Workshop on Material Appearance Modeling (2018)
H. Rushmeier and R. Klein (Editors)

A Method for Fitting Measured Car Paints to a Game Engine's Rendering Model

Tom Kneiphof[1] and Tim Golla[1] and Michael Weinmann[1] and Reinhard Klein[1]

[1] University of Bonn, Germany

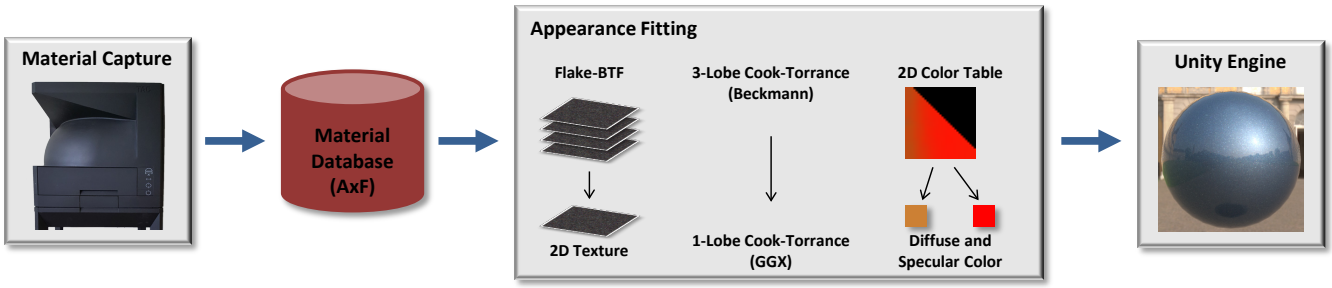


Figure 1: *We present an approach for efficiently integrating measured car paint materials into state-of-the-art game engines.*

Abstract

Car paints are visually complex materials that are of great importance for numerous real-time applications, including not only the game and movie industries but also virtual prototyping and design as well as advertisement. In addition to the creation of plausible materials by designers, more and more industrial effort is spent on capturing large databases of digitized materials. However, capturing complex reflectance characteristics of car paints involves the use of specialized, commercially available devices that come with predefined, standardized material formats. Using these digitized materials within other frameworks such as game engines is a challenging task due to the lacking compatibility of the involved rendering models. In this paper, we address these compatibility issues by fitting the available parameters of the game engine's material model to best match the appearance of the measured material.

1. Introduction

Virtual materials are of great importance for numerous applications in entertainment (i.e. game and movie industry), virtual prototyping and design or product advertisement. Many scenarios rely on the photo-realistic reproduction of the characteristic *look* and *feel* of the materials of the scene contents to create an immersive user experience. Besides occupying designers for the creation of plausible materials, there is also a demand for integrating accurately digitized real-world materials, in particular for design/prototyping or advertisement applications. Therefore, several companies spent huge efforts in capturing large databases of digitized materials. Among others, the materials of interest include also car paints.

Digitizing materials with a high degree of realism involves the use of special hardware setups. One such device that is commercially available and meets the industry standards is X-Rite's TAC7 device. The data measured with this device is stored in the AxF file format [ML15]. AxF has already been included in several commercially available software packages like Autodesk VRED, Nvidia Iray or X-Rite Pantora. Golla and Klein [GK17] presented a method for compressing the memory-consuming AxF car paint metallic flake BTF representation.

For car paints, the AxF format uses a hybrid representation that combines a multi-lobe BRDF model, color lookup table and a BTF for the effects caused by the metallic flakes in car paints, which is fitted to the real-world measurements. It is desirable to use these materials in a wide variety of applications. However, including such measured materials within current game engines like Unity or the Unreal Engine is not directly possible due to compatibility issues.

In this paper, we focus our attention on the efficient integration of measured materials given in industry-standard file formats within

DOI: 10.2312/mam.20181196

commonly used game engines at the example of integrating car paints given in the AxF format into the Unity game engine. This allows making use of the efficient rendering system of the game engine. However, this also imposes tight constraints regarding the material representations that can be handled. For this purpose, we investigate the engine's underlying BRDF model and fit its parameters so that the resulting appearance approximates the data of the AxF car paint model well. We demonstrate an automatic procedure to import AxF car paints into the Unity engine and show initial results.

2. Preliminaries

2.1. Basic Notation

In this section we briefly outline the used notation. Vectors are written in boldface (e.g. $\mathbf{x}$). The following vectors are defined in the local tangent space:

- $\mathbf{i}$ denotes the normalized incoming direction.
- $\mathbf{o}$ denotes the normalized outgoing direction.
- $\mathbf{h}$ is the half vector $\frac{\mathbf{i}+\mathbf{o}}{||\mathbf{i}+\mathbf{o}||}$.
- $\mathbf{n}$ denotes the surface normal.

We denote the scalar product of two vectors $\mathbf{u}$, $\mathbf{v}$ as $\mathbf{u}\cdot\mathbf{v}$. Since $\mathbf{n}=(0,0,1)^t$, we often write $\mathbf{u}_z$ instead of $\mathbf{n}\cdot\mathbf{u}$.

