Interactive Goniochromatic Color Design

Clement Shimizu, Gary W. Meyer, and Joseph P. Wingard
Department of Computer Science and Engineering, University of Minnesota
Minneapolis, Minnesota

Abstract

An interactive program has been developed to assist in the design of new goniochromatic colors. The program gives the user a unique set of controls over a second order polynomial that defines these color families at a sequence of aspecular reflection angles. One approach, based on traditional metallic colors, allows the user to adjust the average hue, saturation, and brightness of all of the colors interpolated by the polynomial. Another method, appropriate for the newer effect colors, permits the designer to establish face and flop colors to be reached at either end of the interpolation. In a final technique, variations produced by adjusting model parameters can be evaluated and selected.

1.0 Introduction

Computer graphics is on the verge of being able to provide computer aided design tools to color appearance professionals. These tools will let industrial designers and color technologists have the same interactive control over color appearance that engineers and architects have had over geometry since the beginning of computer graphics almost forty years ago. The emergence of computer aided color appearance design (CACAD) will allow color appearance designers and scientists to examine how existing paints and coatings look on new products. More importantly, CACAD will make it possible to hypothesize and visualize new surface coatings with heretofore unseen reflection properties (Meyer, 2000).

The recent computer graphic advance which has made this possible is the development of per pixel shading hardware that permits display, in real time, of objects with arbitrarily complex surface reflection functions. While earlier graphics devices evaluated a limited shading model at polygon vertices and interpolated the result across the polygon's interior, the new hardware can compute a complex reflectance model at every pixel that composes a polygon. This makes it feasible to treat surface reflection in a very general manner, including the use of a bidirectional reflectance distribution function (BRDF) (Heidrich and Seidel, 1999; Kautz and McCool, 1999) and wavelength based color calculations.

While the ability of computer graphics hardware to display complex surface reflection has been improving, new "effect" paints are being developed for use in a variety of different design applications. These so-called goniochromatic colors have the property that the color of the reflected light changes with the angle of reflectance. Metallic and pearlescent automotive paints are two examples that are in wide use. The appearance of these paints is difficult to characterize because a single color measurement is not sufficient. This complicates the job of a designer who invents new goniochromatic colors, because they must specify not just one color but a whole family of colors.

In this paper we describe an initial attempt to develop a CACAD program for designing and visualizing hypothetical goniochromatic colors. The work is based upon an existing reflection model for metallic paint which assumes that the variation in metallic color with reflectance angle can be fit with a second order polynomial. The program gives the designer several unique ways to modify this second order curve. These techniques range from editing the individual data points to which the curve is fit, to adjusting the average hue and saturation of the color represented by the curve, to selecting the so-called *face* and *flop* colors that the curve must interpolate near specular and far from specular. The program uses per pixel shading hardware to allow the designer to see the new color in real time on a three dimensional surface. The new color's variation in reflectance with aspecular angle can be written to a file for manufacturability analysis.

2.0 Metallic Reflection Models

It is well known in the field of computer graphics that the reflection of metallic objects is primarily specular (Cook and Torrance, 1982). In terms of appearance, this means that metallic surfaces appear brightest when viewed in the specular direction and become much darker as the line of sight shifts away from specular. Metallic paints exhibit this property even though it results from a directional diffuse component that comes from below the surface. The paint and coatings industries refer to how lightness changes with viewing direction as the flop or travel of the color. This property of metallic colors accentuates an object's curvature and may account for the popularity of metallic automotive finishes (Rodrigues, 1995).

Our program employs a shader that can render the appearance of a metallic paint from a limited number of spectraphotometric measurements. Alman (1987) performed a systematic study of metallic paint which revealed that only a few data points need to be taken in order to characterize a metallic color's flop. He found that a second order polynomial could successfully interpolate CIE Lab color coordinates derived from in plane goniospectrophotometric measurements. To fit the curve, he suggested that only measurements at specular, near grazing, and one angle in between were necessary. Others confirmed his results (Rodrigues, 1990; Saris, et al., 1990; Venable, 1987). We utilize a shader that can interpolate a set of such measurements and render the appearance of metallic automotive paint (Westlund and Meyer, 2001).

