.

Chapter 3

VR Airbrushing

Allowing artists to ply their craft on a computer has always been a major goal within the field of computer graphics. However, while there are programs that permit artists to create virtual artwork and decorate virtual models, far fewer software packages allow artists to do so using skills they learned with real-world tools such as a paint brush or sculpture chisel. For example, most drawing and painting applications available today provide a brush that generates an airbrush spray pattern, but few of these programs work with three dimensional objects and none of them provide a true airbrush interface. This chapter presents an airbrush tool with correct spray paint dynamics and a complete three dimensional interface, and demonstrates how it can be used to produce both traditional airbrush artwork and texture detail for three dimensional models.

The airbrush has several advantages over a traditional paint brush. First, it leaves no brush stroke marks, permitting the creation of a much smoother and more realistic image. Second, the rate of application can be easily varied, allowing for very smooth paint gradients. Finally, the width and density of a single airbrush stroke is very easily varied mid-stroke, allowing the production of artistic effects that are difficult to achieve

Portions of this chapter have appeared in NPAR 09: Non-Photorealistic Animation and Rendering [44]

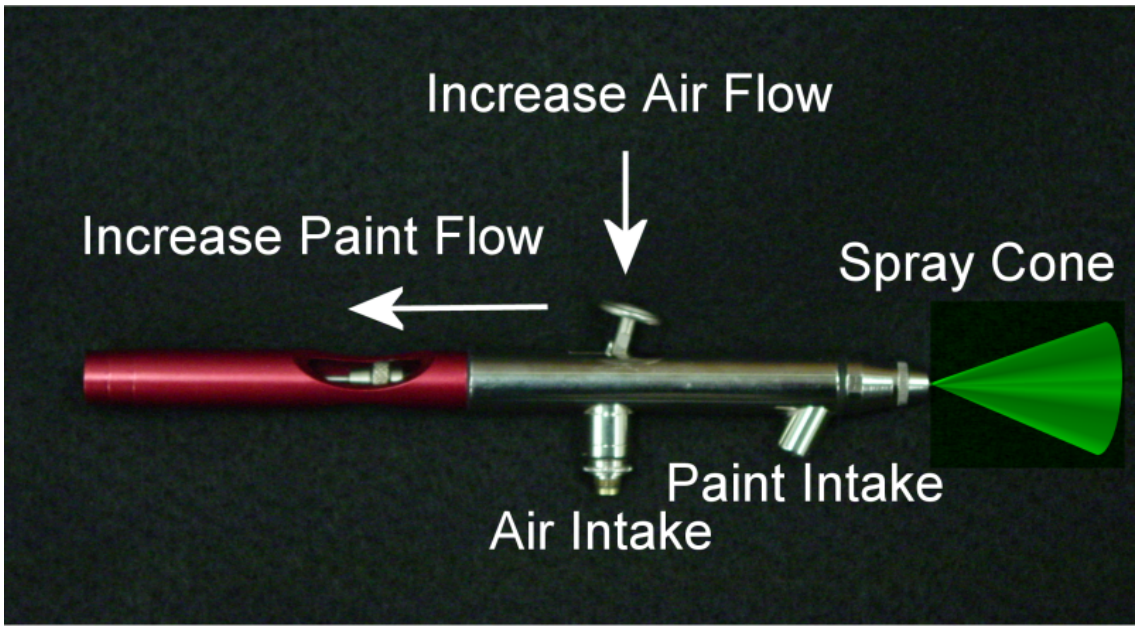

Figure 3.1: A typical airbrush. Pulling back on the trigger increases paint flow. Pushing down increases air flow. Both paint spray area and thickness is altered when moving the brush closer or further from the surface

with a bristle brush (see the fade away stroke in Figure 3.2 for an example of such effects). These unique properties make the airbrush a commonly used tool in the artistic and design worlds. While most photo-retouching is now done using programs such as PhotoShop, the airbrush is still employed by commercial artists to produce architectural renderings [18], medical illustrations [57], advertising signs, and automotive graphics. Fine artists use the airbrush to apply ceramic glazes [57], to decorate textiles [54], and to create wall murals. Model makers add detail to a wide variety of objects including train dioramas [6], costume masks, stuffed animals [58], wax figures, and fish lures.

While bristle brush painting has been fairly well researched in computer graphics, airbrush has not. Computerized airbrushing has several differences with simulation of bristle brushes. First, no haptics are needed to perform a proper simulation. While the feel of a bristle brush against the surface of the painted object is an important feedback

device in bristle painting, an airbrush artist works almost entirely via visual feedback. This significantly lessens the equipment requirement for full airbrush simulation versus brush simulation. Second, most of the potential advantages a real airbrush has over a bristle brush still apply: brush stroke size and rate of application are easily varied, and a smooth final appearance is achieved - brush bristles and resulting brush patterns on the canvas need not be simulated. Airbrushing also stands as a prime example of a true three dimensional real world interface. Studying its success as an interface could provide interesting insights into virtual interface design.

In this chapter, an airbrush simulation tool is introduced that closely replicates the process of real airbrushing. This is done not only to give automotive painters the ability to train themselves with the tool, but also to give a virtual version of the real world tool, as [4] does for paint brushing. The virtual airbrush can also be used as a new way to decal virtual models and supplement tools such as Photoshop for texture map creation. In order to create as realistic a simulation as possible, many aspects of airbrushing are explored. First, the spray particle simulation given in the previous chapter is optimized for airbrush simulation. Second, a Kubelka-Munk paint simulation is provided for accurate rendering of both wet and dry paint layers as they are sprayed onto the canvas in real time. Third, a realistic airbrush interface is created which closely mimics what a real airbrush looks and feels like. Fourth, interface/artistic tools which airbrush artists use are added into the system, such as true three dimensional stencils and easy paint color selection. Finally, display of the canvas or model, as it is painted, is provided in real time. The result is a WYSIWYG system which allows existing airbrush artists to create virtual artwork as well as non-photorealistic model texture materials. This system has been evaluated by professional airbrush artists, and some of the resulting artwork created from those trials is presented.

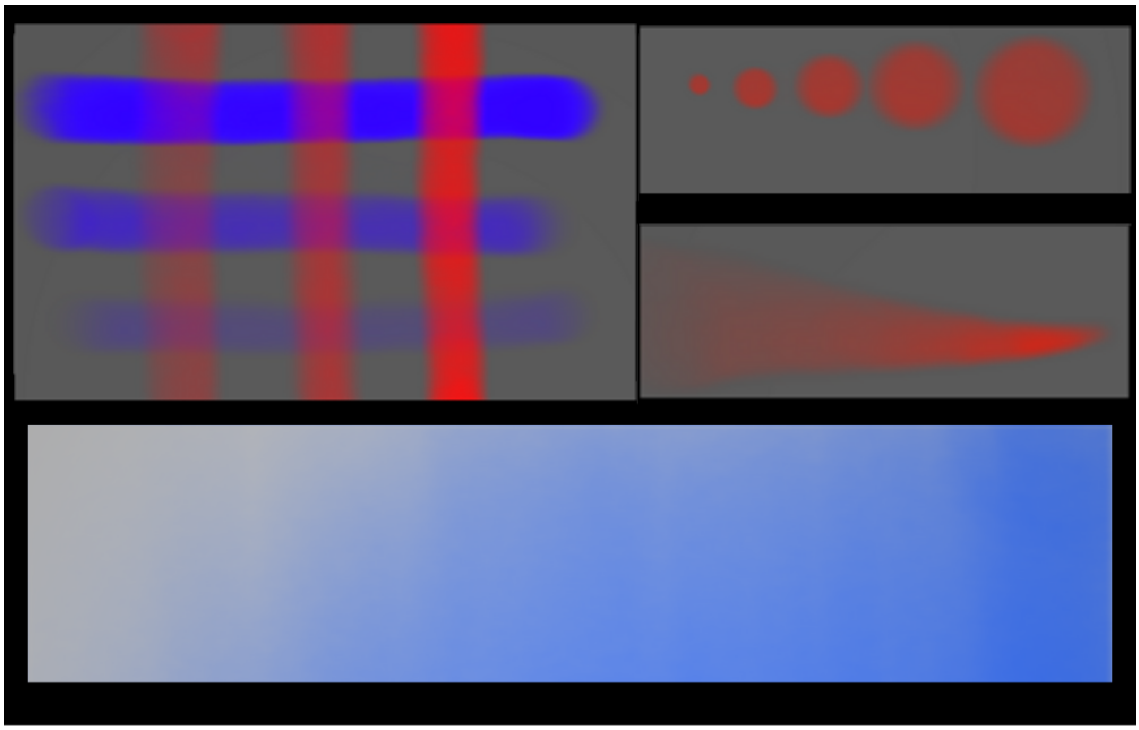

Figure 3.2: A range of various brush strokes and beginner exercises. Top-Left: Red lines of differing paint thickness are drawn over blue lines, created by varying how far the trigger is pulled back. This also shows the Kubelka-Munk paint mixing. Top-Right: Dots of differing size, but similar thickness, obtained by pulling back from the canvas, but varying the trigger. Top-Right: A fade away stroke, obtained by pulling away from the canvas while moving the brush. Bottom: A shading exercise creating a smooth transition from dark to light paint thickness.

3.1 Airbrush Mechanics and Interface

Mechanically, an airbrush is similar to a standard spray paint gun, except that it generates much finer spray particles and provides more precise control over the flow rate. Figure 3.1 shows a typical airbrush. Pulling back on the airbrush trigger increases flow rate, while pushing down on it increases air flow rate. Typically, air flow and paint flow are used to control the rate at which paint builds up on the surface. The distance of the gun to the target surface is altered to change the radius of the resulting dot of paint that lands on the

canvas: closer makes a smaller dot.

Through the manipulation of air flow, paint flow, and distance an airbrush artist is able to create a wide range of brush stroke widths and thicknesses (see Figure 3.2). There are several advantages over typical brush painting: first, an airbrush can rapidly and widely vary the width of the stroke size, even in a single brush stroke. This is very difficult to achieve with a normal paint brush. Second, no actual brush marks are left: the paint lands on the surface almost perfectly smoothly. This allows for a much smoother and more realistic result from using an airbrush over normal brushes. Finally, the rate of paint flow can be easily varied, allowing both small and large pieces of artwork to be created with a single tool.

The biggest disadvantage of an airbrush is that it is more difficult to create a very thin line with the airbrush: most airbrush artists keep a fine tip paint brush around to paint detail work such as eyelashes on a person's face. Also, it takes more effort to switch paint colors with an airbrush: many airbrush artists use multiple airbrushes so they can rapidly swap between desired colors. It is worth noting, however, that both these disadvantages can be overcome with a computer simulation: color selection is easily made in the program simply by pointing the airbrush at a color palette, and objects can be arbitrarily scaled for varying levels of detail.

One of the major differences between simulation of an airbrush versus simulating a normal paintbrush is that a real airbrush provides almost no haptic feedback: therefore a haptic device is not required to simulate the system. However, some kind of six degree of freedom tracker is required. The current system uses a PCI Bird magnetic tracker, although a cheaper six degree of freedom tracker such as a Wii remote could work as well. In addition to tracking the airbrush, the system described in this chapter also tracks both the object being painted as well as the stencil being used so that the user may easily

place both into any desired position.

In addition to tracking, the airbrush itself must be simulated with a realistic feeling trigger. Figure 3.3 displays the electronic airbrush. A real airbrush was taken and modified to have an analogue thumbstick, and a magnetic tracker was added to the back end. One nicety of simulating an airbrush is that the cording required for these two devices is not unexpected by an airbrush artist, as all airbrushes have a cord hooked up to an air compressor. Therefore, the weight and feel of the electronic airbrush is very similar to that of a real airbrush.

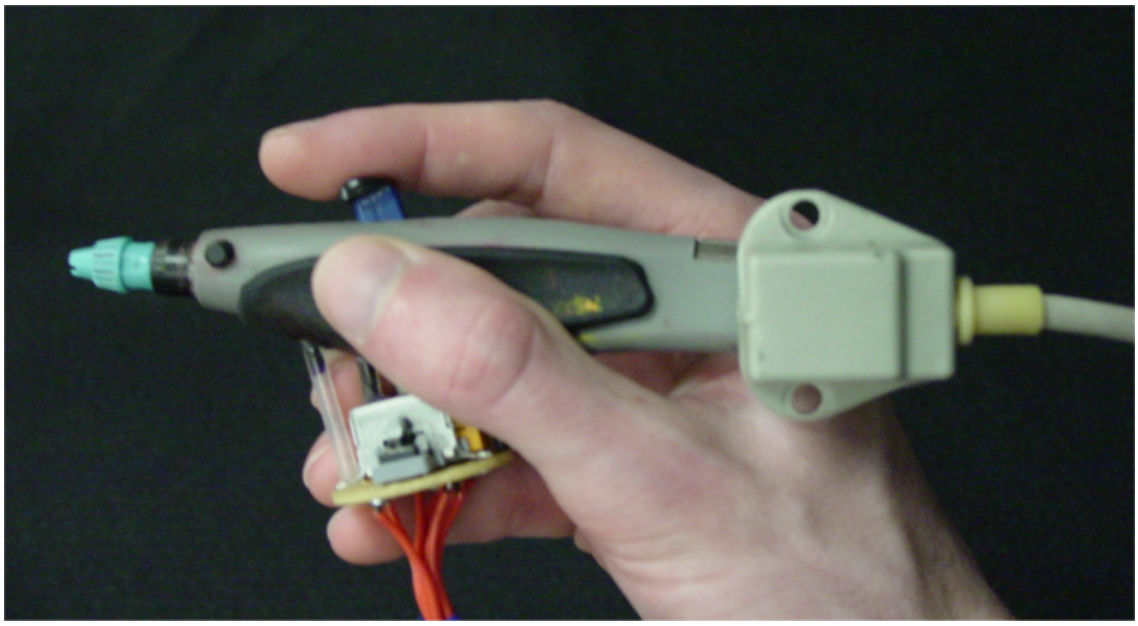

Figure 3.3: The electronic airbrush. A real airbrush was hollowed out and the trigger replaced with an analogue trigger. A magnetic tracker sensor is also attached to the right end. This allows a user to hold a real airbrush, and obtain the data from the airbrushes movements on the computer. Note that real airbrushes also have a hose attached to them for airflow: therefore the electronic airbrush actually has very similar bulk to a real airbrush.

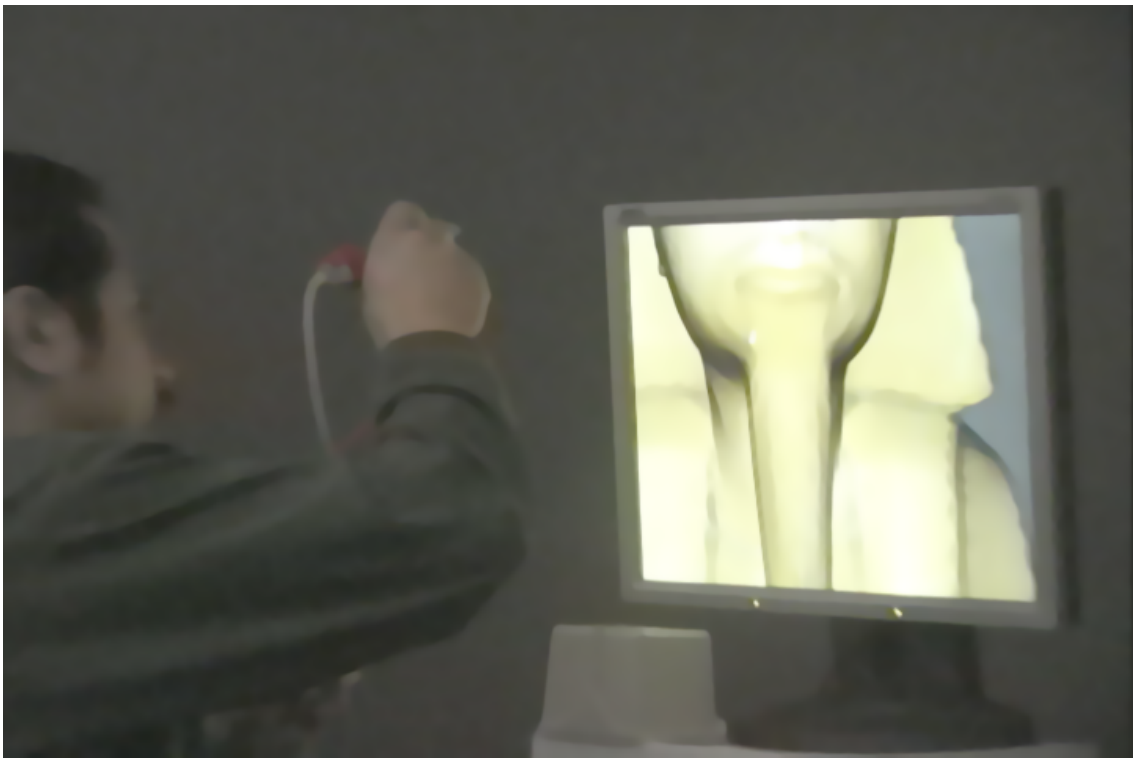

Figure 3.4: A picture of the system in use. An airbrush artist (Gustavo Lira) is shown using the airbrush to paint a 3D model. The tracking emitter is positioned below the screen.

3.2 Airbrush Simulation

The airbrush simulation is built on the spray paint particle simulation given in 2.2.1, but includes multiple modifications to allow for the finer control and stencils required by the airbrush. The implementation was optimized and, as described in this section, the following features were added: dynamic splat size determination, automatic frame rate adjustment & optimization, and paint particle & tracking interpolation. The simulation was also integrated with a paint mixing system detailed in Section 5. These new features make a dramatic improvement in the usability of the system.

3.2.1 Spray Paint Particle Simulation

A ray casting algorithm is used to simulate spray particles as they travel from the spray gun. Each particle's origin is chosen as the center of the spray gun nozzle, and a direction is selected within the gun's spray cone. In all of the images given in this chapter, the spray direction is completely random within this cone. However, if desired, a different spray distribution can be chosen according to a specific spray gun's parameters. In order to provide a fast simulation, particles may also be splatted onto the target surface. This allows a wider area to be painted smoothly without having to cast additional rays.

As a pre-computation, the texture density of each triangle is stored in the system, giving a physical area to each texture pixel on the model mesh. When a particle strikes the target mesh, a UV texture coordinate is found using barycentric coordinates (see 2.2.1). Then, a certain density of paint is stored in the texture pixel that was struck, based on a number of variables: distance of the gun to the surface, viscosity of the paint, air and paint pressure setting of the gun, paint flow rate, and UV texture density (see 2.2.3).

3.2.2 Airbrush Modifications

In order to create an airbrush simulation several important modifications were made to the spray paint particle algorithm described in 2.2.1. First, the frame rate has been increased from 30 frames per second to 60 frames per second. This is because in user testing, it was found that airbrush artists move the spray gun much more rapidly than with normal spray painters, requiring a higher frame rate to keep up with the movements (see Section 3.5 for an explanation on the user sensitivity to frame rate). The number of particles per frame are dynamically altered to enforce this 60 frames per second. This is done by polling the current frame rate, and decreasing the particles per frame by 0.5% per frame until at least 60 frames per second is achieved. The splat size is then increased to compensate for the

decrease in particles, if necessary.

Second, the size of the texture map can have significant effects on the simulation. With a higher resolution texture map, more pixels are altered per frame, forcing more pixels to be altered and uploaded to the graphics card per frame. In order to properly simulate stencils (see Section 3.4) a high resolution texture map must be used: the current simulation uses a texture map of either 1024x1024 or 2048x2048 to store the paint density. In order to maintain 60fps with textures of this size, an array of altered pixels is kept for each frame, and only those pixels are uploaded to the graphics card per frame. It is also worth noting that adjusting the texture map size used is equivalent to changing the splat size: doubling the texture size is the same as halving the splat size. Therefore, for the most rapid possible computation, it is more desirable to simply decrease the texture size rather than increase splat size, except in cases where the splat size may be changing within a single painting session (see Section 3.4.1 for a discussion on splat size).