2.2. AxF Car Paint Model

The model used in the AxF file format [ML15] is largely based on the work of Rump et al. [RMS*08] and consists of a combination of multiple models:

- A measured clear coat layer, that changes $\mathbf{i},\mathbf{o}$ to $\bar{\mathbf{i}},\bar{\mathbf{o}}$, depending on the thickness and refraction index of this layer.
- A Lambertian BRDF $\frac{a}{\pi}$.
- A multi-lobe Cook-Torrance BRDF [CT82], where the k-th lobe is defined as:

$$f^{CT}_{s_k,\alpha_k,F_{0,k}}\left(\bar{\mathbf{i}},\bar{\mathbf{o}}\right)=\frac{s_k}{\pi}\frac{D_{\alpha_k}(\bar{\mathbf{h}})F_{F_{0,k}}\left(\bar{\mathbf{h}},\bar{\mathbf{o}}\right)G\left(\bar{\mathbf{i}},\bar{\mathbf{o}}\right)}{\bar{\mathbf{i}}_z\bar{\mathbf{o}}_z}, \tag{1}$$

  where s_k is the specular coefficient,

$$D_{\alpha_k}(\bar{\mathbf{h}})=\frac{1}{\alpha_k^2\bar{\mathbf{h}}_z^4}e^{\frac{\bar{\mathbf{h}}_z^2-1}{\bar{\mathbf{h}}_z^2\alpha_k^2}} \tag{2}$$

  is the microfacet distribution,

$$F_{F_{0,k}}(\bar{\mathbf{h}},\bar{\mathbf{o}})=F_{0,k}+(1-F_{0,k})(1-\bar{\mathbf{h}}\cdot\bar{\mathbf{o}})^5, \tag{3}$$

  is Schlick's approximation [Sch94] of the Fresnel term and

$$G(\bar{\mathbf{i}},\bar{\mathbf{o}})=\min\left(1,\frac{2\bar{\mathbf{h}}_z\bar{\mathbf{o}}_z}{\bar{\mathbf{h}}\cdot\bar{\mathbf{o}}},\frac{2\bar{\mathbf{h}}_z\bar{\mathbf{i}}_z}{\bar{\mathbf{h}}\cdot\bar{\mathbf{o}}}\right) \tag{4}$$

  is the geometry term.
- A 2D color table $\chi(\theta_{\bar{\mathbf{h}}},\theta_{\bar{\mathbf{i}}})$, in order to represent view-dependent color shifts. It is parametrized by the angles $\theta_{\bar{\mathbf{h}}}$ and $\theta_{\bar{\mathbf{i}}}$, where $\theta_{\bar{\mathbf{h}}}=\arccos(\bar{\mathbf{h}}_z)$ is the angle between half vector and normal and $\theta_{\bar{\mathbf{i}}}=\arccos(\bar{\mathbf{h}}\cdot\bar{\mathbf{i}})$ is the angle between half vector and incoming direction.
- A BTF representing the effects caused by the metallic flakes. It is parameterized by $\theta_{\bar{\mathbf{h}}}$, $\theta_{\bar{\mathbf{i}}}$ and $\mathbf{x}\in\mathbb{R}^2$, the position on the surface, i.e. it is a 4D table, denoted as $\Xi(\mathbf{x},\theta_{\bar{\mathbf{h}}},\theta_{\bar{\mathbf{i}}})$. According to Rump et al. [RMS*08], the angular lifetime of a metallic flake is around 6-7 degrees, which is why an angular sampling of 24-30 samples along each direction was chosen. Each combination of $\theta_{\bar{\mathbf{h}}}$ and $\theta_{\bar{\mathbf{i}}}$ results in a 2D texture. In the following, we call this function *flake BTF*.

The complete model below the clear coat is then:

$$f(\mathbf{x},\bar{\mathbf{i}},\bar{\mathbf{o}})=\chi(\theta_{\bar{\mathbf{h}}},\theta_{\bar{\mathbf{i}}})\left(\frac{a}{\pi}+\sum_{k=1}^{K}f^{CT}_{s_k,\alpha_k,F_{0,k}}\left(\bar{\mathbf{i}},\bar{\mathbf{o}}\right)\right)+\Xi(\mathbf{x},\theta_{\bar{\mathbf{h}}},\theta_{\bar{\mathbf{i}}}) \tag{5}$$

Choosing three lobes, i.e. $K=3$, has proven to deliver good results [GCG*05]. The complete model above the clear coat is then:

$$f(\mathbf{x},\mathbf{i},\mathbf{o})=F(\mathbf{i})\cdot\delta(\mathbf{h}-\mathbf{n})+[1-F(\mathbf{i})]f(\mathbf{x},\bar{\mathbf{i}},\bar{\mathbf{o}})[1-F(\mathbf{o})] \tag{6}$$

where $F(\mathbf{i})$ and $F(\mathbf{i})$ are the Fresnel term of the clear coat for the incoming and outgoing direction, respectively. δ is the Dirac delta function.