2.1 Control of the Metallic Reflection Model

In the program, a metallic color is represented using a quadratic curve that plots the reflected Lab tristimulus values of the color as a function of the angle θ from the *specular* direction (the *aspecular* angle). This curve is initially determined using measured data points from an existing metallic color. The user interface, described in Section 5.0, allows the data points to be adjusted and the curve to be refit. The curve itself is represented as a clamped quadratic polynomial. The control points of this type of curve provide a different way for the user to change the shape of the curve and define a new metallic color.

A clamped quadratic polynomial is a two piece function. The first piece is a parabolic curve from angle zero to the apex of the parabola, referred to here as the clamp point. The second piece starts at the apex and remains a constant value across the remaining angles. The control points of the clamped quadratic polynomial occur at the angle zero and the apex of the quadratic. In the context of automotive paint, this means that the goniochromatic color shifts at a quadratic rate through a range of off specular angles, and the color remains constant for the rest of the angles.

Face is the name for the color at angles close to specular. Flop is the name for the base color or the color that appears at viewing angles far from specular. Travel refers to the rate at which the color shifts from face to flop. Flop, face, and travel colors directly correspond to the control points of the clamped quadratic polynomial defining our internal representation of color.

Since the user interface uses data points, clamped quadratics, and control points all to represent the same color appearance, it is necessary to convert information between the three modes. Using a least squares approximation method, the coefficients to the clamped quadratic polynomials can be found that best interpolate a set of data points. It is important that the least squares approximation method fits the clamped quadratic curve to the data points, and not to a standard quadratic polynomial. The clamped quadratic polynomial can be evaluated at specific angles to obtain new data points. Evaluating the clamped quadratic polynomial at the angle zero and the apex of the quadratic give the control points. The clamped quadratic can be solved for directly given its control points. We work through the mathematical details only for the L curve, but the a and b curves are done similarly.

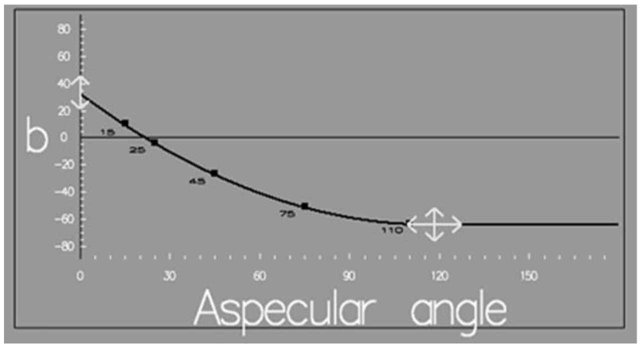

Figure 1. The clamped quadratic curve's data points are shown as black dots and control points are shown in yellow.

$$L(\theta) = L_a\theta^2 + L_b\theta + L_c \qquad (1)$$

Setting the derivative of the quadratic polynomial to be zero and solving for θ, we can find the clamp point so we can evaluate the clamped quadratic polynomials.

$$\begin{aligned} &\frac{\partial}{\partial\theta}L(\theta) = 0 = 2 * L_a\theta + L_b \\ &\theta_{Clamp} = -L_b /(2 * L_a) \\ &L_{ClampedQuadratic}(\theta) = L\left(\min(\theta, \theta_{Clamp})\right) \end{aligned} \qquad (2)$$

We specify the control points of the L, a, and b curves to be *(0, L(0))* and *(θ_{Clamp}, L(θ_{Clamp}))*. Normally three control points would be needed to define a quadratic polynomial, but since a quadratic polynomial is symmetric about its apex, the clamp point counts for two control points. In order to obtain the clamped quadratic polynomial from the control points, insert the control points into the quadratic formula and solve for the coefficients. The solution is shown below.

$$\begin{aligned} L_c &= L(0) \\ L_a &= \frac{L_c - L(\theta_{Clamp})}{\theta_{Clamp}^2} \\ L_b &= -2 * L_a * \theta_{Clamp} \end{aligned} \qquad (3)$$