Third, an algorithm to handle tracker latency and frame rate inconsistency has been added. Normally, the tracker is polled at its current position each frame, and a spray cone is generated based on that reading. However, if the user moves the spray gun rapidly, this can result in a splotchy appearance. In order to create a proper, smooth application of paint across the surface, the new system now interpolates the position/orientation of the paint gun between frames and paints everything in between using the following algorithm each frame:

1. Obtain the position and orientation of the tracker of both the last frame and the current frame.

2. Linearly interpolate between the two tracker readings, stepping from the previous frame to the current one.

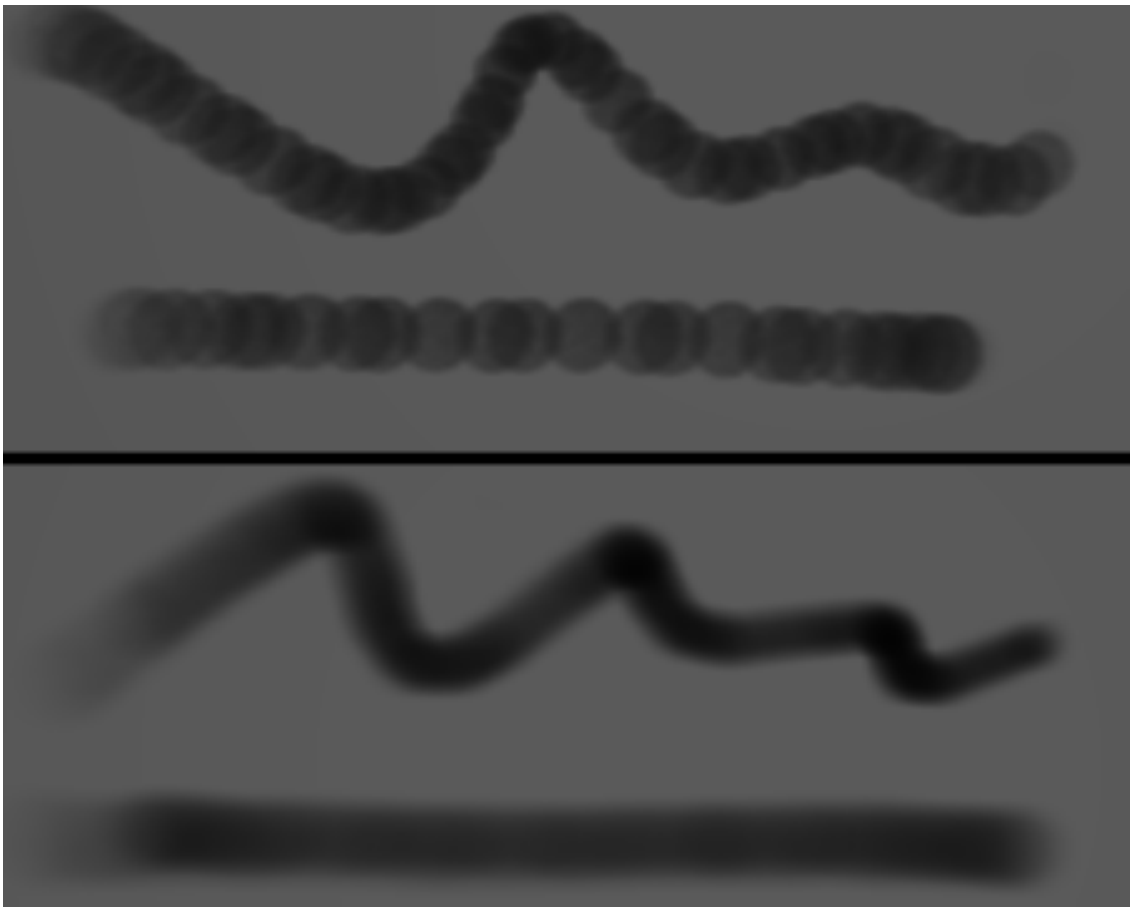

Figure 3.5: An example of the previous spray simulation versus the new one. The simulation has been modified to allow rapid hand movements, and automatically interpolates those movements between frames, allowing a much more smooth final paint appearance.

3. At each step, generate a spray cone and randomly fire paint particles within that cone, as per the usual algorithm.

This generates a much smoother application of the paint on the object than the algorithm given in 2.2.1. See Figure 3.5 for an example of using the old per-frame spray algorithm versus the new interpolation method.

3.2.3 Performance

As stated above, the current system is required to run at 60fps in order to provide a proper interactive experience to an airbrush painter. On a Pentium IV 3.2 Ghz processor with a GeForce quadro 5800 graphics card, the system was able to run with a 2048x2048 texture. The number of particles per frame varied from approximately 20,000+ particles per frame with a 10 polygon model to 4000 particles per frame on a 50,000 polygon model. In each case splat size was adjusted to yield a smooth look to the resulting spray deposition For the 10 polygon model, the optimal splat size was 1 (a single pixel per particle). For the 50,000 polygon model, 9 texels were painted per particle.

3.3 Paint Simulation

Once a deposition of paint has been built up on the texture map, that paint must be simulated. For airbrushing, the paint most commonly dries before the next layer of paint is applied. However, in some cases artists purposely smear the paint to force the paint pigments to mix rather than dry one over the other. Therefore, in order to properly simulate airbrush paints, both realistic mixing of paint pigments as well as layering of those pigments must be performed.

In addition to simulation of wet and dry paint, paint transparency is extremely important. Many airbrushing effects are achieved by layering multiple layers of transparent paint, sometimes mixing in the occasional opaque paint in areas. Therefore, in order to create a fully functional simulation of airbrushing, a very general paint mixing and appearance simulation must be used.

One solution to paint mixing that is currently regarded as quite accurate is Kubelka-Munk (K-M). K-M provides a solution to an arbitrary number of layers of paint as they

are mixed together by finding an absorption (K) and scattering (S) coefficient for each paint being mixed. Haase and Meyer [30] notes that the K/S ratio and spectral reflectance of a paint can be analytically related in the case of complete hiding: when the material underneath the mix of paints cannot be seen. Unfortunately, this does not hold in the case of semi-transparent paints. Since airbrush paints are almost always partially transparent, a more complicated simulation must be used.

The solution used in the airbrush system is similar to that given by [4], which gives a real time solution to K-M paint mixing. In order to perform K-M, they equate spectral reflectance and K/S by setting S to be an arbitrary number. Once these have been found a final reflectance for a mix of paints can be calculated (from [4]):

$$b = \sqrt{(K/S)(K/S+2)} \tag{3.1}$$

$$R = \frac{1}{1+K/S+b\tanh(bSd)} \tag{3.2}$$

$$T = bRsinh(bSd). \tag{3.3}$$

$$R_{tot} = R + \frac{T^2 R_{prev}}{1 - RR_{prev}} \tag{3.4}$$

where d is the thickness of a layer of paint, R and T are the reflectance and transmittance of a layer, and R_{tot} is the final reflectance for a layer on top of previous layers.

If a user has the specific K and S values, the algorithm allows the user to input them for a given paint. Otherwise, S will default to one and a solution found according to the equations above. As in [4], the user may dry a layer, at which point the system calculates the final reflectance of that layer combined with all previous layers, and places that result into a single layer in order to save processing time. In this manner, many paints can be mixed together and shown, all in real time. The system filters the full spectral distribution

and takes eight samples, as performed in [4], although the implementation could be easily extended to include additional wavelengths in the calculation.

The current system performs this calculation per-pixel in the fragment shader. The total number of textures required to simulate K-M is one plus the number of wet layers. Currently, the system keeps four paint density maps to perform this simulation, or a single floating point RGBA texture. However, if more layers of paint are desired, more textures could be added with minimal computation cost, as the current implementation is CPU bound (from the ray cast simulation). Also, only the currently active layer of paint has altered pixels (the paint color that is being currently sprayed), and therefore the amount of information transferred to the graphics card per frame remains constant regardless of the number of layers of paint being represented.

3.4 Computerized Stenciling

The use of stencils is extremely important in actual airbrushing. Various cutouts are taken and put on top of the object to define a certain set shape that the user wishes to paint. Not only are solid stencils used, but also semi-permeable stencils such as cotton balls and feathers are employed to create effects such as clouds and fur. Therefore, the inclusion of stencils is critical for proper airbrush simulation.

Computer graphics lends itself well to both the use and creation of stencils. The airbrush system allows arbitrary 3D objects to be used as stencils, and those objects may be semi-permeable (for instance, a model of a feather could be used as a stencil). This is accomplished through the use of models with an alpha texture placed on them. When a particle of paint is cast, the program checks both the stencil object and canvas object for an intersection. If the canvas object is not struck, the particle is immediately discarded

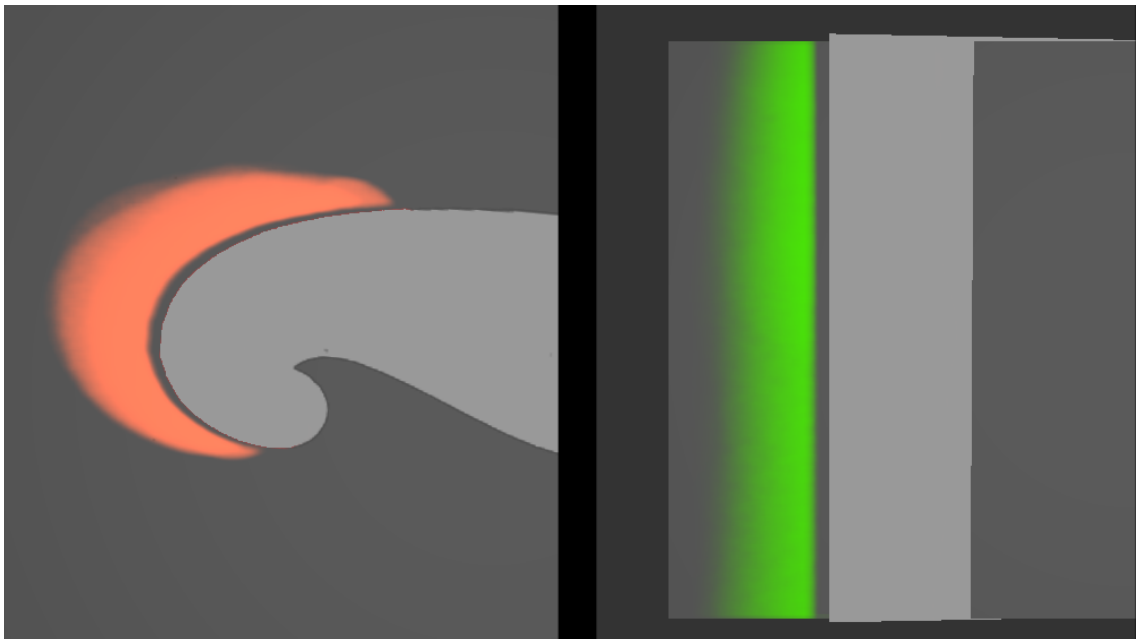

Figure 3.6: Examples of free-hand use of the stencil. Left: A french-curve is used to create a curving line across the surface. Right: A straight stencil is held at an upward angle from the canvas to create a hard edge on one side, but a blurred line on the other side. This is useful for creating many artistic effects.

without checking the stencil. If it is struck, the stencil is also checked for an intersection. If it is not struck, the paint is applied to the canvas normally. If it is also struck, than a texture lookup is performed on the stencil alpha map. The resulting paint deposition on the canvas is reduced by the value of the alpha map, between 0 and 100%.

In this manner, any arbitrary object can be used to stencil out regions of the canvas object. As with the canvas object, the user is given controls to move the stencil into any desired position and lock it into place. In addition to using stencils free-hand, airbrush artists frequently affix the stencil to the surface in order to allow them to precisely mask out the stencil area. See Figure 3.6 for examples of the use of stencils.

In order to achieve this functionality, the current implementation allows the user to projectively texture any stencil onto the surface of the object. The canvas object is given

an extra alpha texture map with the same UV coordinate system as is used for the paint density map. The user then moves the stencil in space to where they want to attach it to the object. The system projects the stencil onto the canvas object from the user's current viewpoint, copying each stencil alpha value onto the object's alpha map. The region of the canvas outside the projected stencil can also be masked out so the user can prevent excess spray from accidently painting unwanted areas of the object. See Figure 3.7 for examples of projective stenciling.

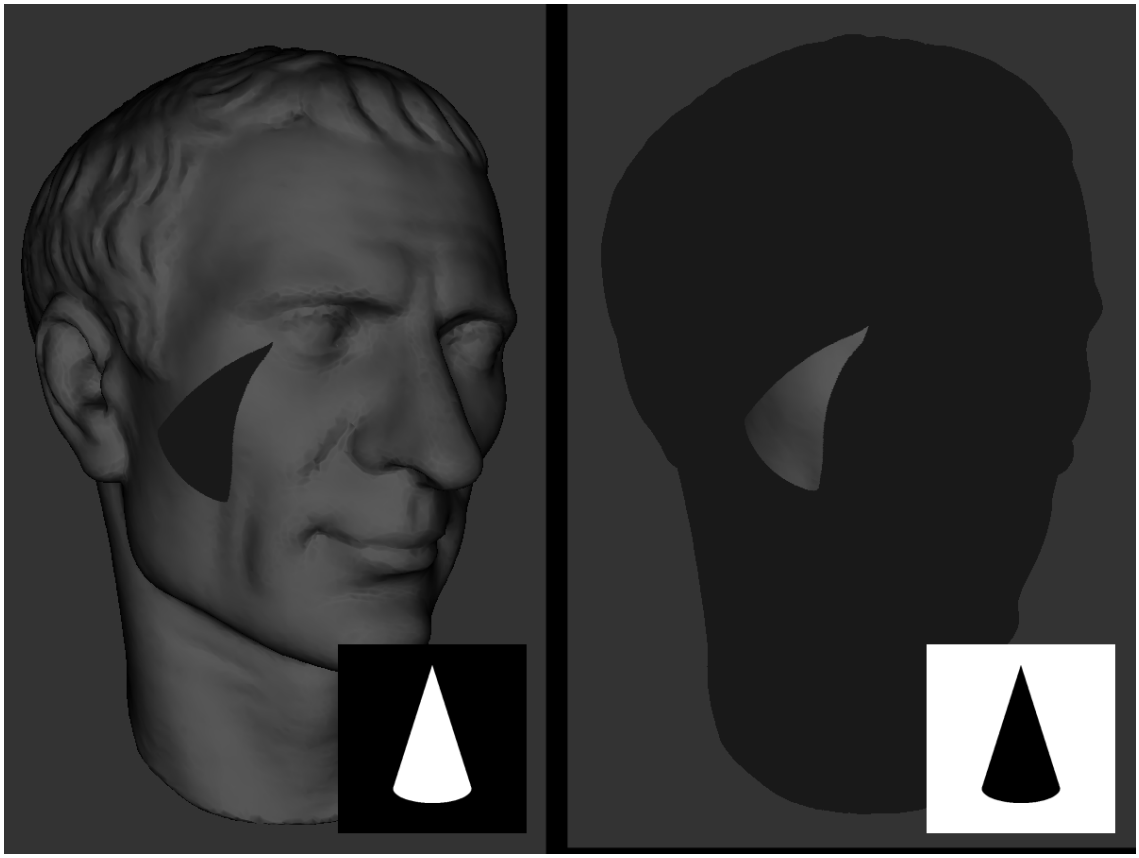

Figure 3.7: Projective texturing is used to attach a stencil to the surface of the object when desired. Left: the stencil affixed to the object. Right: the inverse of the left stencil is affixed to the surface, with the rest of the surface automatically also stenciled out. The stencil used in each case is shown in the lower right corner. Such inverted stencils are frequently used so that artists can create differing images inside and outside of a particular region.

Figure 3.8: Canvas artwork created with the system. The stencils used are shown on the right. Artist: Leah Gall.

The most commonly used stencil is a flat cut out of a desired shape, and is easy to create with a computer: any picture can just be converted into a gray-scale image and used as a stencil. More complicated stencils can be constructed using 3D objects with a custom alpha map laid over them. One interesting possibility is to use the airbrush program itself to create new stencils: the program can be used in conjunction with white paint to create any desired object with an opacity that the user sprays onto it.

To allow the artist to manipulate the stencil in the scene, a tracked, hand held prop is given to the them. The user can then manipulate the prop and see the corresponding movements in the virtual scene. This allows the artist to easily maneuver the stencil to the desired location and then freeze it in space, attach it to the object, or paint freehand. The current system provides no collision detection between the stencil and the canvas object, but users appeared to have no difficulty adjusting to this.

3.4.1 Stencils, Paint Distance, and Splat Size

In order to provide both a real time solution as well as an accurate paint application, splat size must be properly set. As noted in Section 3.2.1 a larger splat size gives a more smooth look and saves computation time, but using smaller splats with more particles gives a more realistic result. While the necessary splat size is largely based on the computation capability of the computer being used, there are also two important run time variables that have a significant impact on the required splat size: the use of stencils and distance from the gun to the canvas object.

While using a stencil, it is desirable to have as small a splat size as possible. The reason for this is that larger splat sizes can cause fine detail in the stencil to be blurred out of the painted canvas. As the airbrush is pulled further from the canvas, the area which it paints increases rapidly. Therefore, a splat size that previously looked quite smooth when the brush was held close to the canvas may no longer look acceptable when the airbrush is further away.

Figure 3.9: Left: a stenciled out region is blurred out due to paint splats interpolating between texture pixels that should be blocked out by the stencil. Right: The same stenciling is done with the automatic splat size algorithm. The stenciled out region is much sharper.

The solution to these problems is to automatically detect when these situations occur and adjust the splat size accordingly. Therefore, the user currently selects a minimum splat size that looks acceptable to them at a close distance between the brush and the canvas. Then, the algorithm tests the distance each particle traveled between the gun tip and canvas, and increases the splat size as that distance increases. The result is that a smooth look is maintained no matter how close or far away the brush is from the surface. In addition, when a particle intersects the stencil, the splat size is automatically set to the minimum to insure detail is preserved while painting stencils. See Figure 3.9 for an example of how this fixes the blurry stencil problem mentioned above.

3.5 User Feedback and Examples

This section details some of the applications that the computer airbrush simulation is capable of performing. Some of them are typical examples that a real airbrush would be used for, such as artistic painting on canvas. Others use the airbrush to decorate 3D models, showing the potential of the system to aid 3D modelers and artists with decorating their work. Most of the illustrations and examples were created by real airbrush artists brought in to use the system and evaluate it.

3.5.1 User Trials

Several airbrush artists were brought in during the course of development, both to create artwork with the virtual airbrush and to make suggestions for system improvement. The hardware and software were first demonstrated for the artist, and they were then allowed to familiarize themselves with the system. Once an artist felt comfortable using the tools (typically in about 15 to 30 minutes), they were then allowed to take as long as they wanted to paint a piece of artwork. Generally, they were permitted to select their own

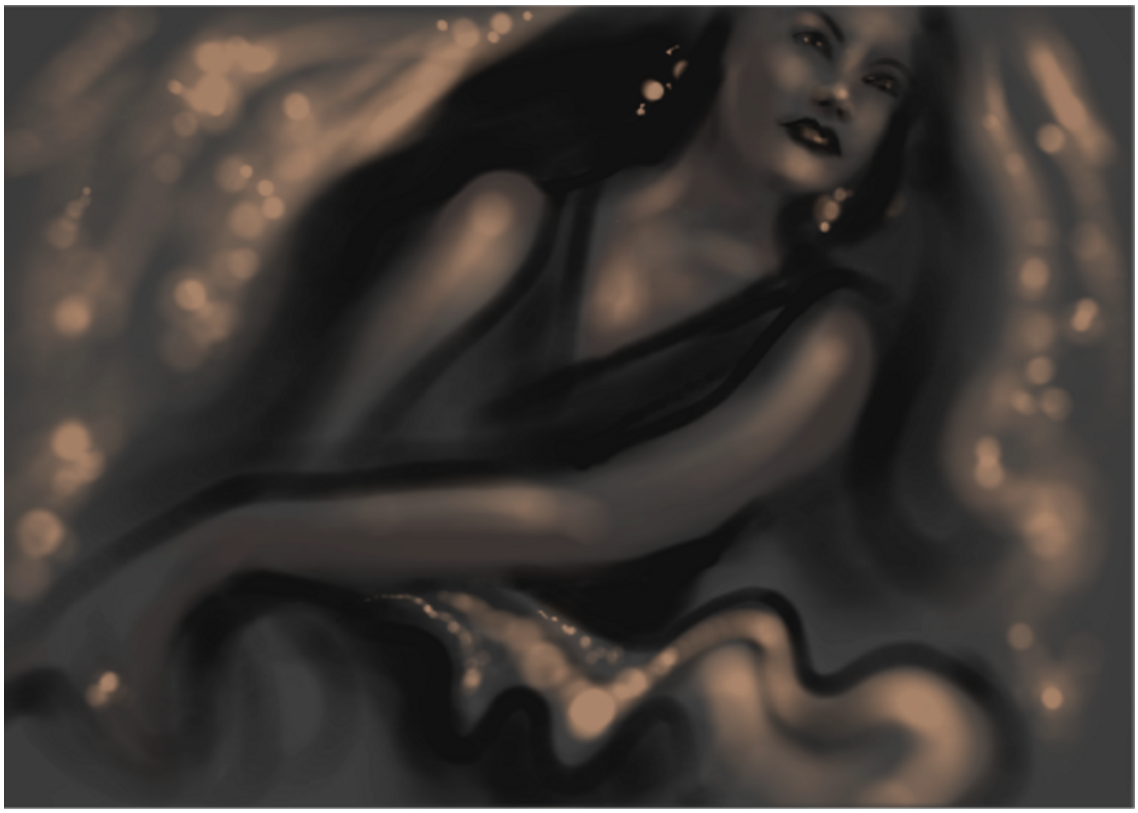

Figure 3.10: Canvas artwork created with the system. Artist: Leah Gall.

subject, although for some of the examples they were asked to paint a specific model or piece of art (such as the model shown in Figure 3.13).

Most of the artwork required less than an hour to make, and none took over two hours. Figures 3.8, 3.11, and 3.13 each required around one hour to create, while Figures 3.12 and 3.10 took under a half hour each. Although no formal timing comparisons were made, all of the artists reported that painting similar artwork with a real airbrush would have required about the same amount of time, not counting time spent waiting for paint to dry and swapping colors (both of these maintenance tasks were significantly sped up by using the virtual system). Simple tasks such as adding the eye detail to the model in Figure 3.13 took only a couple of minutes.

Overall, feedback from artists was quite positive. Most reported that the virtual system felt like a real airbrush after using it for only a few minutes. Some portions of the system that they liked were easy color selection, rapid paint drying, and ease of altering the airbrush gun's parameters. The ability to arbitrarily zoom in on a surface was also noted as being very nice, as detail work not normally possible with an airbrush could be performed quite easily with the virtual system. The mock electronic airbrush (shown in Figure 3.3) was also remarked as feeling very similar to a real airbrush, which the artists liked.