2.2 Complete Metallic Reflection Model

Our program implements an interactive version of the metallic reflection model developed by Westlund and Meyer (2001). This model treats the overall reflection from a metallic paint as a simple linear combination of a subsurface and a first surface reflection. The subsurface reflection is modeled using the clamped second order polynomial described in Section 2.1. Because we start with Lab data, a tristimulus version of a BRDF is constructed from the three separate L, a, and b curves. This BRDF is addressed using the aspecular angle θ determined from

$$\theta = \operatorname{acos}((\theta_i - 2 * (\theta_i - \bar{n})) \bullet \theta_r) \qquad (4)$$

where θ_i is the incident direction, θ_r is the reflection direction, and $\bar{n}$ is the surface normal. The first surface reflection is modeled using a modified Phong reflection model (Lewis, 1993) with the specular exponent selected from tables in Westlund and Meyer (2001) so as to simulate a particular ASTM standard 60 degree gloss value.

3.0 Overview of Graphics Shading Hardware

Before discussing how the metallic reflection model was implemented in our interactive design program, we provide some background on the development of computer graphics shading hardware. Until recently, computer graphics shading hardware was quite limited in its capabilities. There was only a single simple reflection model available (the Phong model), this model could only be evaluated at the vertices of a triangular mesh, and the interior of each triangle had to be filled in by interpolation. This shading approach, called Gouraud shading, was good enough for traditional Computer Aided Geometric Design, but it is too limited for CACAD.

Over the last decade advanced texture mapping techniques such as environment mapping and BRDF decomposition have provided some alternatives to simple Gouraud shading. Some of these methods achieved their results by making creative use of the limited functional extensions provided by video card manufacturers. Although these advanced texture mapping techniques can be rendered in real-time, most require a preprocessing step that cannot be computed in real-time.

The recent increased availability and capability of programmable shaders is a big advancement in computer graphics hardware relative to CACAD. This improvement makes it possible to replace the default shading model computed at the vertices of triangles, and to also do a per pixel calculation across the face of each triangle. Since programmable shaders can be passed variables, this allows for real-time rendering and design of arbitrary reflection models.

3.1 Environment Mapping

Texture mapping was originally introduced to enhance Gouraud shaded scenes. The diffuse color rendered with Gouraud shading can be modulated by an image or texture map. In effect, using a texture map in this manner is like applying vinyl wallpaper or wood veneer to a surface.

Environment mapping makes use of texture mapping hardware to simulate a perfect mirror surface reflection. From the eye, the reflected vector off the surface is computed and used to index into an environment map. The environment map can hold an image of the sky and other landscape around the object to simulate ray tracing. It can also hold images of light sources, to simulate area light source reflections or approximate per pixel Phong lighting. Finally, the environment map can be pre-filtered to simulate how a non-perfect mirror surface will naturally scatter light.

3.2 BRDF Decomposition

A BRDF is a function which describes the ratio of incoming light to outgoing light for every possible incoming and outgoing light direction. This is the most general way of specifying surface reflection. It is, however, a complicated four dimensional function, making it impractical for interactive rendering.

Kautz and McCool (1999) present an ingenious method of approximating arbitrary BRDFs thereby allowing them to be rendered in real-time. In their approach, the four dimensional BRDF is represented as the sum of products of two dimensional matrices. These matrices can be encoded as texture maps and the rendering can be accelerated using hardware extensions.

The use of separable BRDF decomposition is a step towards CACAD since it allows interactive display of arbitrary BRDFs. We utilized this technique in an earlier version of our program. However, because the decomposition can't be done at interactive rates, this method isn't suitable for use in an interactive design tool. For interactive design it is important for the rendering to immediately update as changes are made to the BRDF.

3.3 Programmable Shaders

Programmable shaders allow an arbitrary shading model to be evaluated, on every refresh cycle, at each pixel of the display screen. However, the shading model must be uploaded to the video card in a video card specific machine language. Although the machine language can be difficult to program, development tools permit the shader to be written in a portable high level language similar to C and compiled into machine language. Some of the emerging standards for writing vertex and pixel shaders are OpenGL 2.0, NVIDIA CG, and DirectX 9.0.

A programmable shader has two separate parts: a vertex shader and a pixel shader. The vertex shader has access to the vertex position, vertex normal, transformation matrices, and light position, as well as user definable per vertex and per object parameters. The vertex shader's input parameters are evaluated at the vertices of an object's triangle mesh. The vertex shader's output values are interpolated across the inside of each mesh triangle, and the pixel shader takes these values as input and evaluates the rest of the reflection model. In addition, the pixel shader can access texture maps. The final color for the pixel is rendered to the screen by the pixel shader.

The number of instructions that the vertex and pixel shaders can execute is a limiting factor in designing programmable shaders. Depending on the video card, 128 to 1024 machine instructions can be evaluated in the vertex shader and on the order of 128 machine instructions can be executed in the pixel shader. Any part of the reflection model that can be linearly interpolated should be put in the vertex shader, to offload work from the pixel shader. Fortunately, since the video card market is highly competitive, the vertex and pixel shader instruction limit is rapidly increasing.

Programmable shaders are a big step forward in making CACAD a reality. If we can model a complex color appearance using a programmable shader, we can interactively render objects that have this color appearance. Since programmable shaders can be passed variables, the interactive design of complex color appearances becomes possible.

4.0 Rendering the Metallic Reflection Model

The implementation of our metallic reflection model, discussed in section 2.0, makes extensive use of environment mapping and programmable shaders. We use a vertex shader to evaluate the second degree Lab polynomials and to convert the Lab values to RGB space. Using a prefiltered environment map computed offline, the pixel shader determines the first surface reflection and mixes this with the color interpolated from the vertex shader. Finally this is displayed on the screen.

We use prefiltered environment maps in order to simulate a first surface reflection that is dependent on the gloss value. Since pre-filtering the environment map takes a long time, it is computed offline for the entire range of gloss values. The gloss values are converted into Phong specular highlight parameters by a method introduced by Westlund and Meyer (2001). The environment map is filtered with the Phong highlight parameters using a simple method discussed in Akenine-Moller and Haines (2002). These prefiltered environment maps are saved to disk and can be loaded very quickly as the user changes gloss values during rendering. Next an area light source, also filtered by the Phong coefficients, is inserted into the prefiltered environment map. This is uploaded to the video card for use in the pixel shader.

As a three-dimensional object is rendered, the triangles it is composed of are sent to the vertex shader. The vertex shader takes as inputs the Lab polynomial, the vertex position, vertex normal, viewer direction, and the light direction. The aspecular angle between the viewer and the reflected direction of the light source is computed. The aspecular angle is clamped to be between zero and the clamp point of the Lab polynomials. This clamped aspecular angle is used to evaluate the Lab polynomial. Noting the reference white point, the Lab values are converted into XYZ space. Noting the monitor's white point, phosphor chromaticity, and gamma, the XYZ values are converted to RGB. The RGB color and the vertex normal is output to the pixel shader.

As vertices of the triangles are sent through the vertex shader, the vertex shader's output values are interpolated across the inside of the triangles and passed to the pixel shader. The pixel shader computes the reflected vector from the viewer and uses this to index the reflection map for the current gloss value. Finally this reflected color is mixed with the vertex color and rendered to the screen.

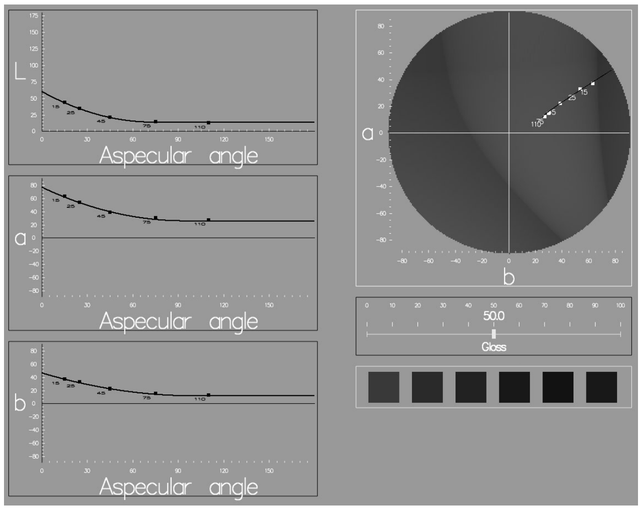

Figure 2. The user interface.