The primary complaint about the system was the tracker latency (as well as the low frame rate in early versions). Airbrush artists were sensitive to any latency in the system, and even in the latest version artists reported that they slowed their brush strokes down slightly in order to compensate for the tracker latency. This accounted for the majority of the reported training time. However, artists stated that, in the end, the slowed strokes did not significantly impact their ability to make the artwork.

Two other qualitative outcomes from the user tests are also worth noting. First, correct reproduction of feathering effects at the end of an airbrush stroke is critical for successful airbrush simulation. Several of the artists focused on this aspect of the system when providing us with feedback, and the improved interpolation techniques described in Section 4.2 were, in part, the result. Second, artists were interrupted by the calculation time necessary to affix the stencil to an object surface (as shown in Figure 3.7), which currently takes a few seconds. When creating rapid stencil effects such as that shown in Figure 3.12, this slowed the artist more than using a real stencil. This calculation could be sped up in future versions for better interactivity.

Figure 3.11: Canvas artwork created with the system. Artist: Marc Dahlin.

3.5.2 Artwork

The primary use of the system is to allow an airbrush artist to paint using a computer system in the same way that they would paint on a real canvas or a real object. This work parallels the research done by [3] for bristle brushes, only for the airbrush.

This is the baseline application for this system, and it incorporates the simulation of spray particles, K-M mixing, and stencils mentioned above. Paint can be stenciled in, free handed, or both as the artist desires. This offers an artist all the benefits they normally receive from using an airbrush along with the advantages of using a computer: undo, zoom, easy stencil creation/loading, easy color selection, instant paint drying and mixing, and no cleanup. Figures 3.8, 3.11, and 3.10 all give examples of artwork created using the system by a real airbrush artist.

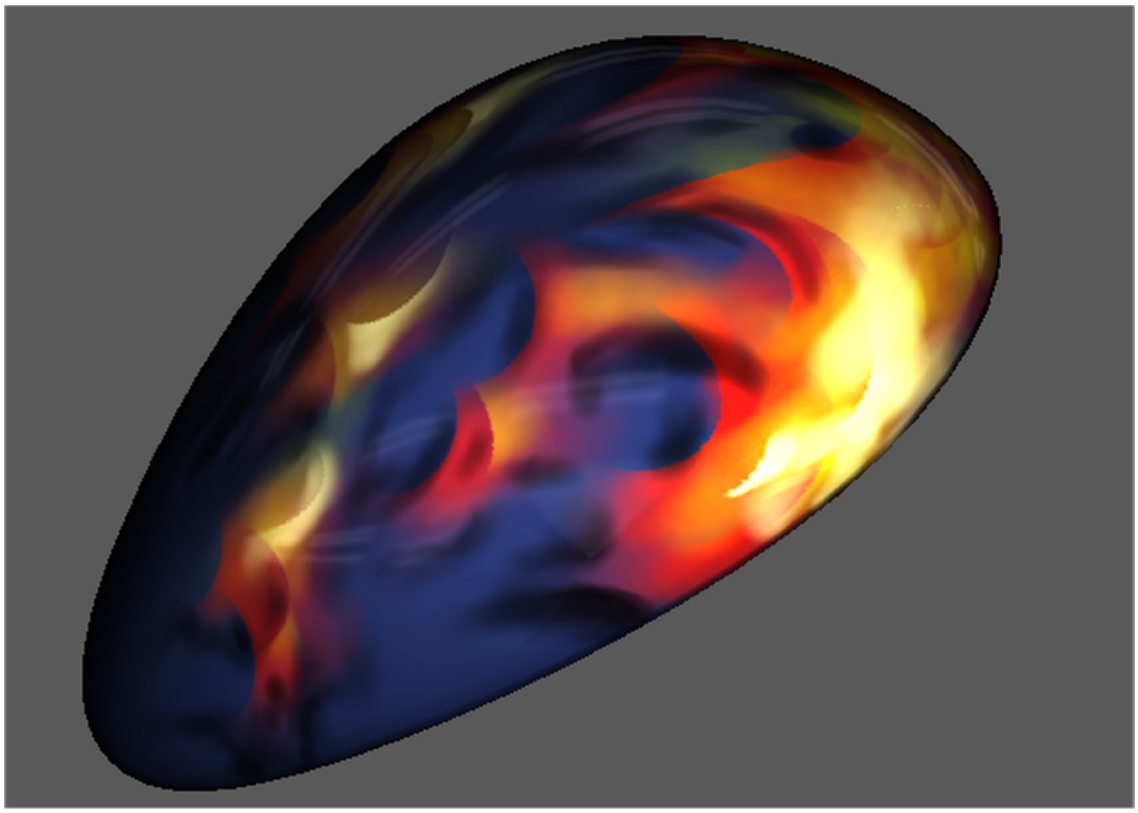

Figure 3.12: Flames are drawn on a motorcycle gas tank, and a gloss coat added. Artist: Leah Gall.

In addition to artwork created on a flat canvas, airbrushes are commonly used on 3D objects such as ceramics and automobiles. Figure 3.12 shows such a piece of artwork. Thus, the system can be used to prototype and/or practice artwork intended to be painted later with real airbrushes, as well as create pieces of virtual artwork.

3.5.3 Modeling

The final application for the airbrush simulation is computer model decoration. The airbrush tool allows artists to paint arbitrary 3D models with no training requirement other than their existing airbrushing skills. This opens up the possibility of using airbrush artists to paint models for video games and movies.

Figure 3.13: 3D model artwork created with the system. Artist: Leah Gall.

Even for artists who know how to use current graphics packages, airbrushing could provide an alternative method of decorating computer models, which has some advantages. First, it is a direct WYSIYG form of texture creation. The artist always sees exactly how the texture they are creating will look on the model as they create it. Second, it is a true 3D artistic input device: both the amount of paint sprayed on the surface and size of the painted area can be easily adjusted in a single stroke. Layered effects can also be created very rapidly. Finally, combining the concept of computer airbrushing with stencils allows artists to quickly generate artistic effects with the program that may be difficult to produce with current programs.

Figure 3.13 gives an example of using the airbrush to decorate models. An undecorated model is taken and painted by an airbrush artist. Note that detailed effects such as the eyes and facial mole can be very quickly and easily added by artists using the system, allowing pre-existing models to be easily detailed.

Chapter 4

Simulation of Surface Roughness

Quality control of surface finishes requires measurements to be made with special purpose instruments. The primary purpose of these measurements is to determine whether or not a particular sample is acceptable to put out to market, or if modifications need to be made to the surface finish or how it is applied. These instruments are therefore designed to provide an intuitive and accurate method for this determination.

As these measurement instruments become more sophisticated, they begin to not only measure a material's general acceptability, but also give a very accurate characterization of the surface finish's color, gloss, and even small scale geometry. This can aid in making specific corrections to the manufacturing or application process. It also provides an opportunity to the computer graphics community: the accuracy and usability of these instruments can be leveraged in order to make accurate renderings of simulated surfaces.

While many computer graphics techniques for measuring geometry and color require elaborate rigs to take pictures of the material from all possible angles, these industrial instruments instead measure only the portions of the sample that professionals have found are relevant to the final appearance of that material. This has three important advantages over completely characterizing the surface reflection of the sample. First, the

measurement instrument itself can be much less complicated, allowing for a very rapid and simple measurement process that anyone can do with little or no training. Second, by studying the instrument that industry professionals use, a computer graphics simulation can be set up that mimics the measurements made by that instrument. This allows a designer that is not trained in the computer graphics field to make use of a tool with which they are familiar to design and create new prototype colors and materials on a computer. Finally, these instruments are commercially available: they can be purchased "off the shelf" already set up and properly calibrated. A user doesn't need to worry about constructing an elaborate data acquisition gantry or calibrating the instrument against known measurements: this is already done for them.

This research leverages the above advantages of commercially available instruments to create graphical simulations from the Byk-Gardner Wave-Scan^{TM}. This instrument measures orange peel: a rippling across the surface of the paint so named due to its visual similarity to the skin of an orange (although paint orange peel usually has a smaller amplitude and therefore more subtle look than an actual orange). At close viewing distances, these ripples can be directly seen by the eye. At longer distances, the surface changes integrate into each other, causing the reflection on the surface of an object with orange peel to appear more blurry than on an object without the defect. Figures 4.5 and 4.6 illustrate the effects of orange peel. While orange peel is seen in the automotive industry as a defect, in some industries, such as some furniture and upholstery, an orange peel like roughness is purposely added to diffuse light or hide defects such as scratches. Thus, orange peel and similar surface roughness can be designed to both give defect tolerances as well as be purposely added into furniture, automotive interiors and other materials.

The rest of this chapter gives research on both rendering of orange peel as well as the detection of surface roughness under varying viewing conditions and with different

surface reflectance properties. First, an overview of spatially varying reflectance algorithms is given. These algorithms lay the foundation for the rest of the research given on representing orange peel and similar surface effects. Then, information is provided on how the Wave-Scan works. Next, a rendering algorithm is developed based on the measurements given by the Wave-Scan. Finally, this system is used to run user experiments that equate surface reflectance and roughness properties with the ability of humans to detect how the roughness of the surface.

4.1 Relevant Work

Computer simulation of various materials has become quite advanced in recent years. BRDFs, BTFs and BSSRDFs allow for realistic display of solid materials, spatially varying materials, and translucent materials, respectively.

BRDF stands for Bidirectional Reflectance Distribution Function, and is a generalized description of outgoing radiance given incoming light direction irradiance. Figure 4.1 gives a graphical example of two different BRDFs. What this figure depicts is the outgoing radiance (both color and strength) given a normal direction (the direction pointing directly outward from the material surface) and incoming light direction. BRDFs deal with micro-structure effects: that is, the effects that microscopic material properties have on the appearance of an object. They do not deal with such phenomena as occlusion and shadowing. BTF stands for Bidirectional Texture Function, and is effectively a texture of BRDFs that represents the material changes across the surface of an object. This allows materials such as marble or wood grain to be simulated. See Figure 4.2 for an example of a BTF.

There are a number of methods for the storage and simulation of BRDF and BTF

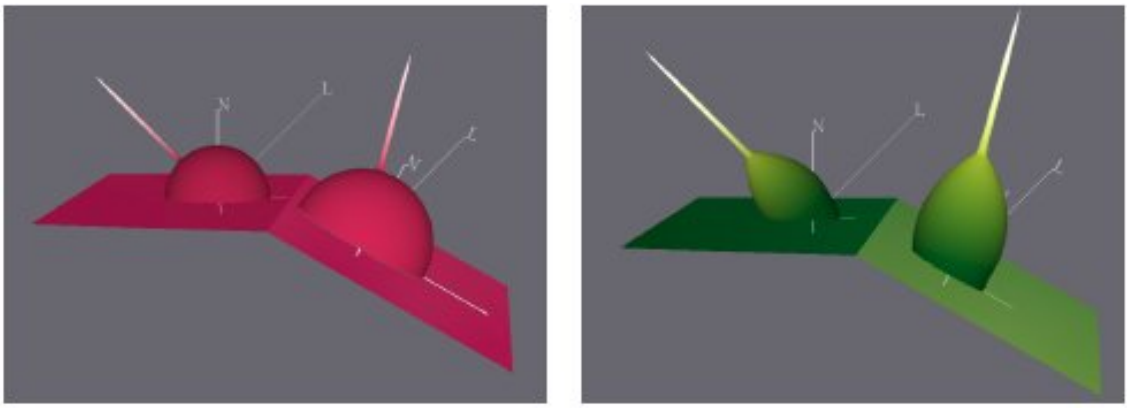

Figure 4.1: An example of a simple Phong BRDF (left), and a more complex metallic BRDF (right). A single Phong lobe cannot simulate the BRDF shown on the right. Taken from [73].

data, and the following sections split the various techniques into three categories: analytic, basis function, and data driven representations. Both analytical techniques and basis function techniques are used in the final representation of the automotive paint and orange peel. For completeness, data driven rendering techniques are also discussed in this section.

4.1.1 Analytic Techniques

An analytic technique uses a closed form solution to a particular BRDF. Inputs, usually a surface normal, incoming lighting direction, surface material attributes, and light source attributes (wavelength spectra for instance), are input into the equation, and a resulting color for that part of the object's surface is given at output.

The most famous and widely used of these is the Phong model [63]. This model sums contributions from ambient (an approximation of secondary scattering of light), diffuse (non-directional light from the surface), and specular (directional light) components. While this calculation is very quick and provides reasonable results for many materials,

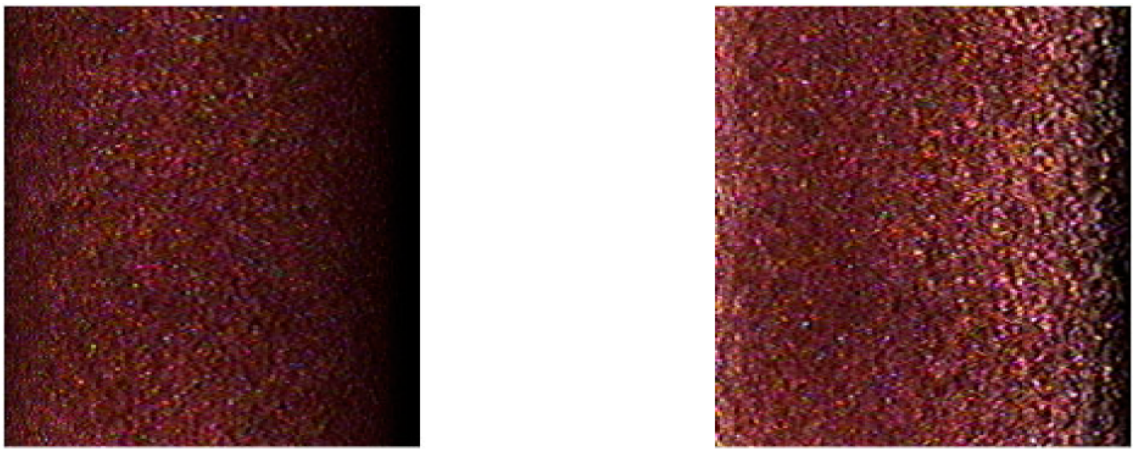

Figure 4.2: An illustration of the difference between a normal 2D texture (left), and a BTF (right). Notice how the lighting characteristics of the material change realistically across the surface the object. Taken from [15]

it is too simplistic for many types of materials. For instance, metallic paints (which are of interest in this thesis) have a directionally diffuse component: the color of the surface shifts across the viewing direction. The Phong model is unable to account for this. Another shortcoming is the model assumes the material is isotropic, a property that is violated by many materials, including brushed metal, hair, and even wood grain. In Figure 4.1, the left image represents a BRDF the Phong model can simulate. The spherical lobe is the diffuse portion of the reflection, and the pointed part is the specular. The right portion of the image shows a BRDF that a single Phong lobe cannot simulate, as the diffuse portion of the calculation isn't independent of the direction.

Three other well known BRDF representations are the Ward [81], Lafortune [47], and Cook-Torrence [10] models. These models were developed to fit measured data. The Ward model uses Gaussians to model anisotropic BRDF data. This model effectively allows "elliptical" BRDFs to be represented instead of the perfectly circular ones the Phong model represents. Lafortune uses non-linear fitting to fit measured data to a series of cosine lobes. The result is capable of representing anisotropic, retroreflective, and off

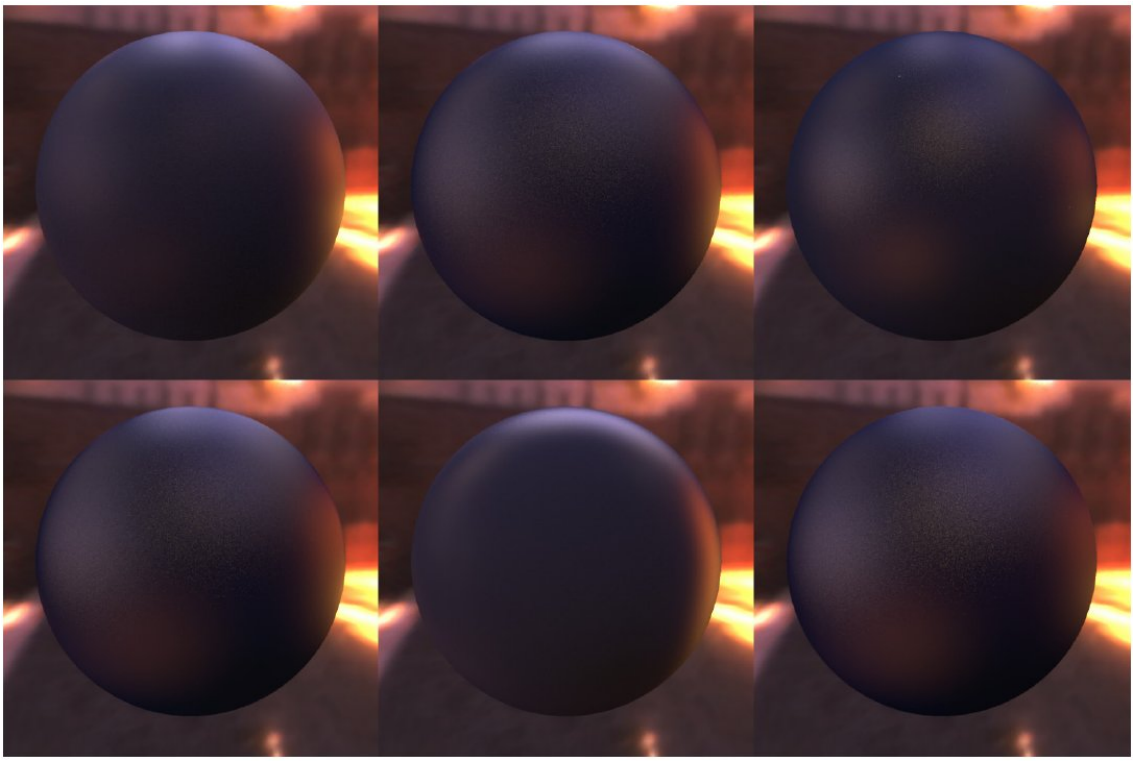

Figure 4.3: A comparison of different analytical BRDF models with a dark blue paint sample. From Top-left to bottom-right: ground truth, Ashikhmin, Blinn-Phong, Cook-Torrence, Lafortune, Ward. From [60]

specular highlight effects. Cook-Torrence is a physically based model that simulates the surface as a series of microfacets. The final equation includes terms for the density and roughness of the facets, along with a Fresnel term.

The result of all of these models is a relatively compact representation of the measured (or generated) reflection data. This can allow for a simple and quick rendering algorithm to provide realistic results. There are, however, some downsides to using a strictly analytical model:

1. It can be difficult, if not impossible, to have the model perfectly fit measured data. In some cases, the reduction of a dense set of measurements to a few parameters can introduce significant error [60]. The result is not guaranteed to have some minimum

desired error. See Figure 4.3 for a visual comparison of various analytical models.

2. Performing the actual reduction of measured data to the parameters can be a computationally expensive process that requires non-linear optimization. This can rarely be done in real time, or even interactively. Thus, there is little if any possibility of measuring the data and showing the results of that measurement to a user interactively. Also, such optimization schemes can be numerically unstable, and therefore sometimes fail altogether.

3. There is no known consistent way to cluster sets of these analytic equations together. This is important for level of detail algorithms. For instance, one can imagine mipmapping a BTF where each pixel in the BTF texture is a Lafortune BRDF, either for level of detail purposes (increasing the viewing distance, for instance), or simply to allow the program to run more quickly. Solving this, however, has no known and consistent solution.

4.1.2 Basis Function Techniques

Another method for representing BRDFs is to consider each particular BRDF as a series of basis functions that add up to the full representation. This allows pre-computed basis functions to be used, and any single BRDF is represented as a series of weights that give the contribution of each basis function in that BRDF. This can allow for a compact representation to a variety of potentially complicated BRDFs.

The downside to this method is that inaccuracy is introduced by the basis functions, and there is the possibility that any particular BRDF may be impossible to fully model using the given basis functions. Thus, it is frequently necessary to tailor the basis functions for a particular class of BRDFs ahead of time. Three representative methods of using basis functions are spherical harmonics [82], wavelets [71] [48], and Phong lobes [73].

4.1.3 Data Driven Techniques

In addition to creating an analytical model from measured or generated data, it is also possible to use the data more directly to represent the BRDF by sampling the data given the viewing direction, surface normal, and lighting direction. Although such direct data sampling was previously computationally infeasible, more recent graphics hardware has enabled such techniques to not only be viable, but usable in real time. As such, more and more recent research has been in this area.

There are two primary portions of data driven techniques: gaining the data to sample (or generating it algorithmically), and effectively sampling/interpolating that data to give convincing results (preferably in real time). For data sampling, the classical method is to use a full goniospectrophotometer to measure the outgoing light from every possible angle for every possible incoming light direction [76] [27]. This generally gives a huge data set of recorded radiances, and takes a long time to acquire. Another method is to enable more data to be acquired at once by placing a mirror around the object that gathers light into a CCD device [81] [7] [56]. This allows many angles of reflectance to be measured at once (potentially an entire hemisphere), leaving only the position of the light source that must be changed. However, this adds some inaccuracy into the measurements, especially at grazing angles [52]. Dana et al. used a single moving camera and light source pair to measure the desired material from all possible viewing and light source positions, and produced a large, open source database of measured materials. Finally, Marschner et al. [52] developed a method for using two cameras (one static and one moving) along with a light source to more rapidly measure and interpolate the data. However, their method only works with isotropic materials that can be placed into a cylindrical shape.