5.0 Interface

This section is a description of the graphical user interface (GUI) components for the design of goniochromatic colors. The components discussed below are the L, a, and b plots, the Hue Saturation Brightness control, the Face, Flop, and Travel Picker, the Variations mode, the Gloss slider, and the Color Palette. Before we talk about the individual components of the user interface we should mention some of our design goals. Our program must be made useable for a wide range of users, including chemists, designers, and consumers. Our main issue with designing a user interface is how to control the complicated definition of the surface appearance in a user friendly way. On one hand we would like to give the user access to the complete range of colors possible, but on the other hand we should avoid overwhelming the user with a complicated interface. Our software explores a variety of options for designing goniochromatic colors, where some methods are more appropriate for certain types of users than others. Many ideas were taken from methods of dealing with traditional monochromatic colors and extended to work with the family of colors that defines a color shift.

5.1 Data Point Editor

The program's internal representation of a goniochromatic color consists of data points interpolated in Lab space as a clamped quadratic polynomial. Displaying these data points on separate L, a, and b plots, and allowing the user to move the data points up and down, provides a very rudimentary way of interacting with the color definition (see Figure 2).

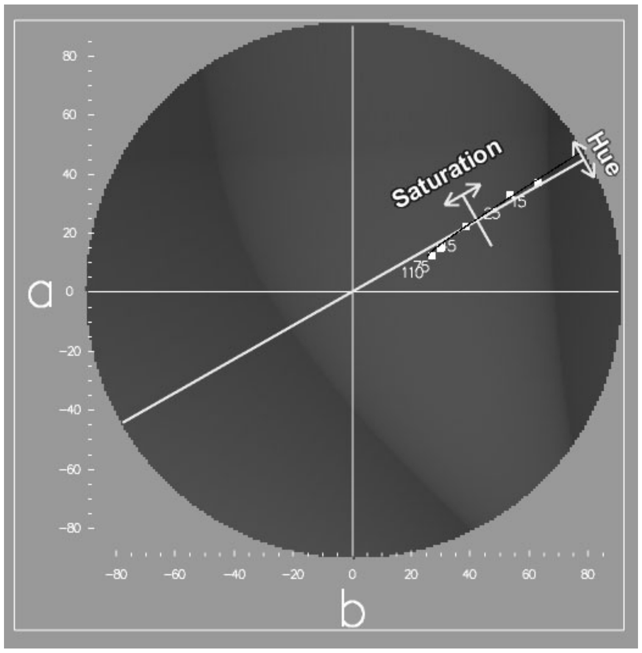

Figure 3. Hue and saturation control

.

As a point is moved, the clamped quadratic polynomial that interpolates the data points is updated in real-time.

It is difficult for most users to relate the separate L, a, and b curves to the final color they represent. Combined, a and b define the hue and saturation of a color while L by itself corresponds to the color's brightness. A color wheel superimposed over an a versus b plot permits the user to edit the individual data points in the familiar space of hue and saturation. The one dimensional a and b components are combined to become a two dimensional set of data points on the a versus b plot. This locus of data points forms a slightly curved line through a versus b space. The colors that the locus passes through represent the goniochromatic color shift across aspecular angles.

Editing the data points directly is tedious since the goniochromatic color can be sampled at several different viewing angles in Lab space. Defining a completely new color requires the user to move many data points. As a result, the main use of this control would be to enter color data sampled from a goniospectrophotometer.

5.2 Hue, Saturation, and Brightness Control

The color shift of a traditional metallic automotive color consists primarily of a change in saturation and brightness while hue remains constant. This can be seen in Figure 3 where the locus of points for a metallic red can be approximated by a straight line that extends radially outward from the center of the a versus b plot. This is a line of constant hue on the a versus b diagram.

The hue, saturation, and brightness control allows the user to edit, as a group, the family of colors that defines a goniochromatic color shift. Moving the saturation control in and out from the center of the a versus b plot changes the average saturation. Adjusting the angle of the hue control rotates the color to a new average hue. This is implemented using rotate and scale linear transformations upon the original data points. The resulting color appearance will have a different color, but it will retain the relative spacing between the color control points. The user manipulates brightness by moving a line drawn through the average value of the L plot.

The hue, saturation, and brightness method is useful for adjusting a traditional metallic paint to see what it would look like in a different hue. This approach has the disadvantage of only permitting the user to change the average hue and saturation; it does not allow the user to independently adjust the color appearance across different viewing angles.

5.3 Face, Flop, and Travel Color Picker

The locus of data points on the a versus b plot forms a straight line segment through ab space. The face, flop, and travel color picker allows the user to control the endpoints of the locus individually as well as move the entire locus around as a unit. Face, flop, and travel colors correspond directly to the control points of the clamped quadratic polynomials used to represent a goniochromatic color.

The user interface displays an arrow on the a versus b plot that corresponds to the control points of the a and b clamped quadratic polynomials as seen in Figure 4. The arrowhead is positioned on the face color and can be moved to define the color that occurs in the specular direction. The base of the arrow is situated on the flop color and can be adjusted to establish the color that is produced far away from specular. The arrow can also be dragged as a complete unit. Controls for the face and flop colors are also displayed

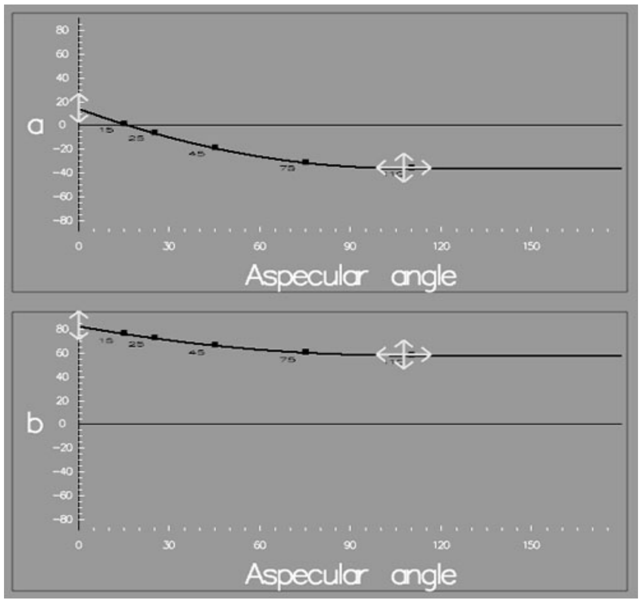

Figure 4. Modifying a color by its control points.

on the L, a, and b plots. The travel can be modified through the L, a, and b plots by dragging the flop/travel control horizontally.

This method is the best way to design a metallic color with dramatic changes between the face and flop colors. These “effect” colors are becoming more commonplace as paint technology continues to improve. Since this method of dragging the ends of the arrowhead and adjusting the control points on the L, a, and b plots are complex tasks, the face, flop, and travel color picker may be difficult for people without training.

5.4 Variations Control

We would like to have a method for designing a color that allows a novice computer user full control over all aspects of the color in a straightforward manner. The Variations mode is simple yet powerful goal-oriented approach for designing color. Starting with any color definition, that color is rendered on an object surrounded by variations on that color (see Figures 6 and 7). The user clicks on any variation that is a closer match to the desired appearance. The colors are then updated. This cycle is continued until the user reaches a color with which they are satisfied. Since there are so many variations on a color, the variations are arranged into the categories face color, flop color, and travel and gloss. The degree of variation can also be modified by the user.

There are strengths and weakness to the Variations method. This method is good for novice computer users, since pointing to the variation they prefer and clicking is all that is required. Since variations on all aspects of the color definition are displayed, the variation mode is powerful enough that a user can achieve any color that our system can render. The main drawback is that it takes many clicks to design a completely new color, so a person familiar with the program’s other modes of operation may find variations too tedious. Adobe Photoshop also has a variations mode. It is worth noting that an artist suggested the use of this method.