The result of any of these methods is a relatively dense data set of samples. Directly rendering these samples would result in severe aliasing problems, and would likely be

Figure 4.4: An example of lightfield mapping [8]. The images from left to right use more compact, but less accurate, representations.

very slow. Therefore, a method for filtering the data into a usable rendering algorithm must be used. Once again, there have been a variety of methods developed to do this.

The most direct method of using this data is simply to resample the data down to a usable size and use the sampled data to render the material. Chen et al. [8] (see Figure 4.4) takes a dense set of samples, partitions, and factorizes it for efficient rendering. The effective result is a hemisphere of data that is used as a lookup table according to a given reflection direction, and all data points near the given lookup position are sampled in order to give smooth, continuous results. The exact factorization technique they use is non-negative matrix factorization (NMF). Other similar techniques use principal component analysis (PCA) [39] [28] [79], homomorphic factorization [55], and k-means clustering [49]. These all result in more compact representations of the data that can be sampled more efficiently and with less aliasing errors when compared to any sparse sampling.

The work given in this chapter is a combination of a basis function technique with an analytical technique. The underlying paint coating is given by a series of phong lobes, as given by [73]. The top coating is then represented by a single phong lobe perturbed by an analytically generated surface. This surface is created from the output of the wave-

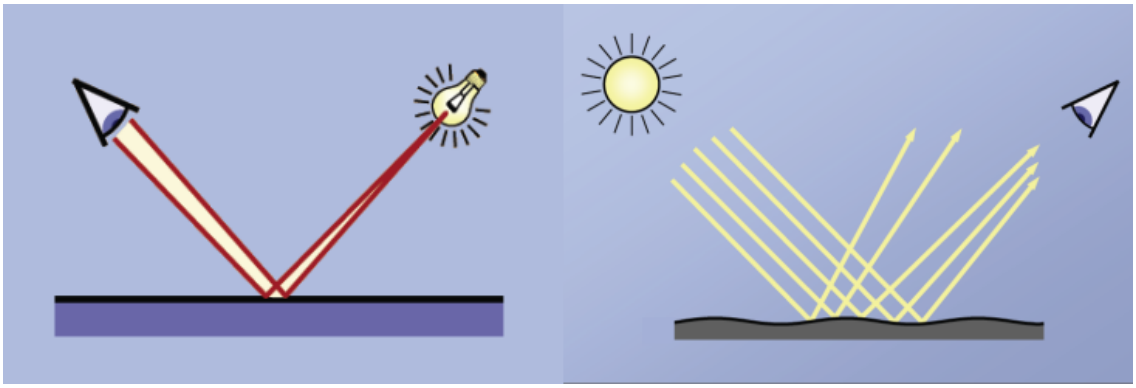

Figure 4.5: An illustration of the difference between a surface with and without orange peel. The surface on the left is a perfect mirror, reflecting the light directly to the eye. The surface on the right has orange peel, causing the reflected rays to be distorted. Taken from [40]

scan device, and represented with a normal map. The combination of the basis function BRDF representation with the spatially varying normal map effectively gives a BTF of the resulting automotive paint.

4.2 Measurement of Orange Peel

Orange peel is common enough in the paint industry that special devices have been created to measure it, and determine if any particular paint job has an "acceptable" level of orange peel. The Byk-Gardner wave-scan [40] measures orange peel by running a laser over the surface of the paint, and measures the reflection of that laser off the paint. Ideally, the laser should reflect in the perfect mirror direction off the paint, which causes it to be directly bounced into the center of the photometer in the instrument. However, orange peel will cause the laser to be distorted (by shorter wavelength orange peel) or shifted away from the center of the photometer (for longer wavelength orange peel). See Figure 4.7 for an illustration of how the wave-scan makes individual measurements.

The wave-scan records measurements of these distortions as the instrument is run

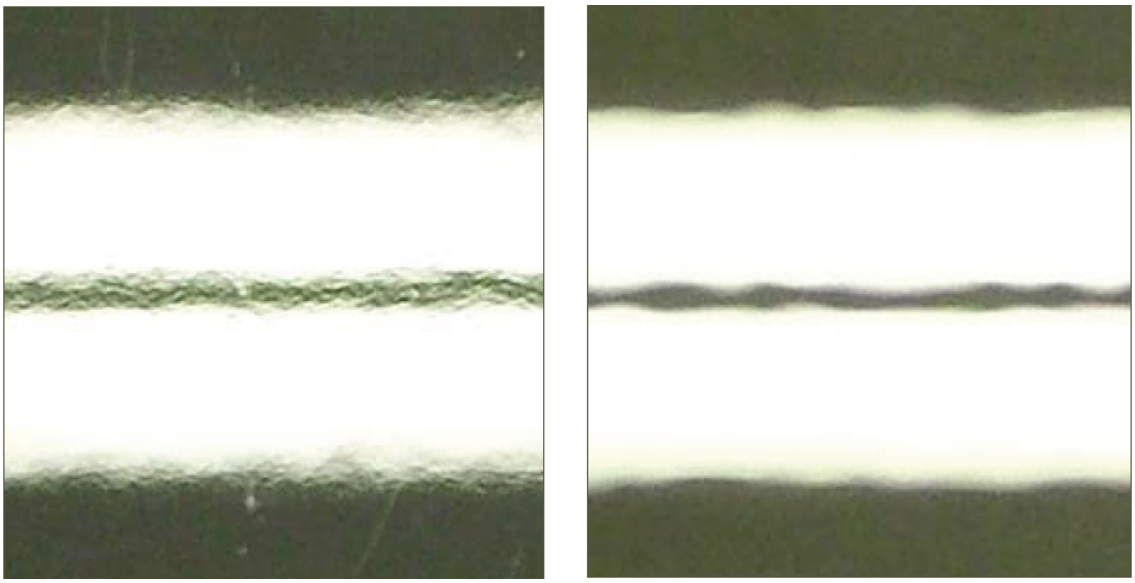

Figure 4.6: A picture of the effects orange peel has on the reflection of a light source. The image on the left is a gloss coat painted over rough steel, and therefore has worse orange peel than the image on the right, which is a gloss coat painted over smoothed steel. Taken from [40]

across the surface of a painted sample. Once a sufficient number of samples have been taken, the surface is characterized by the magnitude of distortions at five different magnitudes: 0.1-0.3mm, 0.3-1mm, 1-3mm, 3-10mm, and 10-30mm (see Figure 4.8). The magnitudes of these five wavelengths are then used to determine if the orange peel on the surface is acceptable or not, as well as possible reasons for what needs to be fixed if the orange peel is not acceptable (for instance, if the 3-10mm amplitude is too high, one possible fix is to spray on a thicker gloss coat). The actual output of the wave-scan instrument is five numbers ranging from 1-100. These values are scaled such that each step (i.e. 1-2 or 43-44) should provide a perceptually uniform increase in appearance.

The purpose of this simulation is to effectively reverse-engineer what this device does: use the five wavelength amplitudes to construct a visual simulation of what that surface looks like. This is done in three steps. The first is a relatively simple user interface

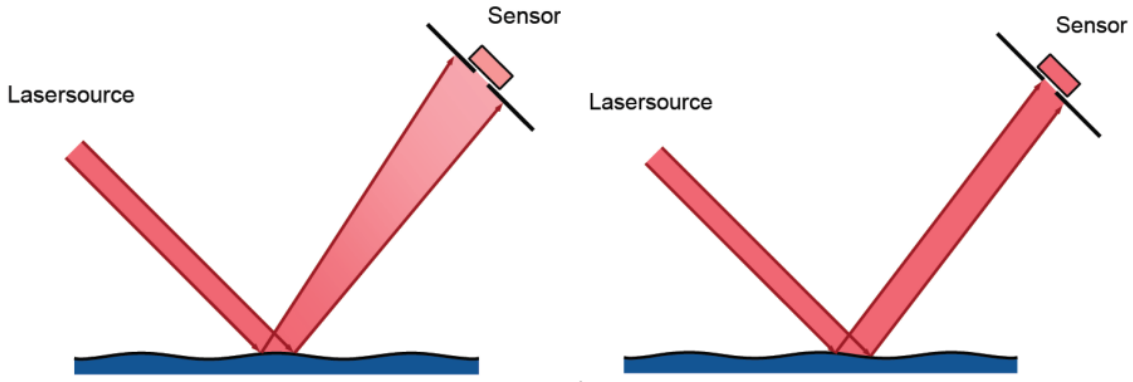

Figure 4.7: An illustration of how the wave-scan instrument takes measurements of the surface profile. The light from a laser strikes the surface, and reflects back into a photo-sensor. Distortions in this reflection are measured to characterize the orange peel. Taken from [40]

that allows the user to enter the five wavelength values, and display them in a similar manner to that shown in 4.8. The second is to take these amplitudes and construct a virtual surface that is representative of the original surface these values were measured from. The final step is to render this surface realistically back to the user.

4.3 User Interface

The user interface provides a method for the user to enter the value of the five wavelengths, and displays the result as a spline-fit curve of the surface profile. This is designed to have an appearance similar to the actual wave-scan graph readings shown in 4.8. In addition to manually inputting the values, the user can directly drag the data points on the display graph. Figure 4.9 shows a screenshot of the provided user interface.

In addition to the wave-scan value input interface, the interface given in [73] is provided to the user. This allows the user to modify the color and glossiness of the paint and view how this affects the visibility of the orange peel.

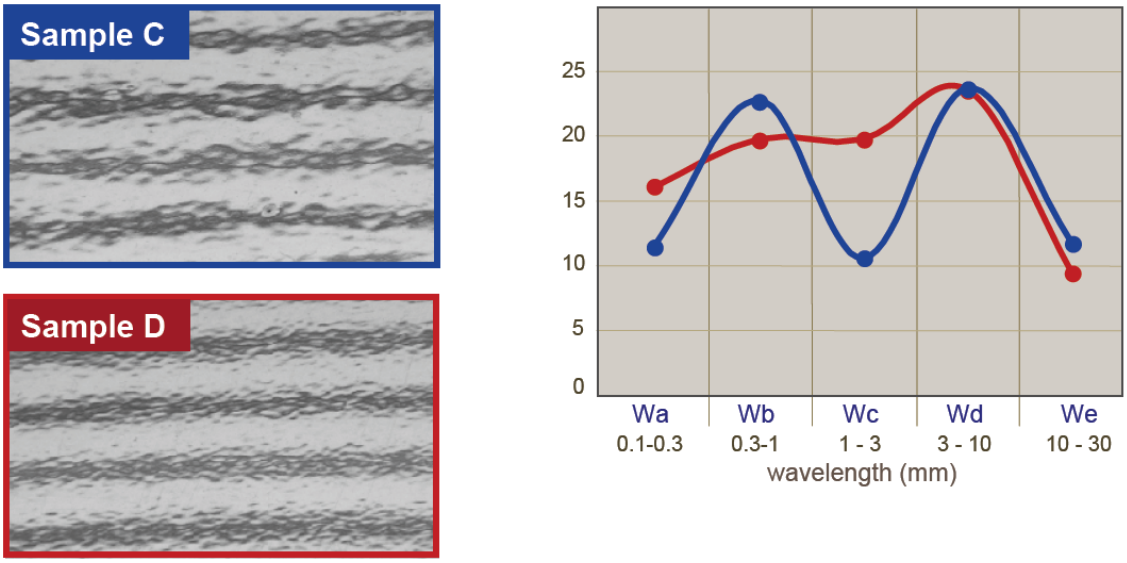

Figure 4.8: A wave-scan graph showing the amplitude profile of the orange peel across varying wavelengths. Taken from [40]

4.4 Surface Construction

Once the five wavelength values have been entered, those values must be converted into a virtual surface that has a similar appearance to the surface originally scanned by the device. This is an under-constrained problem: an infinite set of possible surfaces could have mapped to the same wave-scan values. The information that the wave-scan values gives is the five wavelengths: set at their original values from 0.1-30mm, and their corresponding amplitudes: a number from 1-100. The system must fill in the missing parameters to generate a visually plausible surface.

This first step in this process is to translate the 1-100 numbers to average bump amplitudes, and standard deviation from this average. Byk-Gardner was kind enough to provide us with the conversion equations that changed the wave-scan readings into this number (not discussed here as these equations are the intellectual property of Byk-Gardner). These equations could be reversed to get the average amplitude from the original number.

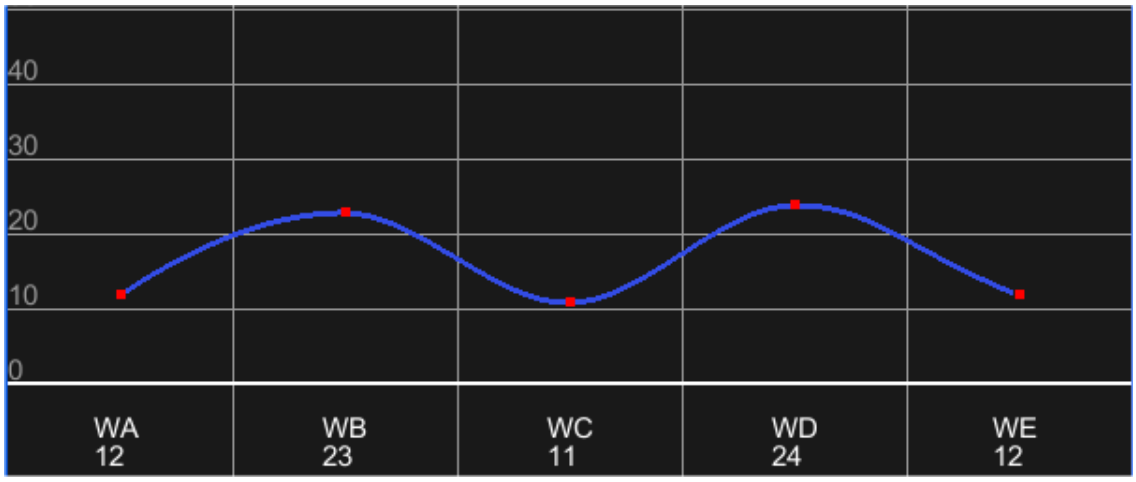

Figure 4.9: An image of the provided user interface for inputting wave-scan values.

The next problem is to generate an actual surface from these parameters. Fortunately, the problem of generating visually plausible surfaces given a sparse set of parameters has already been solved in another area of computer graphics: landscape generation. In particular, the Fourier transform can be used to generate surfaces of a given wavelength and amplitude with convincing results [53]. This method takes a randomly seeded 2D height-field, and applies the Fourier transform to it to convert it to frequency space. Then, a 1/f noise filter is used to convert the random distribution into fractional Brownian motion. Brownian motion is a fractal distribution shown to mimic many natural fractal distributions [29]. After this filter has been applied, the inverse is then used to convert the data back into a spatial representation, resulting in a fractal height field.

This process gives a height field corresponding to a single wavelength value. This process is therefore repeated four more times to create a height field for each of the wave-scan values. These five height fields are then linearly combined to generate a final surface distribution. While this algorithm normally cannot be run in real-time, a height field of a single wavelength can be linearly scaled to create a surface of the same wavelength and bump distribution, but differing amplitude. Therefore, various surface distributions can

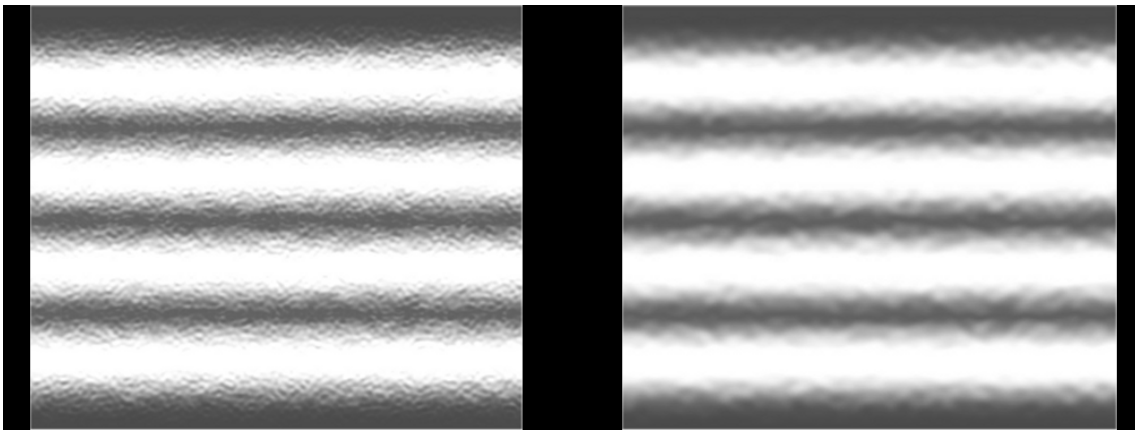

Figure 4.10: On the left is the simple rendering system with high values for the lower wavelengths, similar to the rough steel shown in 4.6. On the right is a sample with low values for low wavelengths, leading to an image closer to the smooth steel.

be pre-computed, then scaled and combined with each other in real time. This allows a user to alter the wave-scan values and have a new surface presented to them interactively.

4.5 Rendering

The final step in the system is to display the orange peel surface to the user. Two rendering methods have been developed. The first is a straightforward rendering that shows the reflection off the orange peel of a uniform artificial light source. This rendering system is meant to replicate the conditions used to take the pictures shown in 4.6. The second system is a more complete paint simulation system, which effectively adds orange peel effects to the metallic paint designed presented by Shimizu et al. [73].

In both systems the height map is generated in the above section is converted into a normal map. Normal mapping is a widely accepted method for real time display of surfaces that do not require full geometric representations. Usually, a normal map consists of three channels: the R 0-255 channel represents X ranging from -1 to 1, G maps to Y, and

Figure 4.11: Two images of the complex rendering system. On the left is an image of orange peel on metallic dark blue paint. On the right is the same orange peel on a less glossy sample of light blue paint, leading to the orange peel being less noticeable

B maps to Z. However, this mapping leads to an unacceptable level of error due to orange peel having very slight variations in normal direction (all orange peel has its Z value at very near 1). This caused portions of the orange peel that should have had different normal directions mapping to the same RGB values in the normal map. Therefore, the map was dynamically rescaled to have 0 represent the lowest possible value in the map (usually -0.1 or even less) and 255 the highest, in order to allow the system to always use the full 0-255 range of values. This change gave the normal map more acceptable results. However, this method may no longer be required, as modern graphics cards are now capable of representing floating point textures with a much higher possible range of values.

The first rendering system uses the normal map to make a reflection calculation to an artificial light source, which is just a series of cylindrical white lights. There is no simulation of any of the diffuse underlying paint: it effectively just renders the gloss coat

of the paint. The second rendering system adds the normal map calculation into the environment map lookups performed by Shimizu's system. This rendering system is similar to that used in Chapters 2 and 3, and is therefore easily incorporated into the airbrush and spray paint simulation. Currently, only the gloss layers of the paint simulation have the orange peel added, although orange peel effects in the underlying diffuse layers could also be estimated using this method. However, this is not done because the wave-scan instrument does not effectively measure diffuse surfaces, as it makes the assumption that the laser light bounces off the surface as if that surface was a mirror: diffuse reflections add error to the calculation. See Figures 4.10 and 4.11 for examples of the two different rendering systems in action.

4.6 Detection of Surface Roughness

The ability to predict if humans detect surface roughness can be beneficial in a number of industries, including the automotive, fabric, interior design, and computer graphics industries. There are two possible motivations for predicting visibility of surface roughness. The first is to design defect tolerances for phenomena such as orange peel. In the automotive area, viewers should ideally see the surface as perfectly smooth. However, in other industrial design applications, such as electronics packaging, adding roughness into a surface can prevent small scratches and normal wear and tear from being seen. Thus, effective design and control of such roughness would be a major benefit to these industries.

How we detect surface roughness is a combination of multiple factors: the lighting environment, surface micro and macro structure, and the human visual system. The software created in the above sections combines a photorealistic lighting environment and physical surface to provide a simulation that humans should perceive in the same way

as a real material sample. This section validates the system as a perceptual tool by both providing a mathematical framework for detection of simple, uniform surface roughness, as well as experimental validation through user testing.

4.6.1 An Analytical Model of Surface Roughness Detection

This section presents a simple analytical model of surface roughness detection based on previous mathematical and perceptual studies. In order to simplify the factors involved, certain assumptions are made:

1. The surface has bumps of a single, repeating amplitude and wavelength.
2. The lighting environment is composed of only two neutral colored lights (light and dark).
3. The surface reflectance properties are modeled using the Phong reflection model (a single diffuse layer and a single specular layer).

This simple lighting and surface model is combined with previous research on human perception of contrast sensitivity and brightness [14] in order to predict if a viewer will be able to detect that a surface is or is not perfectly uniform.

The first step in this algorithm is to determine the difference in reflectance between points on the surface. In order to obtain the largest difference in perceived lightness, only two points need to be considered: the point that reflects the most light into the viewers eye, and the point that reflects the least. Assuming the user is viewing the surface at a 90 degree angle (perpendicular), these two points will be at the peak amplitude and halfway between that peak and the trough. These equate to the surface normal pointing directly toward the eye, and the normal oriented at the greatest angle with respect to the line of

sight (see Figure 4.12 for an example of such a surface). All other point pairs will have smaller perceptual differences between them.