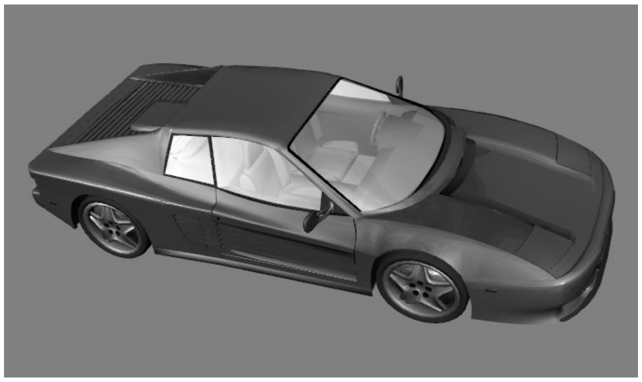

Figure 5. The color corresponding to Figure 4.

5.5 Gloss Control

To design gloss, the user moves a gloss slider from 0 to 100. The values are based on industry standards designed to be perceptually uniform. From the single gloss value, the program then chooses both the specular exponent and weight parameters to the Phong model using a method that was introduced by Westlund and Meyer (2001). The specular exponent and weight are constrained so as to be energy conserving.

5.6 Color Palette

Since a normal color palette does not convey the information needed to fully describe the color shift and gloss of a metallic paint, we must extend the idea of a color palette to work with color shifting and gloss. We have chosen to make a gradient bar that shows the color shift across viewing angles. The gloss value is displayed by the brightness of the border surrounding the color bar. This allows the entire color description to be displayed in compact form for display in a color palette.

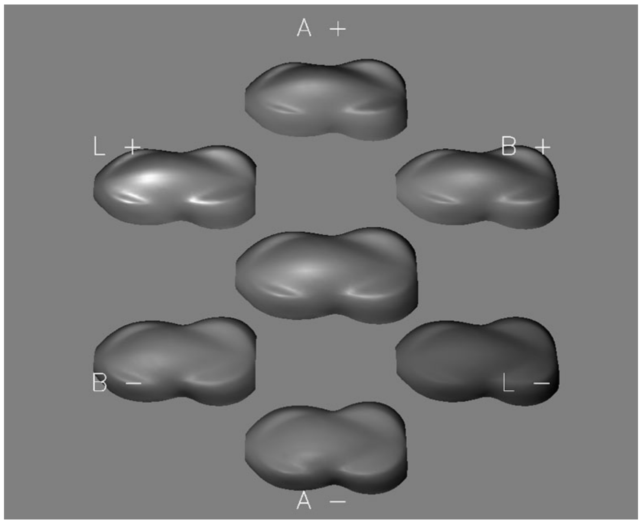

Figure 6. Variations on the face color.

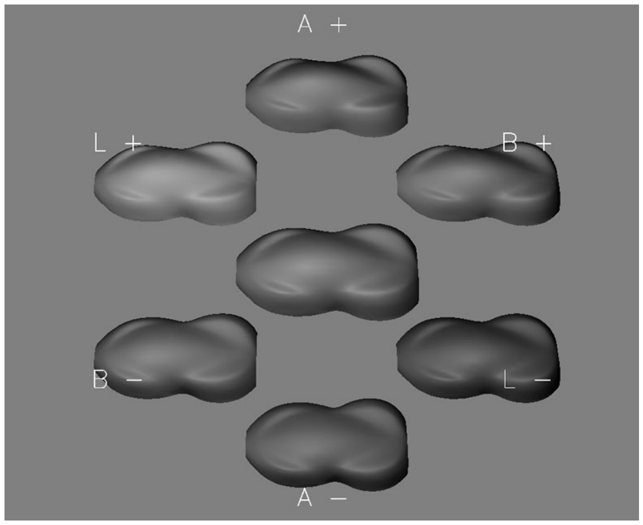

Figure 7. Variations on the flop color.