Once the two points are determined, the reflectance from each of these points is calculated:

$$R = \sum_{lights} k_d(L \cdot N) r_d + k_s(M \cdot V)^{\alpha} r_s \tag{4.1}$$

This is the standard Phong model equation [63], and is performed for each point. For the purpose of analyzing the visibility of surface roughness, all the values except the surface normal are the same for each point. This yields the actual difference in lightness that the eye receives. However, in order to predict the detection of surface roughness, human contrast sensitivity must also be taken into account.

The two equations used in this research for contrast sensitivity are the contrast sensitivity function (CSF) [51]:

$$CSF = 2.6 * (0.0192 + 0.114 * f) * e^{-(0.114*f)^{1.1}} \tag{4.2}$$

where f is the frequency of the bumps; and just noticeable difference (JND) [80]:

$$\Delta L = 0.0594 * (1.219 + L_a^{0.4})^{2.5} \tag{4.3}$$

where L_a is the brightness of the surround and ΔL is the least difference in luminance that would allow a human to distinguish the lighter foreground from the background. The JND formula takes both lightness adaptation and the varying response of humans to lightness levels into account. The brighter the surround, the more light that is required for a person to be able to perceive a difference between the brighter foreground and dimmer background.

The CSF is used as a normalization factor for the calculated Phong reflectances, and the result if divided by the JND formula to determine the number of luminance steps

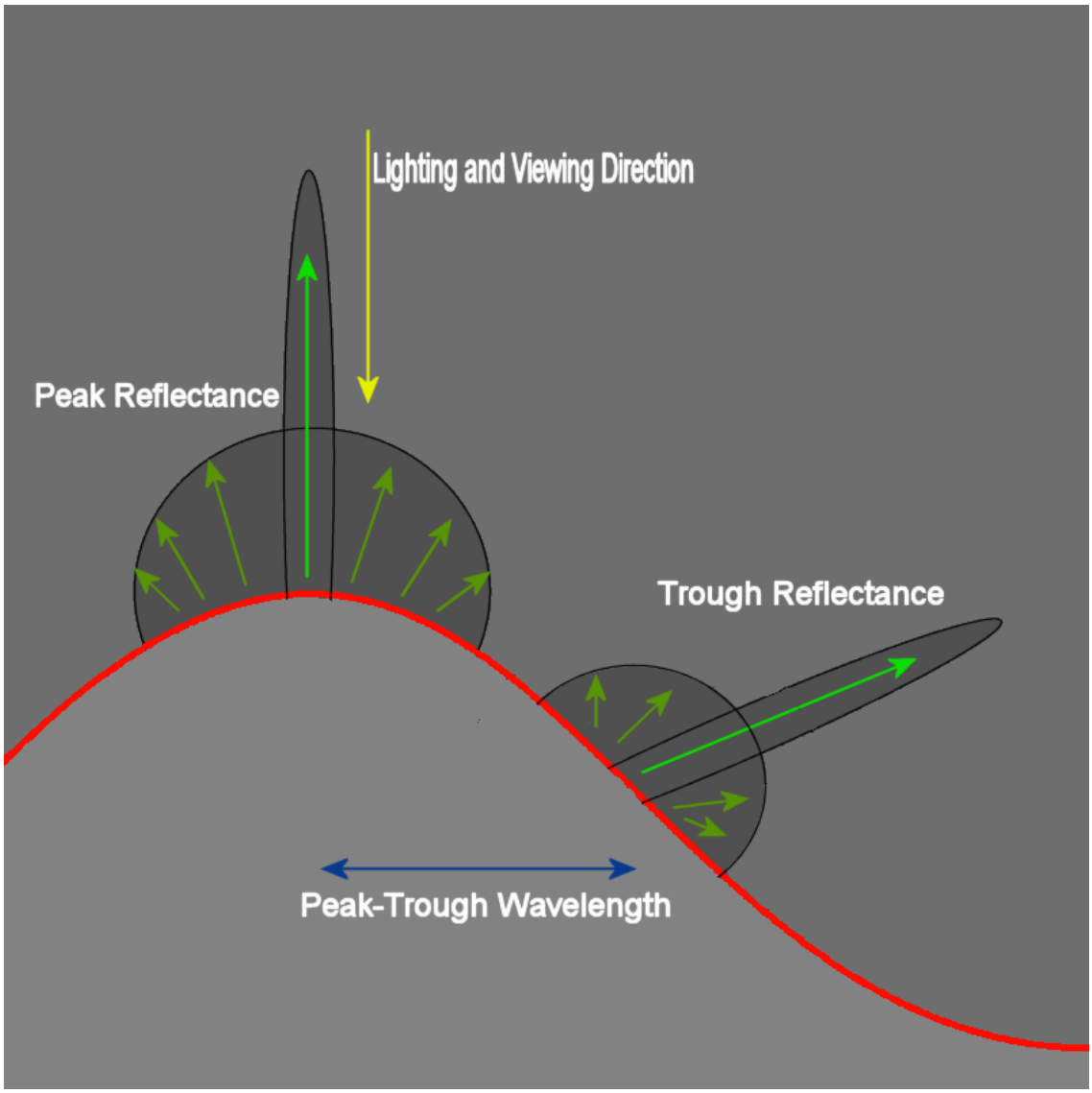

Figure 4.12: An example of a single bump and the determination of the minimum and maximum reflection points along that bump. The reflectance on the left is the peak Phong reflectance: the point where the reflected light from the surface has the specular Phong lobe pointing directly toward the viewing direction. The reflectance on the right is the trough Phong reflectance: the point where the specular direction is furthest from the viewing direction and the diffuse reflectance is also at a minimum. The analytical model takes the physical lighting differences of these points into account along with human visual factors, lightness and spatial acuity, to calculate a final perceived luminance difference between the points.

between the peak and trough value:

$$L_{steps} = ((R_p * CSF) - (R_t * CSF))/\Delta L \tag{4.4}$$

where R_p is the Phong reflectance at the peak reflectance, and R_t is the reflectance at the lowest reflectance. A result greater than one means the bump can be detected, while a result of less than one means it cannot. Note that, as shown in Figure 4.12, high wavelength amplitude causes the specular lobe of the trough point to completely miss the eye, causing sharp contrast between the peak and trough reflectance. Therefore, for higher levels of surface roughness, this model represents a "worst case" scenario in terms of orange peel visibility: there are no obfuscating factors other than human visual acuity and the diffuse reflectance.

4.6.2 User Evaluation

The orange peel simulation discussed in this chapter allows the analytical model derived above to be compared to the physical simulation of the actual bump distribution. Directly comparing the two can validate both the analytical model and the orange peel system.

In order to test the validity of these models, two experiments have been performed. The first test replicates the circumstances given in the analytical model using the orange peel system. The lighting is a checkerboard environment map with alternating light and dark checkers, and only a single bump wavelength is used. The amplitude of the Wave-Scan value determines the normal direction used in the Phong equation, and the result from that equation is used in the JND calculation. The CSF is determined by the wavelength frequency on the computer monitor combined with the viewing distance of the user. The purpose of this experiment is to verify that viewer detection of the simulated orange peel matches that predicted by the analytical algorithm derived above.

Each of 6 subjects first took the Snellen visual acuity test to verify that he/she had 20/20 vision. Next, the subject was placed exactly one meter from a pre-calibrated computer monitor and adapted to the proper level of ambient illumination. The subject was then asked to view a series of 84 randomized images. 75 of the images had a varying wavelength and amplitude of orange peel, as well as one of three different checkerboard contrasts. 9 of the images were control images with no orange peel. Each subject was asked to say "yes" or "no" to whether or not they could see any roughness in the surface of the object displayed on the monitor. In the case of a "yes" answer, subjects were also asked how large the bumps were - either small, medium, or large. Subjects stated "no" to 51 of the 54 control images (94%).

The resulting answers were then averaged to determine at what amplitude a particular wavelength/contrast combination could be seen, and compared to the expected results given by the analytical model. The results of this experiment are summarized in Figure 4.13. Overall, the results match the analytical model very well. In nearly every wavelength/contrast combination, the subjects' ability to see the surface roughness averaged to within 10% of the expected orange peel wavelength value given by the analytical model.

The only wavelengths that did not match within 10% were 25mm for 90% contrast and 10% contrast. For 90%, the value was within 20% and small in terms of its absolute variance from the expected value. However, the results for 10% contrast were not within a reasonable range of the expected value, and the subjects had a very high variance in their answers. One possible explanation for this is that at such a high wavelength, other structural cues were present in the surface of the sample that were not predicted by the analytical model. Another possible explanation is that the users had such a difficult time determining this wavelength that many more samples need to be taken to gain a valid

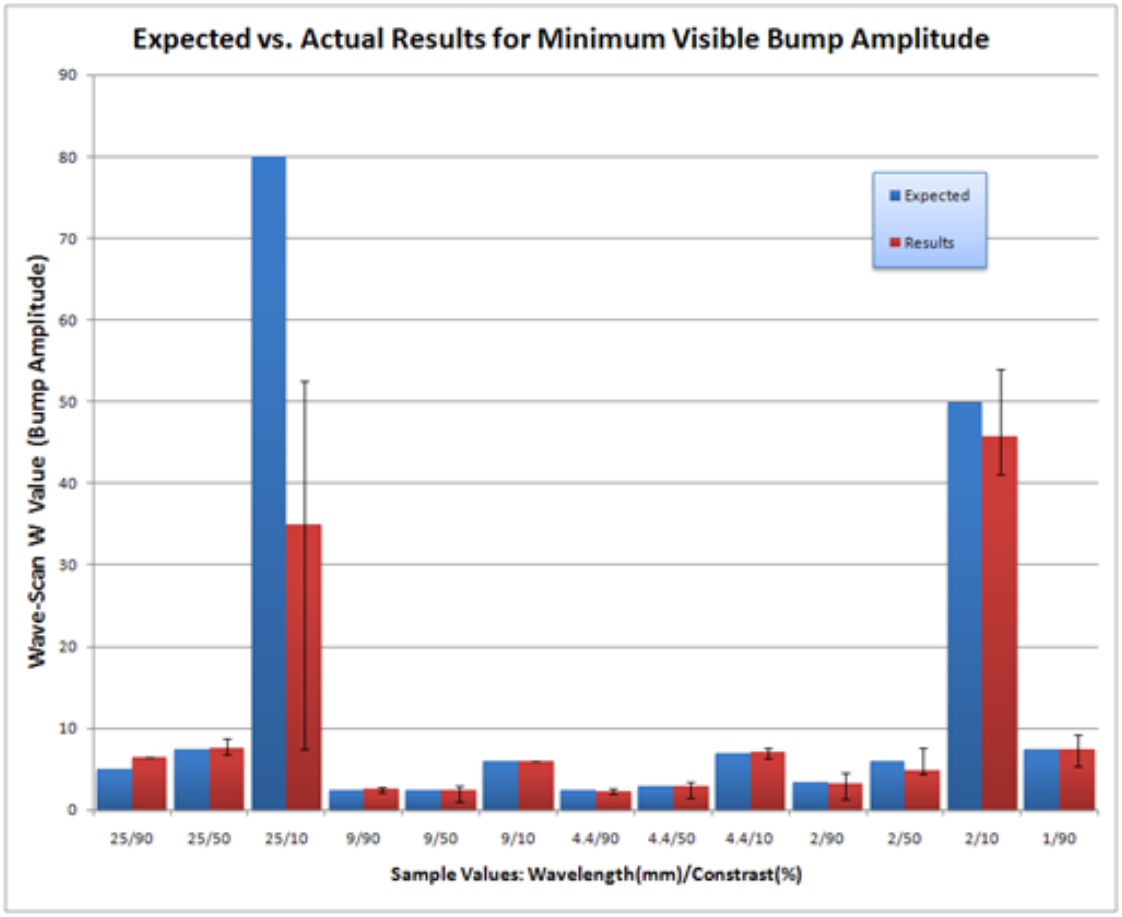

Figure 4.13: A summary of the first orange peel experiment results. The blue bars represent the expected value at which viewers should be able to perceive surface roughness predicted by the analytical equation. The red bar represents the actual value at which the average user was able to perceive the surface roughness. The 1mm wavelength bars for 50% and 10% contrast are omitted as both the expectation and user result was that the roughness was completely imperceptible at all the tested wavelength amplitudes.

Figure 4.14: Two images from the second orange peel experiment. Left: An image with differing lightness and orange peel values. Right: One of the control images with varying orange peel values but the same lightness.

experimental result. In either case, more experiments will need to be performed in order to address the discrepancy.

In the second experiment, a more complicated lighting environment was used along with varying Wave-Scan parameters to show that the orange peel program can be utilized to prototype surface tolerances based on various factors such as lighting environment and paint color. Specifically, it shows that lighter paint colors obscure visibility of orange peel, and therefore less stringent surface quality requirements can potentially be set for lighter surfaces than dark surfaces.

For this test, each of 8 subjects was asked to view a series of 55 images. Each image consisted of a shape half coated with one value of lightness/orange peel, and the other half coated with a different orange peel/lightness combination (see Figure 4.14). All the samples had neutral coloration (gray). Each user was then asked which half of the sample was “rougher.” They were not allowed to say the samples were the same.

The set of images consisted of 36 images with differing lightness and orange peel

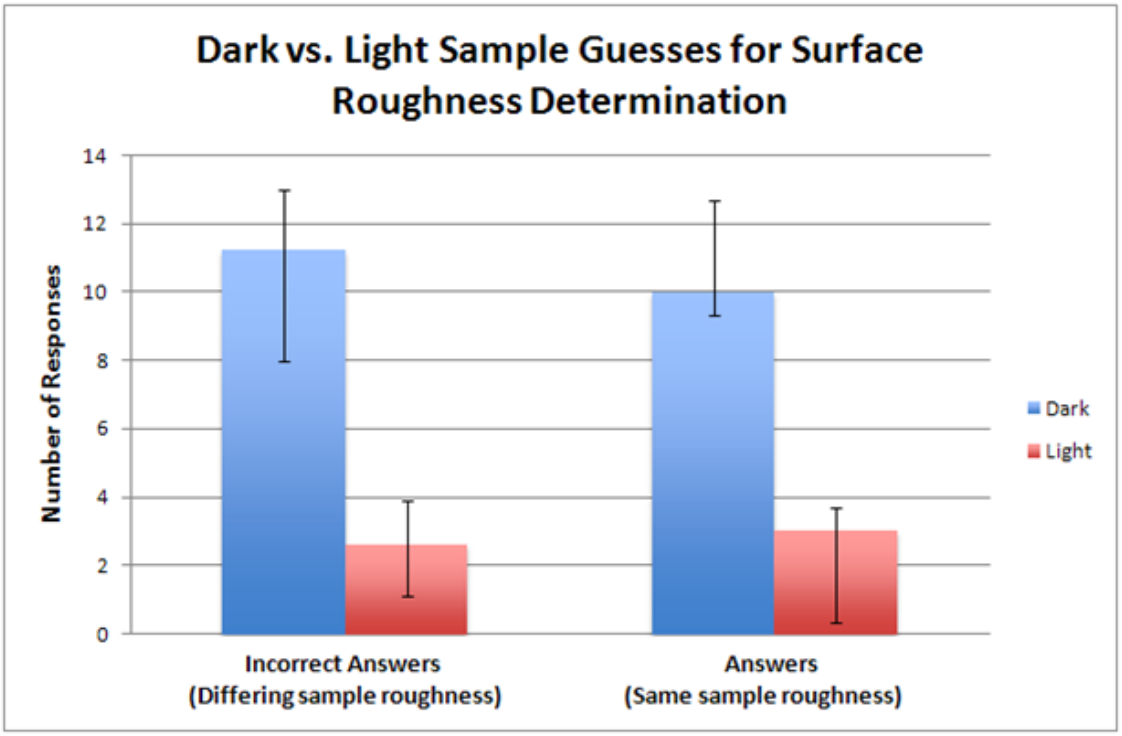

Figure 4.15: The result of the second orange peel experiment. Left: The plot of average number of dark vs. light incorrect guesses. Note that dark was chosen far more than light (approximately 81% of the time). Right: Plot of dark vs. light guess when orange peel was actually the same. Once again, dark was seen as being rougher a significant percent of the time (77%)

between the halves, 13 images of differing lightness but the same orange peel, and 6 images of the same lightness and different orange peel. All images that had differing orange peel and lightness had an image with the lightness-orange peel combination swapped (the same orange peel on different halves of the sample). This way, if users viewed the orange peel in the same way regardless of lightness, they should pick the lighter side 50% of the time, and the darker side 50% of the time. A result of greater than 50% selection of darker samples means that the lighter samples obfuscate the visibility of surface roughness.

The task of identifying precise orange peel roughness is fairly difficult. Users got 87.5% of the control cases correct, and 64.5% correct overall. In cases where the user

guessed wrong, dark was selected 81% of the time and light 19% of the time. In cases where orange peel was the same, users selected dark 77% of the time and light 23% of the time. Figure 4.15 summarizes the results of this experiment. Note that the deviation between the dark and light selection does not overlap. This means that there is effectively no chance that the choice is simply a 50/50 guess ($p = 2.34 * 10^{-5}$) confirming the original hypothesis that lighter shades of color obfuscate the visibility of surface roughness. However, in order to determine the exact deviation caused by lightness, more samples would need to be taken.

The results of these two experiments show that the orange peel program described in this chapter has the potential to be used to prototype surface roughness. When designing a new orange peel/color combination, a designer can use the orange peel program to study the effects of wavelength, amplitude, color shade, and lighting environment. Such use of the program could therefore facilitate faster and better design of automotive paints and spray systems.

Chapter 5

Display Devices

To this point in the thesis, two aspects of paint simulation have been covered: application of the paint and physical simulation of the paint's reflectance properties. Now, to generate the greatest verisimilitude in comparison to real paint, these simulations must be displayed to the user in a convincing manner. Display technology has undergone major changes over the past 10 years in an effort to give users higher fidelity images than old CRT monitors have been able to provide. LCD, LED, DLP, Plasma, and HDR displays have all been introduced in efforts to make displays lighter, brighter, higher contrast, and have better color. In addition, more specialized displays such as virtual reality goggles and projection displays have been developed. Ideally, a given paint color should be displayed to the user in such a manner that they cannot tell the difference between the virtual sample and a real paint sample. While typical two dimensional displays can achieve good color reproduction, they are poor at convincing people they are looking at a real sample: it's easy for users to see they are looking at a flat monitor. VR goggles can provide better immersion, but have the problem that for a user to be totally convinced, the entire scene must be rendered with very high quality, not just the paint sample. Not only that, but

Portions of this chapter have appeared in CGIV 06: Color in Graphics, Imaging, and Vision [43] and IEEE Visualization 05 [45]

current VR displays have low pixel density, poor color reproduction, and narrow fields of view. While this is probably sufficient for the training simulation given in Chapter 2, more exacting applications such as the airbrush simulation and paint design will suffer from poor color output.

An attractive compromise between 2D displays and VR is augmented reality. Augmented reality involves altering the appearance of an object in a real environment. In order to achieve this, projectors are frequently used. Projectors can have high color quality, and can be positioned such that a user may not even know the scene in front of them is being altered. This chapter covers research into both front and rear projection systems that are designed to show the user the most convincing possible illusion of how the final paint will look.

First, the object onto which to project the simulated image is considered. Since the projected light is reflecting off the object, the material of that object can have a dramatic effect on the user's final perception of the presented illusion. In addition, it is important that the object have a shape that provides all the cues necessary to make a good judgement about the designed paint. Therefore, this chapter studies both the shape and material of the object that is to be virtually painted.

Second, ambient lighting conditions and proper simulation of a light source in both the virtual and real environment is researched. Since ambient illumination in the environment will reflect off the object in addition to projected light, ambient light will "wash out" the simulated paint color. Therefore, it is desirable to eliminate ambient light from the environment. However, this results in the object appearing to be self-luminescent, which is incorrect for an automotive paint. The solution to this problem presented in this chapter is to create a custom lighting setup using projectors. Projectors can create a natural lighting environment while leaving the augmented object almost completely unlit by

ambient illumination.

Third, proper simulation of reflections and gloss are taken into account. Gloss is an important effect in automotive paint, since nearly all such paints are highly reflective. Since the appearance of gloss is dependant on the surrounding environment, simulating this effect can be tricky. Therefore, this chapter also considers how to properly capture the surrounding environment and use that environment to provide a convincing simulation of gloss.

Finally, interactivity between the user and the system is explored. When evaluating a painted surface, many users will either move around the sample or hold it in their hands to view the paint from varying angles and distances. It is therefore desirable to provide the user with this sort of interaction when viewing virtual samples. Both viewpoint tracking systems and a specialized spherical rear projection system are presented in this chapter that allow the user to hold a virtually painted sample and manipulate it in a natural manner.

5.1 Relevant Work

This section reviews the basic concepts and implementation details for creating computer graphics using projectors. These applications involve the use of projectors in non-traditional ways, for example to make an object look as if it has a different surface finish. This section describes how to project onto planar and non-planar surfaces using different projection techniques. It also describes how to correctly light objects using projectors for different user viewpoints. Finally, it discusses real world lighting considerations that must be taken into account when using projectors. This section is intended as a survey of projection graphics techniques that are the current state of the art, and the material covered forms the basis for the projection display work that is described in the rest of this chapter.

5.1.1 Projection Onto Moving and Non-Planar Surfaces

In most projection setups, a projector is used to display an image on a flat screen a set distance away. However, some new applications for projectors are emerging that call for projecting onto non-planar and moving objects. This section discusses the techniques involved in projecting onto these kinds of surfaces.

Non-Planar Surfaces

When projecting onto non-planar surfaces, merely projecting the desired image is insufficient, because a curved object will distort the projected image. Raskar et. al presents methods for projection onto non-planar surfaces [65]. In order to accomplish this, two things must be done. First, the projected image must be pre-warped such that the image is undistorted when projected onto the object. Second, the geometry must be registered so that the warped image is correctly aligned with the real object.

Performing the image warping is relatively simple. The geometry of the object is loaded into the program, and any image to be displayed is textured onto that object. Thus, on a normal monitor, the scene appears to be a picture of the real world object with the image textured onto it. However, when this scene is projected onto the real object, it will simply appear to be an undistorted image on that object. The most common way to solve this problem is to add a second pass to the graphics program. The first pass calculates the desired scene to be projected, and the second pass textures this scene onto the object. While this is an effective solution to the image distortion, it does necessitate having a geometric representation of the real world object in the computer. For complex objects, acquiring this mesh can be non-trivial.

Once the image has been correctly pre-warped, the result must be aligned with the

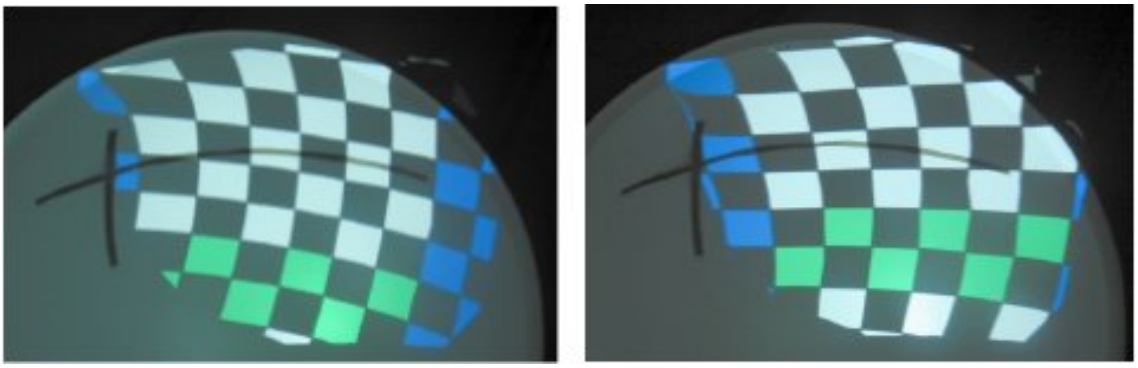

Figure 5.1: Left: Image of a checkerboard projected onto the inside of a hemisphere. Note the stretched checkers on the left side. Right: A corrected projection of the checkerboard. Reproduced from [64].

object. This can be done by moving the projectors manually, or by translating and rotating the image in virtual space. Since physically moving the projectors is often impractical, the second of these two solutions is generally preferable. This can either be done manually, automatically, or using some method in between. Shader lamps [66] introduces a method to perform this alignment by having the user line up known points on the object with projector pixels. These points are then used to automatically create a translation/rotation matrix for correct alignment. Totally automatic registration is accomplished for quadric surfaces in [64] using pre-calibrated projector-camera pairs.

Tracking Projection Objects and User Viewpoint

Once the projector image has been correctly pre-warped and calibrated, we can consider what to do in the case that the object is moved. In order to correctly project onto a moving object, the object must be tracked. This can be accomplished using one of many available commercial trackers. Once accurate position and rotation data for the object is received, correction for a moving object is simple. The matrix used for the geometric alignment is simply updated with the new tracked coordinates of the object. Thus, if the object is

moved, the translation part of the matrix is modified. If it is rotated, the rotation part of the matrix is changed. Assuming the initial alignment was done correctly, this will properly update the projected image.

One thing to note about projection on moving objects is that most projectors have a limited depth of field and field of view. The field of view is the rectangular viewing volume into which the projector emits light. Only objects within this viewing volume are valid projection targets. Even more restrictive (and less obvious) is that without adjustment, normal projectors can only focus properly within a limited portion of this viewing volume. If an object is moved significantly, the image will go out of focus, most likely ruining the desired effect. In the case of large objects, it is even possible that a portion of the object will always be out of focus no matter how the projector is adjusted. Section 2.1.4 discusses the use of a projector lens that helps to solve these problems.

In addition to tracking a moving object, the ability to track a moving viewer can be important for certain effects and lighting considerations (see sections 2.2.1 and 2.2.2). For this reason, it may be desirable to track the viewpoint of the user. This is done in the same manner as an object, with the viewer's location being loaded into the computer.

Rear Projection

The most common method of projection is front projection. This is where the projector is placed in front of the target object. In most cases, the object has a mostly diffuse surface, allowing the reflected light to be seen from any direction. In this case, the viewers and the projector are on the same side of the object. However, a second kind of projection exists: rear projection. In this kind of projection, a special projection material is used that transmits light. In this case, the projector is on the opposite side of the object as the viewer. In rear projection, the viewer sees a mirror image of what is seen in front

projection.

Many projectors have automatic controls to correct for rear projection. However, this may not be sufficient in the case of non-planar objects, as the mirrored image may not line up correctly with the real world object. Fortunately, correcting for this is very similar to front projection on non-planar objects. Instead of texturing onto the front of the object, it is instead textured to the back. In both cases, this leads to the object being rendered from the perspective of the projector. One important thing to remember for rear projection is that the object movement in relation to the viewer and projector is also mirrored. It is therefore helpful to set a world coordinate system that centers on the projector and everything moves in relation to it. This removes the need to worry about coordinate reversals due to rear projection.

Spherical Projection

The Elumens Corporation has created a spherical lens system that is used in this research. This is a fish eye lens, originally created to project onto a hemispherical dome, that projects 180° along the horizontal axis and 135° degrees along the vertical axis. A special property of this fisheye lens, shown in Figure 5.2, is that it projects with an exact f-θ pixel distribution. This means that the angle to which a specific pixel is projected is linearly proportional to its distance from the optical axis. The result is an equiangular pixel distribution across the entire projected field. Typical rectilinear projection lenses project with an f-tan(θ) distribution which results in smaller angular pixel distribution at the center than at the edge of the projected field.

In addition to creating an extremely large viewing volume, this lens also has nearly infinite depth of field. In optics a common rule of thumb is that for a lens of focal length "f" an object that is farther than 20f away is essentially infinitely far away. Revers-

Figure 5.2: A close up of the spherical projector lens.

ing this rule for projection, if the screen is closer than 20f from the lens the image will always be in focus. The fisheye used in our system has a focal length of 6mm. Therefore, the screen can be placed anywhere from infinity to 120mm from the lens while maintaining a focused image.

One side effect of this spherical lens is that the projected image is distorted by the spherical distribution of pixels. There are two ways of correcting this distortion. The first is to create a virtual sphere in space and texture the desired image to that sphere. This creates a pre-warp that will fix the spherical image distortion. This is very similar to the method discussed in 2.1.1 for projection onto non-planar objects. This is the method used by the Elumens Corporation for their hemispherical domes. The second method utilizes the ability of the hardware vertex shader to dynamically move vertices as objects are drawn. This method is accomplished by creating a mapping between the spherical projection space and the rectangular screen space. The equiangular projected pixel distribution allows for the use of a single transform that will move every vertex in spherical space to its proper location in rectangular screen space. First, the world location of each vertex is

converted into spherical space:

$$r = \frac{2}{\pi}\arccos(\frac{z}{d}) \tag{5.1}$$

$$\vec{\phi} = \left(\frac{x}{\sqrt{x^2+y^2}}, \frac{y}{\sqrt{x^2+y^2}}\right) \tag{5.2}$$

$$\left(x', y'\right) = r * \vec{\phi} \tag{5.3}$$

where Z is defined as the axis parallel to the center of the projector's field and d is the distance of the vertex to the center of the projector. The origin of this system, $x = y = z = 0$, is defined as the center of the projector lens.

Now $\vec{\phi}$ corresponds to an x, y screen space vector and r is the length of that vector. This moves every vertex into a two dimensional screen space between $-\pi$ and π. Dividing r by π as in the above equation normalizes this result to be between -1 and 1. In order to preserve depth, each vertex is then given a depth value equal to its distance from the projector. This method is similar to that used by Raskar et al. [64] for quadric image correction, and by Coombe et al. [11] who use the technique to project points onto a hemisphere in graphics hardware for radiosity calculation.

Each method has its advantages and disadvantages. The vertex mapping method has the advantage of requiring no extra geometry to be created, and no extra passes. However, its accuracy is based on the tessellation of the objects being warped. Therefore, objects with low vertex counts may have to be re-tessellated to achieve acceptable results. This will slow the program down and create extra work for the programmer. Also, complex scenes may become slow due to the extra vertex calculations being performed.

The texture mapping method requires additional geometry for a texture mapped sphere. It also necessitates an extra pass to texture the scene onto the sphere. It has two advantages over the vertex mapping method. First, the accuracy is based on the tessellation of the sphere that is textured, not the geometry in the scene itself. This makes

it easy to guarantee a desired level of accuracy. Second, the speed of the method is not related to the complexity of the scene being projected. For the purposes of this research the vertex mapping method is employed. This is because the scenes that are used are simple and lend themselves well to object tessellation, thus making the vertex mapping method easier and faster.

5.1.2 Rendering Virtual Lighting With Projectors

All of the techniques discussed thus far make no assumptions about the scene that is projected onto the objects. The projectors can either be "painting" an image onto the object, or projecting some kind of virtual lighting onto the object surface. In both of these cases, the required image warping techniques are the same. This section focuses on techniques that involve projection of virtual lighting and material properties onto an object. This technique is referred to as "shader lamps" as defined in Chapter 1.

Diffuse and Ambient Light

Performing diffuse and ambient lighting with projectors is very similar to a normal graphics program. Since diffuse and ambient lighting are not view dependant effects, the only important variables are the surface normal of the object and location of the light. The scene is defined so that the viewpoint is coincident with center of the projector lens. The object being viewed should have the geometry of the actual target object, and its surface normals set to the normals of the object. The location of any desired virtual lights in the room are then set. Finally, the object is rendered using the appropriate lighting equations.

One noticeable difference between non-view dependent lighting using projectors and normal graphics illumination calculations is when the normals of the object do not point directly back at the projector. In this case, assuming the real object is diffuse (which

is usually the case) the brightness of the projected light will fall off with the cosine of the angle between the object's normal and the projector.

Thus, when the normal points directly back at the projector, no correction is required. However, as the normal moves away from the projector's position, the intensity of the image will drop off, ending with no light reflected at 90°. This can be corrected by dividing the lighting equation by $\cos(\theta)$, where θ is the angle between the vector pointing from the object to the projector and the object's normal [66].

Specular Lighting

View dependant lighting introduces a further difference between normal computer graphics lighting and projector based lighting. With a conventional graphics program, the location from which the scene is being viewed and the location from which the scene is being rendered are almost always the same. However, this is not the case with projector based lighting. The scene is always rendered from the position of the projector. The viewer, on the other hand, could be anywhere in the room. Therefore, the viewpoint for rendering and lighting calculations must be separated.

Raskar et al. introduce a technique for altering the modelview matrix for the viewing and rendering viewpoints [66]. However, this separation can be achieved more easily on modern graphics hardware. Current graphics vertex and pixel shaders allow arbitrary variables to be passed into them. Therefore, instead of using the usual modelview matrix to determine the viewpoint employed in lighting calculations, the observer's viewpoint is instead passed in as a separate variable. This is then used for the lighting calculations, and the modelview matrix is used only for vertex placement. This solution is very easy to program, and requires very few extra resources for lighting calculations beyond what is employed by a normal graphics program.

5.1.3 Physical Lighting and Projection Object Material Considerations

For most computer graphics applications, ambient lighting conditions in the room are rarely controlled, and no objects other than the monitor itself must be considered. In order to obtain the best results from projector graphics, not only is the projector important, but also the target projection object, as well as the ambient lighting conditions of the room. This section discusses both the properties of ambient lighting and the choice of target projection object, and the impact of these on the quality of rendering results.

Ambient Light Conditions

The ambient lighting conditions in a room where projectors are being used can be very important. Just as projector lighting, ambient light also reflects off of objects. This has the effect of "washing out" the image produced. This is because the darkest a projected image can be is the sum of the projector's black level (projectors always emit some amount of light, even if a pixel is set to black) and the ambient light hitting the object. For instance, if a projector has a white level of 1000 lumens, a dark level of 10 lumens, and the ambient light adds 20 lumens, the effective contrast of the projector has been cut from 100:1 to 33:1. This, of course, is very undesirable. Therefore, it is important to dampen the ambient lighting of the surroundings to achieve the highest contrast possible.

Projection Screen and Object Reflectance Properties

The material properties of the target projection object can have a very significant impact on the quality of the graphics produced. For this reason, there are currently many companies that produce projection screens that have different properties. The primary properties of these screens are the lightness, (how much light they absorb vs. reflect) and the gain of

the screen (how much light is reflected back at different angles).

Gain is increased by making the screen retro reflective, so that more light is bounced back in the direction of incoming light. A gain of one means the surface is perfectly diffuse, reflecting light equally in all directions. A higher gain will reflect more light in the direction of the projector, but at the cost of producing a dimmer image at other angles. This, however, can have the benefit of reducing the effects of high ambient light. Any ambient light not coming in from the same direction as the projector will be dampened. Thus, even if the ambient light is coming in equally from all directions, the amount of projector light being seen is enhanced even though the ambient light remains constant. If the ambient light is directional, and not from the same direction as the projector, it will even be dampened.

Screens that are darker, usually some shade of grey, are frequently referred to as "high contrast" screens. This is because they lower the darkest value that the projected image can achieve. This makes the difference between a "black" and "white" screen appear larger. This difference, however, is purely perceptual: the lightest image that can be projected is dimmed by the same fraction as the darkest image.

If a target projection object has a color of its own, it can still be made to assume a different color by projecting a large amount of dissimilar color onto it. However, this will dramatically reduce the possible range of colors and brightness that the object can be made to appear. For this reason, it is almost always best to project onto neutrally colored objects. It should also be noted that if an object completely blocks some spectra of light, it can never be made to appear that color.

For the purposes of this research, all objects are neutrally colored and have a gain of nearly one. This gives the widest possible dynamic range, and makes it easier to precisely calculate the reflected color at any point on the object.

Secondary Scattering

Secondary scattering is the effect of light bouncing off one object and onto another. For projector based graphics, this effect can be either helpful or detrimental. Since the target projection surface is almost always diffuse or nearly diffuse, the light from the projector is reflected equally in all directions. This creates the same secondary scattering effect as shining a neutrally colored light on a colored diffuse surface. If the surface is supposed to look diffuse, this secondary scattering will appear correct, and thus help to complete the desired illusion.

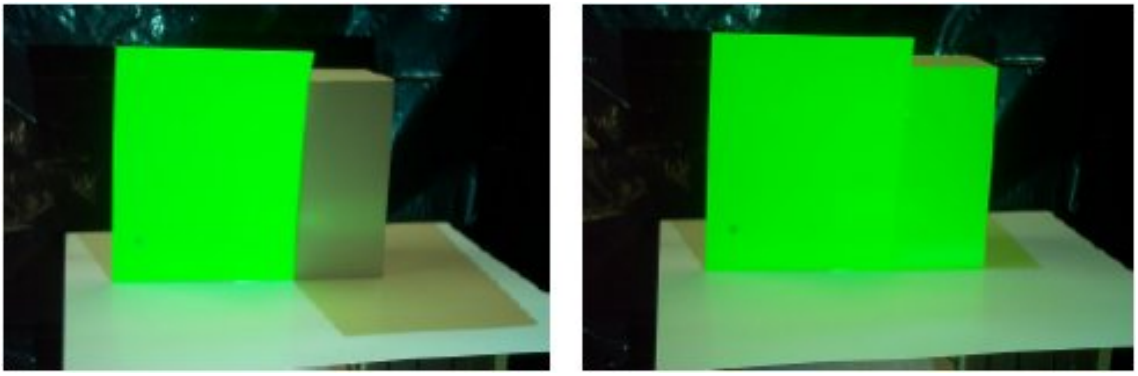

Figure 5.3: Left: Image of green paper illuminated with white light. Right: The white diffuse surface is illuminated with green light. Note how the secondary scattering is similar in both cases. Reproduced from [66].

However, if the desired effect is to create an object with specularity, then the generated secondary scattering will be incorrect. A normal specular object will reflect light directionally, creating a caustic on nearby surfaces. However, a specular highlight being simulated with a projector will instead create a uniform brightening on nearby surfaces. The expected caustic will not be present, and this may harm the illusion being created.

Another situation in which secondary scattering can be of importance in projector graphics is when a directional light source is used. A strong directional light source pointed at or near the target object can create secondary scattering on that object. This

can add the effect of extra ambient lighting, or even alter the color of the image.

5.2 Physical Setup

5.2.1 Object Properties and Placement

The target projection object has a substantial effect on the quality of the illusion produced by the color simulation. The lightness, color, and shape of the object all affect the final result that a viewer will see from a projected color simulation. In addition, the relationship between object and projector placement is also important.

The simplest of these considerations is the color of the object. A neutrally colored (gray) object is best, as this allows the widest range of possible colors to be simulated. For choice of object lightness, it was found that a darker surface gives a more convincing image. This is because projectors always emit a certain minimum amount of light, and it is important that this be perceived as black. On the other hand, it was found that the maximum lightness of the projector was not as important, except when simulating specular highlights (Section 5.2.3). In addition, by using a light source projector (Section 5.2.2), the incoming light can be calibrated to be within range of the projector brightness. That is, the simulated material reflectance multiplied by the magnitude of the incoming light is made to be less than the maximum brightness of the projector multiplied by the actual object reflectance.

In addition to object material properties, the shape of the object is also important. While a correct color simulation could be projected for any object shape, choice of shape can alter the perception of the object surface [33]. A reasonable shape to use for color simulation would be a cylinder, which is a shape currently used for material samples by car paint companies. However, when simulating gloss, it was found to be difficult to keep

both a specular highlight and a recognizable environment on the cylinder at the same time. To see this, consider the case where the shape is sitting on a table with a recognizable surface (such as a checkerboard pattern). In order to see the reflection of the table, the viewpoint must be above the shape. However, in order to see a specular reflection of the light at the same time, the light source would have to be below the table. This is an awkward (and uncommon) place to have a light. Therefore, for the final shape, the top of the cylinder was replaced with a hemisphere. The hemisphere will show a specular highlight for any light source placed above the shape (which includes most lamps and any ceiling light), while the reflection of the table can still be seen on the cylindrical portion of the shape.

The relation between object and projector placement is also important. The closer the projector is to the object, the brighter the displayed color is on the object, and the higher the resolution of pixels on the object. Overall, it was found that the closer the object is to the projector, the more convincing the illusion.

5.2.2 Ambient Lighting and Projector Light Sources

As mentioned in Section 5.1.2, ambient light conditions can have a serious effect on projector based graphics. When trying to create the illusion that an object actually has a specific paint color, it is even more important to control all light in the room. Any light different than the simulated color that strikes the target projection object can ruin the illusion, or at least cause the perceived color to be different from the intended color.

Unfortunately, any traditional light source will have the effect of adding ambient light to the room, and will therefore ruin the simulation. However, if there is no light at all in the room, then the object will appear to ”glow” with the projected color. This has an unnatural appearance, as few materials can appear to glow under no visible light. The

solution to this problem is to use a light source that can illuminate the rest of the room without throwing light onto the object itself. For this, projectors provide a good solution. Just as a projector is used to give the object its color without lighting anything else, a projector can be used to light everything except the object. This gives the illusion of a spot light illuminating the object, but only a very small amount of light is actually striking the object (just the projector black level).

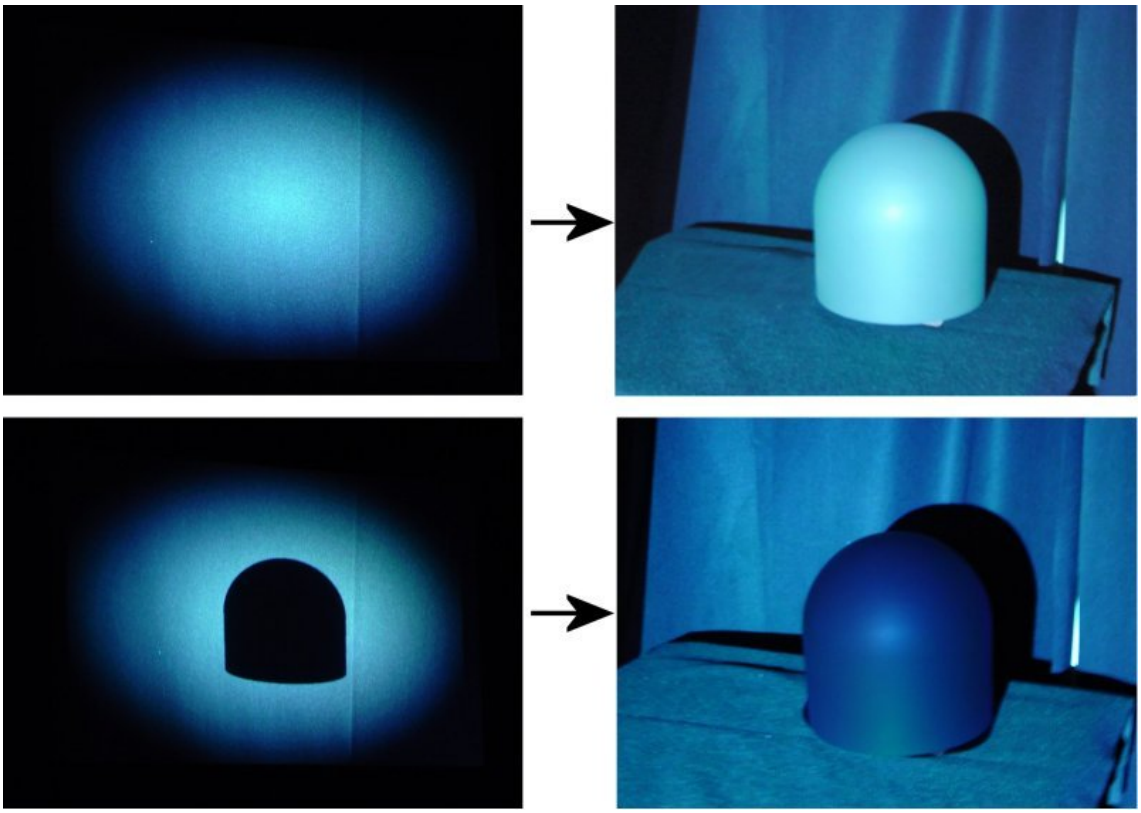

Figure 5.4: Top Left: A projector is used as a light source. Top Right: The result of the top left image when used on a neutrally colored shape. The shape is brightly lit by the emitted light. Bottom Left: The same light pattern as before, but the shape is cut out from the lit area. Bottom Right: The result of the bottom left image when used on the shape.