6.0 Conclusions

An interactive program has been written to assist in the design of new goniochromatic colors. The program is based upon an existing reflection model for metallic automotive paints that utilizes a second order curve to interpolate a set of aspecular measurements. The designer is given an interface that allows them to deal with the fact that goniochromatic colors are not a single color but an entire family of colors. One approach to the design problem treats the group of colors as having an average hue, saturation, and brightness. This is most successful for traditional solid metallic colors and some simple pearlescent paints. Another method provides the designer with interactive control over the color found near the specular direction and the color found far from it. This works best for the newer "effect" paints that are becoming more commonplace. Finally, the user of the program is given an interface that allows them to easily see and select variations of the color that are the result of small changes in the parameters of the model.

The goal of this research was to explore the user interface for an interactive CACAD program that could be tied to existing paint formulation software. This is similar to how computer aided geometric design evolved, with a front end interactive mechanical design program feeding its results to back end stress and thermal analysis software. By utilizing the best publicly available model for metallic automotive paint we have tried to impose constraints on the solution that improve the chances of manufacturability. However, our program must incorporate a more comprehensive model of effect paints to guarantee that the more extreme goniochromatic colors can be fabricated. Alternatively, based on practical knowledge of paint formulation, one could provide the designer with specific suggestions that steer them towards "reachable" colors. The risk, of course, in constraining the solution too tightly is that aesthetically interesting and physically plausible color appearances will be overlooked and the full potential of CACD to push back the boundaries of color appearance design will not be realized.

Acknowledgements

The authors would like to thank DuPont Automotive Products for providing paint samples and reflectance measurements. Harold Westlund assisted in an initial implementation of the metallic reflection model. This research was partially funded by the National Institutes of Standards and Technology, and was performed at the University of Minnesota Digital Technology Center.

References

1. Tomas Akenine-Moller and Eric Haines, Real-Time Rendering. Second Edition, A.K. Peters, 2002.
2. David H. Alman, Directional color measurement of metallic flake finishes, In Proceedings of the ISCC Williamsburg Conference on Appearance, pg. 53–56, (1987).
3. Jan Kautz and Michael D. McCool, Interactive rendering with arbitrary brdfs using separable approximations, In Tenth Eurographics Workshop on Rendering, pg. 281-292, (June 1999).
4. Robert Lewis, Making shaders more physically plausible, Fourth Eurographics Workshop on Rendering, pp. 47-62. (June 1993).
5. Gary W. Meyer, Computer aided color appearance design, In Proceedings of the First International Conference on Color in Graphics and Image Processing, (2000).
6. Gary W. Meyer, Harold B. Westlund, Peter A. Walker, and Joseph P. Wingard, A Computer Graphic System for Rendering Gonio-Apparent Colors. Proceedings of the AIC Color 01 Congress, (2001).
7. Allan B. J. Rodrigues, Color vision in instrumental color matching, 16th International Conference in Organic Coatings, (1990).
8. Allan B. J. Rodrigues, Measurement of metallic and pearlescent finishes, Die Farbe, 37:65-78, (1990).
9. Allan B. J. Rodrigues, Color and appearance measurement of metallic and pearlescent finishes, ASTM Standardization News, 23(10):68-72 (1995).
10. H. J. A. Saris, R. J. B. Gottenbos, and H. van Houwelingen, Correlation between visual and instrumental colour differences of metallic paint films, Color Research and Applications, 15(4) (1990).
11. William H. Venable, A model for interpreting three-angle measurements of flake finishes, In Proceedings of the ISCC Williamsburg Conference on Appearance, Pages 57-60. (1987).
12. Harold B. Westlund. Appearance Based Rendering, Master's Thesis, University of Oregon, June 2001.
13. Harold B. Westlund and Gary W. Meyer, Applying Appearance Standards to Light Reflection Models, Proceedings of SIGGRAPH 2001, pp. 501-510. (2001).

Biography

Clement Shimizu received his B.S. degree in Mathematics and is currently pursuing a graduate degree in Computer Science from the University of Minnesota. He works as a research assistant in Computer Graphics at the U of M Digital Technology Center. His research interests include hardware accelerated computer graphics techniques, BRDF design and display, and Fourier analysis of audio.