Another advantage of using a projector as the light source is that the brightness of the projected color and the brightness of the light source can be precisely calibrated to each other. This means that given a desired material property to simulate, the projector

light source can be brightened or dimmed so that the object is actually as bright as it would be under that light. This not only helps provide a convincing illusion, it also allows a direct comparison between a simulated color and a fabricated prototype of that color (5.5).

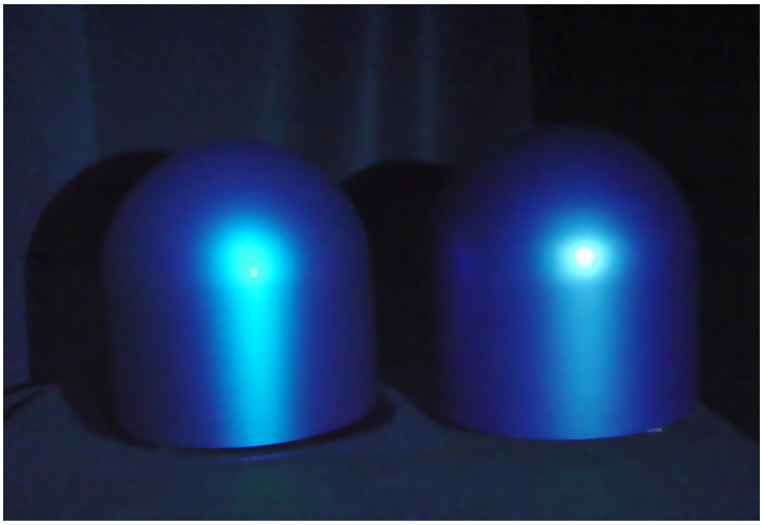

Figure 5.5: Comparison between a projected simulation and an actual metallic paint fabrication under a projector light source.

While using a projector as the light source helps to eliminate extra light from falling on the object, it is still important to remove as much secondary scattering from striking the object as possible. This can be accomplished by putting light dampening materials on the surroundings (such as black cloth). In addition, it is helpful to use a projector with as low a ”black” level as possible, to keep the black pixels from adding too much extra light to the object itself.

5.2.3 Simulation of Gloss

In order to properly simulate many materials, it is necessary to convincingly display gloss. For instance, in the case of real metallic paints, a clear coat is added on top of the metallic pigment. This gives the resulting paint a specular highlight, as well as reflecting the

surrounding environment. The specular highlight can be modeled using normal projector graphics, as in Section 5.1.2.

To simulate the reflection as realistically as possible, the environment around the target projection object (such as the checkerboard pattern mentioned in Section 5.2.1) can be captured and then displayed back on the object. The capture of the surrounding environment is accomplished by taking high dynamic range photographs of a reflective sphere [16]. This sphere is placed in the same location that the target projection object is normally placed. Once the photographs have been taken, environment mapping can be done using a programmable graphics card to simulate the glossy reflections.

One limitation of environment mapping on current graphics hardware is that the reflected environment is always assumed to be at infinity. In the case of many real environments, such as an object sitting on a table, this is not a valid assumption. As the viewpoint alters, reflected objects that are farther away ”move” more rapidly in the reflection than those that are closer. In order to account for this, the size and distance of each of the cubic environment map faces can be entered. The environment map lookup in the pixel shader is then altered based on these dimensions to provide a more realistic reflection.

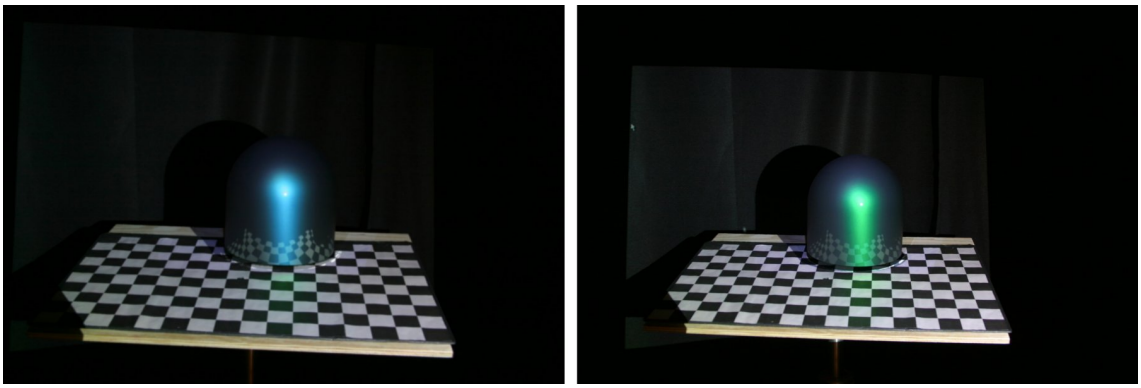

Figure 5.6: Two examples of projectors simulating glossy metallic paints.

5.3 Interfaces for Material Design

An important aspect of projector based graphics is the possibility of enhanced interaction between the user and the simulation. One can walk around a real 3D object and evaluate it from more angles and distances than a computer monitor. There are therefore a number of additions to a material design system that can increase interactivity and immersion into the system.

One such addition is user viewpoint tracking. As mentioned in Section 5.1.1, a tracker can be added to a projection system. This allows viewpoint dependant effects to change realistically as the user inspects the target projection object from different angles. Specifically, both the specular highlight and environment reflection can vary significantly as the user moves around.

When someone wishes to see precisely how a particular material looks, they will frequently turn it in their hands, and see how the light plays off the surface of the material. Therefore, another important interaction possibility is to allow the user to handle the material sample itself, and inspect it in their hands. The system described in the rest of the section uses a lightweight, bendable display surface to allow the user to view a material as if actually holding a sample of that material.

5.3.1 System Setup

The system setup consists of a projector with a spherical lens installed, a magnetic tracker, and a flexible rear projection screen. The spherical projection lens is as discussed in Section 5.1.1. The magnetic tracker currently used is an Ascension Tech PCIBird magnetic tracking system, although any tracker that tracks both translation and rotation can be used.

The choice of screens to use has several potential tradeoffs. Ideally, one would like

Figure 5.7: The system setup.

a large screen to view as much material at once as possible. However, a larger screen will make the system more bulky and harder to use, especially when flexing the screen. The choice of how flexible to make the screen is also important. If it is too rigid, it will take too much effort to bend, and it will not be possible to make a reasonable set of shapes to allow the light to play off the surface. However, if it is too flexible, it will be difficult to hold in a desired shape. Another consideration is the number of dimensions in which the shape can be bent. More dimensions will allow a larger set of shapes to be made, but at the cost of requiring more trackers and being harder to manipulate.

In the end, the selected screen shape is 9x11 inches: about the same size as a piece of paper. This lets a reasonably sized object to be viewed, but still allows for easy manipulation. Also, since people use paper all the time, this is a natural shape for most users to manipulate.

For the flexibility of the screen, two trackers are used, one at each bottom corner of the screen. This allows precise tracking of the screen's position, orientation, and curvature along one curve axis. This also permits a fairly wide variety of curves to be created, while keeping manipulation of the screen easy and straightforward.

In order to calculate the location of the screen, the origin of the generated scene is set to the position of the magnetic field generator, and all other objects are translated to their correct locations in relation to it. The orientation of the trackers is separated into the surface normal, tangent, and bitangent vectors. Let P_0, N_0, T_0, and B_0 be the position, surface normal, tangent, and bitangent of the tracker on the bottom left of the screen. Let P_1, N_1, T_1, and B_1 be the position, surface normal, tangent, and bitangent of the tracker on the bottom right of the screen.

Since each screen support is rigid, the positions of the top two control points can be deduced from the bottom control points and the length of the screen, l. This information is used to construct the geometry matrix as follows:

$$G_H = \begin{pmatrix} P_0 & P_0 + T_0 * l & B_0 & B_0 \\ P_1 & P_1 + T_1 * l & B_1 & B_1 \\ T_0 & T_0 & N_0 & N_0 \\ T_1 & T_1 & N_1 & N_1 \end{pmatrix} \tag{5.4}$$

The geometry of the flexible display can now be approximated by evaluating the Hermite patch Equation 5.5 over the domain $(0,1)^2$.

$$P(x,y) = Y \cdot M_H \cdot G_H \cdot M_H^T \cdot X^T \tag{5.5}$$

$$X = \begin{pmatrix} x^3 & x^2 & x & 1 \end{pmatrix} \tag{5.6}$$

$$Y = \begin{pmatrix} y^3 & y^2 & y & 1 \end{pmatrix} \tag{5.7}$$

$$M_H = \begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \tag{5.8}$$

When the screen is flexed, its stiffness could affect the resulting shape. Screen stiffness is not explicitly addressed in Equation 5.5. However, by scaling the length of the bitangent vectors in Equation 5.5, stiffness can be expressed implicitly.

5.3.2 Material Display

Once the screen shape has been made and correspondence between real-world location and virtual coordinates created, the desired material can be projected onto that screen. This is similar to the shader lamp concept described in Section 5.1.1 in which a real world object is modified with projected imagery. For the purposes of this system, the object to be modified is the projection screen itself.

Therefore, in order to properly display the material onto the screen, a virtual model of the screen is created. This virtual model is placed into the virtual scene at the same location as the real screen is in world space. After the appropriate image warping (described in Section 5.1.1) has been performed, the image of the desired material will precisely light the screen. The user can then give the screen any desired material properties and view how these properties appear in a desired environment by moving the screen around and flexing it.

Figures 5.8 and 5.9 show the system being used as a shader lamp. In this case, the

screen is being made to look at if it was painted with a blue metallic paint. One thing to note is that the location and size of the specular highlight in this application is dependent upon viewer location. In order to handle this, the viewer location can either be tracked or assumed to be static at a reasonable location. The screen is then rendered from the projector position, with the viewer location used for the lighting calculations. This is similar to the method presented by Raskar et al. [7].

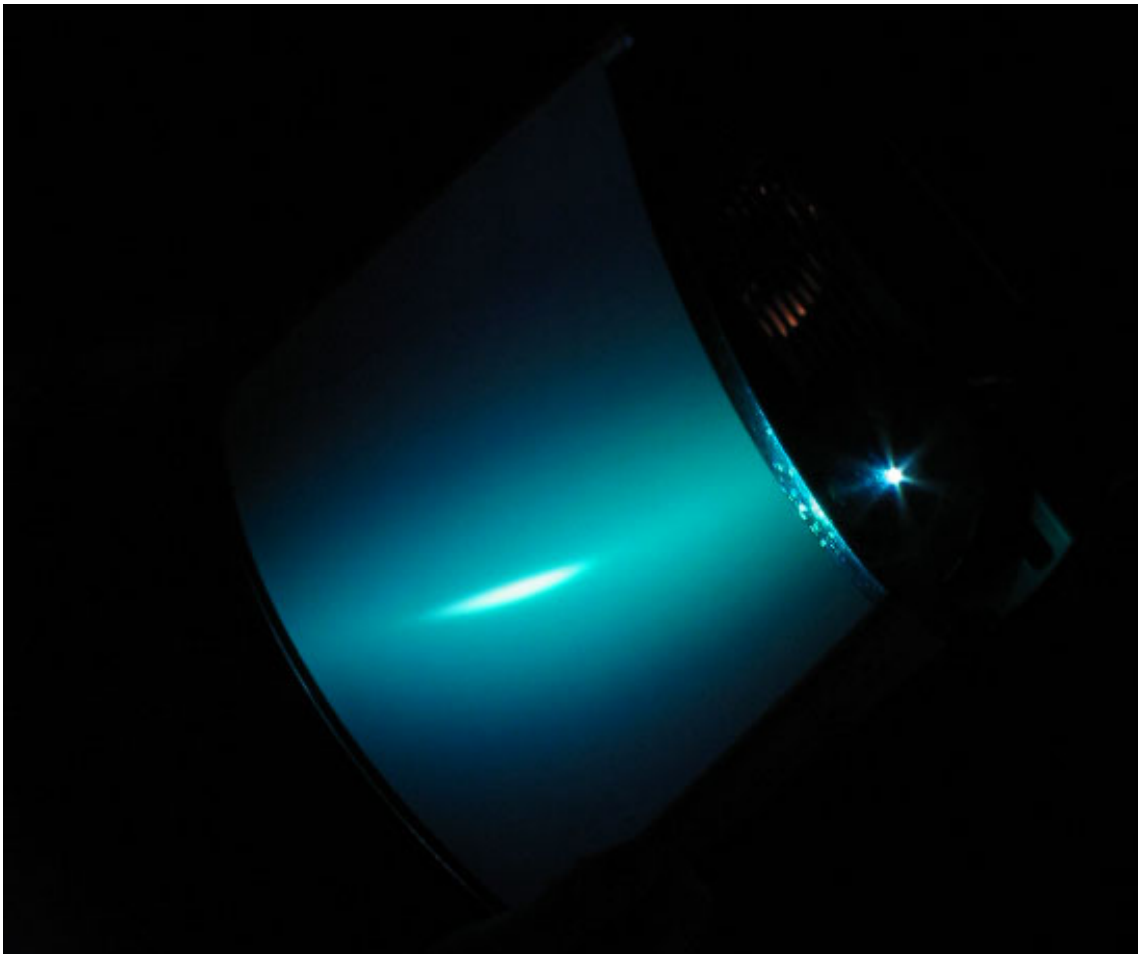

Figure 5.8: The flexible display system being used as a shader lamp. A user is viewing a designed metallic paint.

Figure 5.9: Another view of the flexible screen. The image displayed on the monitor is warped to display properly on the flexible screen.

5.4 Further Display Uses

So far, the displays presented in this chapter have focused on display and simulation of metallic paint. However, these projection systems needn't be limited to display of just this one kind of material. The concepts developed in this chapter can be applied to the display of any type of material, and even to visualization of different types of data such as volume visualization. For instance, the front projection system could be used to display materials such as cloth or concrete instead of metallic paint. The flexible screen system can also be used to visualize different material types, and even display volume data. This section briefly describes some potential other uses of this projection system.

5.4.1 Volume Slicing and Tracking

In this application, instead of projection a material property onto the screen, a slice through a virtual volume is instead projected. First the data set is loaded into the graphics hardware as a 3D texture. The volume is then given a location in world coordinates that corresponds to its desired location in the real setup space. The volume is also scaled to match the size of its real-world counterpart. Now there is a virtual data set ”floating” in space in front of the projector for the user to view.

Once the volume data set has been properly loaded and set up, the screen can be used to slice into it. To do this, the coordinates of the screen are sent into the pixel shader and compared with the coordinates of the 3D volume. When they intersect, the correct voxel of data at that location is found and output from the pixel shader. In this way, the complete intersection between the screen and the 3D volume is output. Since the intersection between the volume and screen is computed on a per-pixel basis, this solution is not only accurate, but is easily extendable to arbitrarily sized and shaped screens. See Figures 5.10 and 5.11 for examples of using the projection system as a volume slicer.

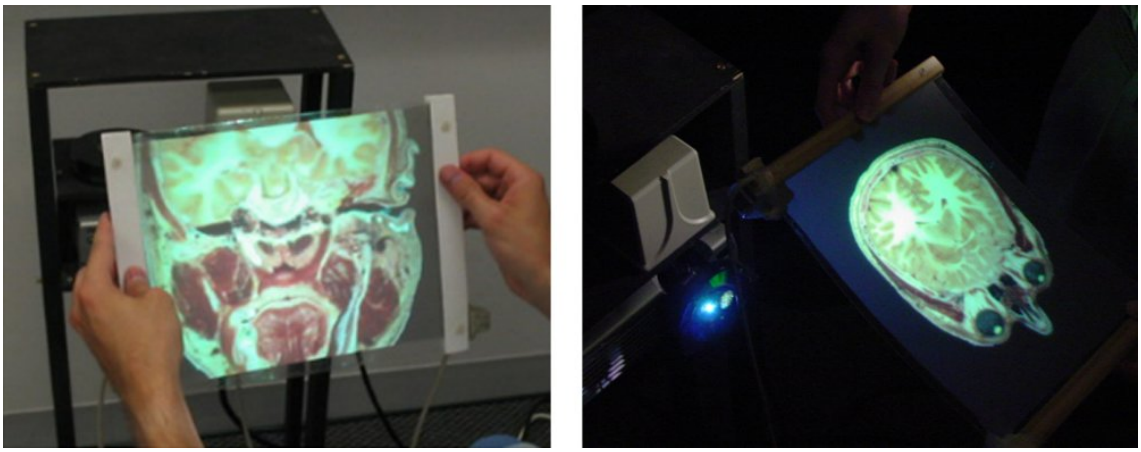

Figure 5.10: Two examples of cross sections of a 3D volume projected onto the flexible screen.

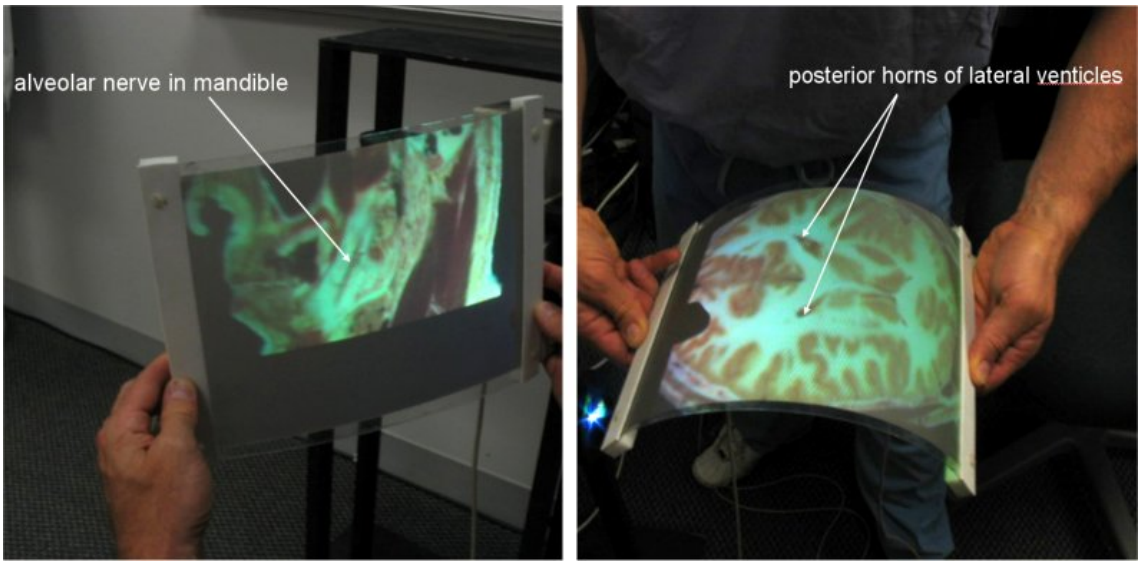

Figure 5.11: Utilizing curved slices through the visible human data set allows for larger portions of a desired data set to be brought into view at a single time.

5.4.2 Magic Window

While it is valuable to view cross sections as if you are actually holding the slicer, it can be somewhat confusing if the user is unable to see the volume prior to slicing into it. In order to help alleviate this issue, the concept of using the screen as a magic window is introduced. This approach involves the use of a projection screen as a window into the virtual world, allowing the user to see any virtual objects that are behind it.

In addition to aiding the visualization of 3D volumes, the use of the screen as a magic window can be useful as a standalone application. Any object with a geometric mesh can be viewed through the screen as if it was a real object in the environment. This could potentially be very useful in design. For example, determining how an object would look in a room before it was purchased or perhaps even created.

In order to do this, the geometry of the desired object is first loaded as in a normal graphics application. Then a viewport is created that matches with the projector screen

dimensions and viewer location. For the current setup, viewer location is assumed to be a certain distance directly behind the screen. However, an additional tracker can be used to allow an arbitrary viewer location. An easy way to think about this is that in a normal graphics program, the screen stays in place and all the virtual objects move. In this setup, the virtual objects all stay static and the screen itself moves.

Chapter 6

Conclusion

In this dissertation, new systems have been introduced for the design, simulation, and application of paint. These computer graphics programs provide the paint and coatings industry, which has previously used very little computer technology with new range of tools to both increase efficiency and to facilitate novel design and visualization procedures. While these systems are intended for the automotive paint industry, they also provide significant contributions in the field of computer graphics.

Chapter 2 described how to realistically simulate the spray paint application process. Specifically, a new spray particle simulation was developed, along with the ability to adjust the spray gun and paint settings. An accurate virtual environment is also provided, yielding a system that spray painters can use as a virtual analogue to actual paint application. As a virtual model mesh is painted with the system, a map of how the spray paint has been deposited on the object is updated. This map can then be used for various paint simulation, design, and training programs.

Chapter 3 enhances the spray system given in Chapter 2 by providing additional tools and improvements for proper simulation of airbrushes. Not only is this useful for practicing automotive decaling, but also provides a method for airbrush artists to paint

artwork directly onto virtual models. Use of this system can therefore be a new way of generating textures that have an artistic touch, allowing existing airbrush artists to create faux material effects to either supplement or replace traditional rendering techniques.

Chapter 4 gave details on enhancing current automotive paint rendering techniques. In particular, an industrial orange peel measurement instrument is used to create a virtual orange peel program. This program can be used to provide a photorealistic rendering of orange peel in real time. This chapter also showed that the orange peel simulation can be used to prototype surface finish tolerances for real paints. Finally, this software is used to study human detection of surface roughness under varying lighting conditions and with different material properties.

Chapter 5 provides new ways to display the simulations described in the previous chapters. The projection techniques presented in this chapter allow users to gain the best possible judgement of how virtual paint samples would appear once fabricated. In addition, these display systems can be used to visualize any material, not just automotive paint. Therefore, this research enhances the state of the art in augmented reality and visualization in addition to aiding the automotive and paint industries.

The work presented in this dissertation could potentially take several new directions. First, further research could be performed in rendering detailed paint appearance. While Chapter 4 yields an effective method for designing and simulating orange peel, other paint effects such as metallic flake sparkle could be explored. Both this phenomenon and orange peel also have significant visual variations based on viewing distance. Integration of these visual effects as viewing distance increases is currently performed using texture based mip-mapping. However, mip-mapping yields incorrect results when used on a BRDF (see Figure 6.1). Han et al. [31] gives a solution for the case when only normals are being integrated with viewing distance. While this case is probably sufficient for orange peel, it

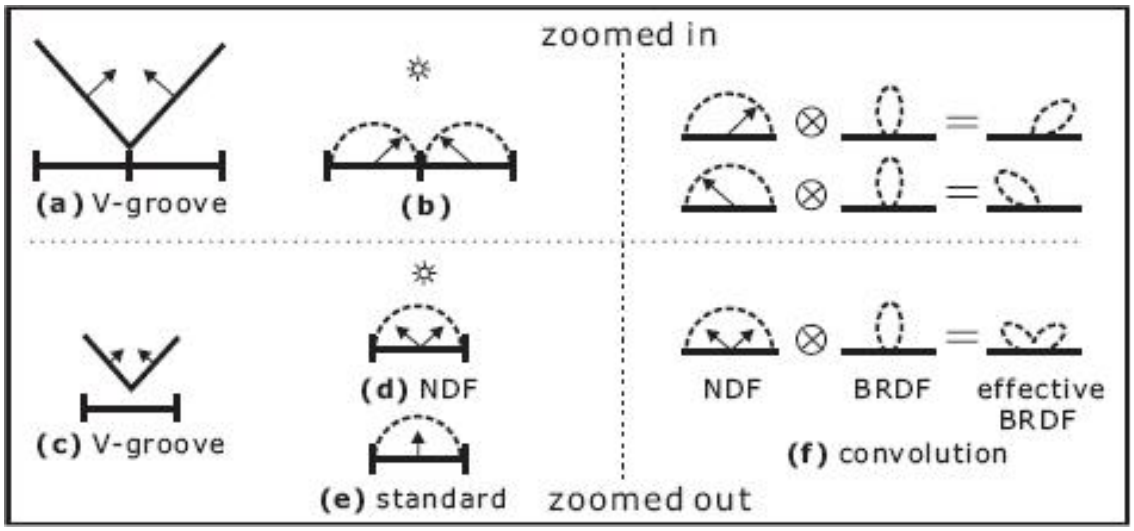

Figure 6.1: An example of the problem with using typical mipmapping to filter normal maps. In this case, a simple "V" groove is displayed on the left. A standard mipmap will filter the two normals to a mirrored normal direction, shown in (e). This is incorrect. The correct filtering is shown in (f), where the two BRDFs are sampled into a third BRDF that take both into account. Taken from [31]

may fail in more complicated cases such as metallic paint sparkle and marble. A general method for integrating BRDF data could be created to solve this problem, and utilized to accurately render paint sparkles.

The possibility of using the various devices as user interfaces and modeling tools could also be explored. For instance, when speaking with doctors, it was mentioned that the rear projection system shown in 5.3 could be altered for use as a medical atlas. Another possibility would be to modify the airbrush to "paint" other materials and textures onto a model. The airbrush might even be used as a model sculpting tool. Such additions could be implemented and tested for usability.

Finally, many portions of the system can be used to test human perception. For instance, the orange peel program could be used to test the visibility of surface roughness based on a number of factors such as bump frequency, amplitude, color, lightness, and

lighting environment. Therefore, the study given in Chapter 4 could be extended with more complex conditions in order to gain further insight into human perception. Another possibility stems from the fact that the front projection system presented in Chapter 5 was found to be quite convincing at long viewing distances, but less so up close. There are a few possible reasons for this, such as pixel resolution, depth of field, and stereo vision. Thus, the system could be tested with and without a stereo image in order to find at what distance the absence of stereo breaks the illusion.

This work demonstrates the potential of using advanced computer methods in industries which have previously made little use of computer technology in their process. Visits to paint facilities at Dupont and Lehmans Garage showed that the current methods for paint formulation and application are very "hands on." When someone wants to test out a new paint, a sample is painted and manually inspected to determine if it will be acceptable on the final product. Likewise, new paint formulations are tested by actually fabricating them and attempting to spray them onto test panels. This leads to a significant amount of wasted time and paint due to trial and error. With the introduction of computer graphics into the process, it is hoped that such trial and error can be eliminated, with the exact results predicted in advance. A consumer could then drive their car into a paint facility, use the projection techniques introduced in this dissertation to view a selected paint on their car, use the VR system described in this thesis to simulate both the spraying of the body color and the airbrushing of any decorations, inspect the virtual result for the visibility of orange peel, and eventually paint the vehicle knowing exactly how the finished product will look.

Bibliography

[1] Maneesh Agrawala, Andrew C. Beers, and Marc Levoy. 3d painting on scanned surfaces. In *SI3D '95: Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 145–ff., New York, NY, USA, 1995. ACM.

[2] Michael Ashikhmin and Peter Shirley. An anisotropic phong brdf model. *J. Graph. Tools*, 5(2):25–32, 2000.

[3] Bill Baxter, Vincent Scheib, Ming C. Lin, and Dinesh Manocha. Dab: interactive haptic painting with 3d virtual brushes. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 461–468, New York, NY, USA, 2001. ACM.

[4] William Baxter, Jeremy Wendt, and Ming C. Lin. Impasto: a realistic, interactive model for paint. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, pages 45–148, New York, NY, USA, 2004. ACM.

[5] James F. Blinn. Models of light reflection for computer synthesized pictures. In *SIGGRAPH '77: Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, pages 192–198, New York, NY, USA, 1977. ACM.

[6] C. Caiati. Advanced airbrushing techniques made simple. Tab Books, 1985.

[7] R. J. Castonguay. New generation high-speed high-resolution hemispherical scatterometer. *Optical Scattering: Applications, Measurements, and Theory II*, pages 152–165, 1995.

[8] Wei-Chao Chen, Jean-Yves Bouguet, Michael H. Chu, and Radek Grzeszczuk. Light field mapping: efficient representation and hardware rendering of surface light fields. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 447–456, New York, NY, USA, 2002. ACM.

[9] Nelson S. H. Chu and Chiew-Lan Tai. An efficient brush model for physically-based 3d painting. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 413, Washington, DC, USA, 2002. IEEE Computer Society.

[10] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graph.*, 1(1):7–24, 1982.

[11] Greg Coombe, Mark J. Harris, and Anselmo Lastra. Radiosity on graphics hardware. In *Proceedings of Graphics Interface 2004*, 2004.

[12] Sabine Coquillart and Gerold Wesche. The virtual palette and the virtual remote control panel: A device and an interaction paradigm for the responsive workbench((tm)). In *VR '99: Proceedings of the IEEE Virtual Reality*, page 213, Washington, DC, USA, 1999. IEEE Computer Society.

[13] Cassidy J. Curtis, Sean E. Anderson, Joshua E. Seims, Kurt W. Fleischer, and David H. Salesin. Computer-generated watercolor. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 421–430, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[14] Scott Daly. The visible differences predictor. In *Digital Images and Human Vision*, pages 179–206. MIT Press, 1993.

[15] Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Trans. Graph.*, 18(1):1–34, 1999.

[16] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 369–378, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[17] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 369–378, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[18] G. Dombek and T. Porter. Airbrush illustration for architecture. W. W. Norton and Company, 2003.

[19] Julie Dorsey and Pat Hanrahan. Modeling and rendering of metallic patinas. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, page 2, New York, NY, USA, 2005. ACM.

[20] Patricia Dumont-Bcle, Eric Ferley, Andras Kemeny, Sylvain Michelin, and Didier Arqus. Multi-texturing approach for paint appearance simulation on virtual vehicles. In *Proceedings of the Driving Simulation Conference*, pages 123–133, Sophia Antipolis, 2001.

[21] Roman Durikovic; and William L. Martens. Simulation of sparkling and depth effect in paints. In *SCCG '03: Proceedings of the 19th spring conference on Computer graphics*, pages 193–198, New York, NY, USA, 2003. ACM Press.

[22] Sergey Ershov, Roman Durikovic, Konstantin Kolchin, and Karol Myszkowski. Reverse engineering approach to appearance-based design of metallic and pearlescent paints. *The Visual Computer*, 20(8-9):586–600, 2004.

[23] Sergey V. Ershov, Andrei B. Khodulev, and Konstantin V. Kochin. Simulation of sparkles in metallic paints. In *Proceeding of Graphicon*, pages 121–128, 1999.

[24] Sergey V. Ershov, Konstantin V. Kochin, and Karol Myszkowski. Rendering pearlescent appearance based on paint-composition modelling. *Computer Graphics Forum*, 20(3), 2001.

[25] O. Figueiredo and R.D. Hersch. Parallel unfolding and visualization of curved surfaces extracted from large 3d volumes. In *Journal of Electronic Imaging*, volume 11, pages 423–433, 2002.

[26] J. Foley, A. van Dam, S. Feiner, and J. Hughes. Computer graphics: Principles and practice. *Addison-Wesley Publishing Co., ISBN 0-201-12110-7*, 1990.

[27] American Society for Testing and Materials. Standard practice for angle resovled optical scatter measurements on specular of diffuse surface. *American Society for Testing and Materials, Standard E 1392-96*, 1996.

[28] R. Furukawa, H. Kawasaki, k. Ikeuchi, and m. Sakauchi. Appearance based object modeling using texture database: aquisition, compression and rendering. In *Eurographics Workshop on Rendering*, pages 257–266, 2002.

[29] Jay S. Gondek. Fractal landscape synthesis in computer graphics. In *B.S. Thesis*, 1992.

[30] Chet S. Haase and Gary W. Meyer. Modeling pigmented materials for realistic image synthesis. *ACM Trans. Graph.*, 11(4):305–335, 1992.

[31] Charles Han, Bo Sun, Ravi Ramamoorthi, and Eitan Grinspun. Frequency domain normal map filtering. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 28, New York, NY, USA, 2007. ACM.

[32] Pat Hanrahan and Paul Haeberli. Direct wysiwyg painting and texturing on 3d shapes. *SIGGRAPH Comput. Graph.*, 24(4):215–223, 1990.

[33] Bruce A. Hartung. *Computer Graphics and Perception: Reaching to Virtual Objects and Material Perception*. PhD thesis, University of Minnesota, 2003.

[34] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 453–460, New York, NY, USA, 1998. ACM.

[35] Aaron Hertzmann. Paint by relaxation. Technical report, New York, NY, USA, 2000.

[36] Aaron Hertzmann. Fast paint texture. In *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, pages 91–ff, New York, NY, USA, 2002. ACM.

[37] Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 511–518, New York, NY, USA, 2001. ACM.

[38] Garrett M. Johnson and Mark D. Fairchild. Full-spectral color calculations in realistic image synthesis. *IEEE Comput. Graph. Appl.*, 19(4):47–53, 1999.

[39] Jan Kautz and Michael D. McCool. Interactive rendering with arbitrary BRDFs using separable approximations. In *Eurographics Workshop on Rendering*, pages 247–260, 1999.

[40] Gabriele Kigle-Bckler. Surface quality control on high and medium gloss surfaces: wave-scan dual. In *4th wave-scan User Meeting*, 2006.

[41] Daeseok Kim, Youngwoo Yoon, Sunyu Hwang, Geehyuk Lee, and Jinah Park. Visualizing spray paint deposition in vr training. In *Virtual Reality Conference*, pages 307–308, 2007.

[42] Jonathan Konieczny, John Heckman, Gary Meyer, Mark Manyen, Marty Rabens, and Clement Shimizu. Vr automotive spray paint simulation. In *ISVC '08: International Symposium on Visual Computing*, pages 998–1007, New York, NY, USA, 2008. Springer.

[43] Jonathan Konieczny and Gary Meyer. Material and color design using projectors. In *CGIV '06: Color in Graphics, Imaging, and Vision*, Springfield, VA, USA, 2006. IS&T.

[44] Jonathan Konieczny and Gary Meyer. Airbrush simulation for artwork and computer modeling. In *NPAR '09: Non-Photorealistic Animation and Rendering*, New York, NY, USA, 2009. ACM.

[45] Jonathan Konieczny, Clement Shimizu, Gary Meyer, and D'nardo Colucci. A handheld flexible display system. In *IEEE Visualization '05*, pages 95–101, Washington, DC, USA, 2008. IEEE Computer Society.

[46] Kui Chiu Kwok. *A Fundamental Study of Air Spray Painting*. PhD thesis, University of Minnesota, 1991.

[47] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 117–126, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[48] P. Lalonde and A. Fournier. A wavelet representation of reflectance functions. *IEEE Transactions on Visualization and Computer Graphics*, 3(4):329–336, /1997.

[49] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vision*, 43(1):29–44, 2001.

[50] Peter Litwinowicz. Processing images and video for an impressionist effect. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 407–414, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[51] J. L. Mannos and D. J. Sakrison. The effects of a visual fidelity criterion on the encoding of images. In *IEEE Transactions on Information Theory*, pages 525–535, Washington, DC, USA, 1974. IEEE Computer Society.

[52] S. Marschner, S. Westin, E. Lafortune, and K. Torrance. Image-based brdf measurement. *Applied Optics.*, 39:2592–2600, 2000.

[53] Gary A. Mastin, Peter A. Watterberg, and John F. Mareda. Fourier synthesis of ocean scenes. *IEEE Comput. Graph. Appl.*, 7(3):16–23, 1987.

[54] S. Maurello. The complete airbrush book. Wm. Penn Publishing, 1955.

[55] Michael D. McCool, Jason Ang, and Anis Ahmad. Homomorphic factorization of brdfs for high-performance rendering. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 171–178, New York, NY, USA, 2001. ACM.

[56] J. R. McNeil and S. R. Wilson. Two-dimensional optical scatterometer apparatus and process. *U.S. patent 5,241,369.*

[57] C. Misstear. The advanced airbrush book. Van Nostrand Reinhold, 1984.

[58] D. Mitchel. Airbrushing 101. ArtKulture/Wolfgang Publications, 2008.

[59] F. Kenton Musgrave. Uses of fractional brownian motion in modelling nature. In *SIGGRAPH 1991 Course Notes: Fractal Modeling in 3D Computer Graphics and Imaging*, pages 5–34, 1991.

[60] Addy Ngan, Frédo Durand, and Wojciech Matusik. Experimental validation of analytical brdf models. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Sketches*, page 90, New York, NY, USA, 2004. ACM.

[61] P. Owen and J. Sutcliffe. The manual of airbrushing. Thames and Hudson, 1985.

[62] Binh Pham. Expressive brush strokes. In *CVGIP: Graphical Models and Image Processing*, page 53, 1991.

[63] Bui Tuong Phong. Illumination for computer generated pictures. *Commun. ACM*, 18(6):311–317, 1975.

[64] Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. ilamps: geometrically aware and self-configuring projectors. *ACM Trans. Graph.*, 22(3):809–818, 2003.

[65] Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. The office of the future: a unified approach to image-based modeling and spatially immersive displays. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 179–188, New York, NY, USA, 1998. ACM Press.

[66] Ramesh Raskar, Greg Welch, Kok-Lim Low, and Deepak Bandyopadhyay. Shader lamps: Animating real objects with image-based illumination. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 89–102, London, UK, 2001. Springer-Verlag.

[67] D. Rudolf, D. Mould, and E. Neufeld. Simulating wax crayons. In *Proc. of Pacific Graphics*, pages 163–172, 2003.

[68] S. Saito and M. Nakajima. 3d physically based brush model for painting. In *SIGGRAPH99 Conference Abstracts and Applications*, page 226, 1999.

[69] L. Saroul, S. Gerlach, and R. D. Hersch. Exploring curved anatomic structures with surface sections. In Greg Turk, Jarke J. van Wijk, and Robert J. Moorhead, editors, *Proceedings of IEEE Visualization 2003*, pages 27–34. IEEE Computer Society, IEEE Computer Society Press, October 2003.

[70] Dieter Schmalstieg, L. Miguel Encarna, and Zsolt Szalav. Using transparent props for interaction with the virtual table. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 147–153, New York, NY, USA, 1999. ACM Press.

[71] Peter Schröder and Wim Sweldens. Spherical wavelets: efficiently representing functions on the sphere. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 161–172, New York, NY, USA, 1995. ACM.

[72] Helge Seetzen, Wolfgang Heidrich, Wolfgang Stuerzlinger, Greg Ward, Lorne Whitehead, Matthew Trentacoste, Abhijeet Ghosh, and Andrejs Vorozcovs. High dynamic range display systems. *ACM Trans. Graph.*, 23(3):760–768, 2004.

[73] Clement Shimizu, Gary W. Meyer, and Joseph P. Wingard. Interactive goniochromatic color design. In *Color Imaging Conference*, pages 16–22, 2003.

[74] Clement Shimizu, Gary W. Meyer, and Joseph P. Wingard. Interactive goniochromatic color design. In *Color Imaging Conference*, pages 16–22, 2003.

[75] Steve Strassmann. Hairy brushes. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 225–232, New York, NY, USA, 1986. ACM.

[76] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. pages 32–41, 1922.

[77] S. M. F. Treavett and M. Chen. Statistical techniques for the automated synthesis of non-photorealistic images. In *Proc. 15th UK Conference*, 1997.

[78] Michael Tsang, George W. Fitzmaurice, Gordon Kurtenbach, Azam Khan, and Bill Buxton. Boom chameleon: simultaneous capture of 3d viewpoint, voice and gesture annotations on a spatially-aware display. In *UIST '02: Proceedings of the 15th annual ACM symposium on User interface software and technology*, pages 111–120, New York, NY, USA, 2002. ACM Press.

[79] M. Alex O. Vasilescu and Demetri Terzopoulos. Tensortextures: multilinear image-based rendering. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 336–342, New York, NY, USA, 2004. ACM.

[80] G. Ward. A contrast-based scalefactor for luminance display. In *Graphics Gems IV*, pages 415–421, San Diego, CA, USA, 1994. Academic Press Professional, Inc.

[81] Gregory J. Ward. Measuring and modeling anisotropic reflection. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 265–272, New York, NY, USA, 1992. ACM.

[82] Stephen H. Westin, James R. Arvo, and Kenneth E. Torrance. Predicting reflectance functions from complex surfaces. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 255–264, New York, NY, USA, 1992. ACM.

[83] Harold B. Westlund and Gary W. Meyer. Applying appearance standards to light reflection models. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 501–51., New York, NY, USA, 2001. ACM Press.

[84] H. Wong and H. Ip. Virtual brush: A model-based synthesis of chinese calligraphy. In *Computers & Graphics*, page 24, 2000.

[85] S. Yamamoto, M. Tsurase, K. Ueda, N. Tsumura, T. Nakaguchi, and Y. Miyake. Reproducing an appearance of the objects using high bright projector. *Proceedings of 10th Congress of the International Colour Association*, pages 1043–1046, 2005.

[86] Ruigang Yang, David Gotz, Justin Hensley, Herman Towles, and Michael S. Brown. Pixelflex: A reconfigurable multi-projector display system. In Thomas Ertl, Ken

Joy, and Amitabh Varshney, editors, *Proceedings Visualization 2001*, pages 167–174. IEEE Computer Society Technical Committee on Visualization and Graphics Executive Committee, 2001.

[87] Ungyeon Yang, G.A. Lee, Seonhyung Shin, Sunyu Hwang, and Wookho Son. Virtual reality based paint spray training system. In *Virtual Reality Conference*, pages 289–290, 2007.