2.3. Unity BRDF Model

We use Unity to render the AxF car paints in real time. However, Unity's BRDF model is not directly compatible with the AxF car paint BRDF. Furthermore, we also want to make use of Unity's real-time global illumination system and deferred rendering to illuminate our materials.

We shortly introduce the BRDF model $\hat{f}(\mathbf{i},\mathbf{o})$ used in Unity:

$$\hat{f}(\mathbf{i},\mathbf{o})=\hat{f}^D_d+\hat{f}^S_{\alpha,s}(\mathbf{i},\mathbf{o}) \tag{7}$$

where $\hat{f}^D_d$ and $\hat{f}^S_{\alpha,s}(\mathbf{i},\mathbf{o})$ model the diffuse and specular part of the BRDF, d is the diffuse color, s represents the specular color and α denotes the roughness parameter. Internally $\hat{f}^S_{\alpha,s}(\mathbf{i},\mathbf{o})$ is a variant of the Cook-Torrance BRDF [CT82] as in the AxF car paint model with a GGX [WMLT07] distribution instead of a Beckmann distribution [BS63].

Unity's global illumination system pre-filters the environment map for different roughness values in $\hat{f}^S_{\alpha,s}$ under the assumption, that $\mathbf{n}=\mathbf{v}=\mathbf{r}$. During shading, the specular part of the BRDF is approximated by accessing the pre-filtered environment with the ideal reflection direction $\mathbf{r}$. For grazing angles, this is only an approximation of the BRDF's true behaviour. The diffuse part of the BRDF does not depend on the view direction. Here, it is enough to compute a spherical harmonics representation and sample it with the surface normal.

3. Our Method

An overview of our approach is illustrated in Figure 1. Car paints captured with state-of-the-art industry-grade setups such as X-Rite's TAC7 and stored in the AxF format are the input to our fitting pipeline that, given the constraints of the Unity engine, optimizes the parameters of Unity's BRDF model to best match the original AxF data. The major steps are represented by (1) approximating the flake BTF component of the AxF car paint using a single 2D texture, (2) computing an approximating 1-lobe Cook-Torrance model

based on GGX from the clear coated 3-lobe Cook-Torrance model using Beckmann distribution stored in the AxF file, and (3) computing a diffuse and specular color from the 2D color table. In the following, we provide details regarding the involved components.

The clear coat of the AxF car paint can easily be modeled using the Unity BRDF by setting $\alpha = 0$ and $s = F(\mathbf{i}) = F(\mathbf{o})$.

3.1. Single Flake Texture

The flake BTF depends both on the incoming and outgoing direction. However, simply sampling multiple incoming directions during rendering to sample the flake BTF is not possible due to performance and compatibility reasons.

Therefore, we replace the flake BTF by a single 2D texture. We assume $\mathbf{n} = \mathbf{i} = \mathbf{o}$, extract the corresponding slice from the flake BTF and use it as a standard texture. During rendering, the flake value is simply added to the specular color in the Unity BRDF model. This delivers good performance, while still conveying a relatively convincing look. However, an expert might notice the missing angular variability in the flakes.

3.2. BRDF Fitting

In this section, we describe how the Unity BRDF is fitted to the AxF car paint BRDF. In contrast to the single lobe used by Unity, the AxF car paints are represented using a superposition of three specular lobes. Furthermore, AxF car paints are mostly clear coated, in which case incoming and outgoing directions are refracted before the actual BRDF is evaluated. This drastically alters the width of the BRDF lobes. Refracting directions is not feasible when using Unity's global illumination system or deferred rendering. Our approach is based on fitting the parameters of the Unity BRDF model to the specular and diffuse part of the given AxF car paint BRDF.

Let $\hat{f}^S(\mathbf{i},\mathbf{o})$ denote the specular part of the Unity BRDF and $f^S(\mathbf{i},\mathbf{o})$ the specular part of the reference AxF car paint BRDF, both including the $\mathbf{i}_z$ term, i.e.

$$f^S(\mathbf{i},\mathbf{o}) = [1-F(\mathbf{i})]\left(\sum_{k=1}^{K} f^{CT}_{s_k,\alpha_k,F_{0,k}}(\bar{\mathbf{i}},\bar{\mathbf{o}})\right)[1-F(\mathbf{o})]\mathbf{i}_z \tag{8}$$

and

$$\hat{f}^S(\mathbf{i},\mathbf{o}) = [1-F(\mathbf{o})]\hat{f}^S_{\alpha,s}(\mathbf{i},\mathbf{o})[1-F(\mathbf{o})]\mathbf{i}_z \tag{9}$$

Considering the specular part of the BRDF, the maximal light transport is achieved in situations of ideal reflection, where $\mathbf{i}_z \approx \mathbf{o}_z$. Assuming $F(\mathbf{o}) \approx F(\mathbf{i})$, we augment the Unity BRDF with the Fresnel term of the incoming and outgoing direction. Since $F(\mathbf{o})$ only depends on the outgoing direction, which is always known during rendering, we can add this Fresnel terms into s in the fragment shader.

In order to find appropriate parameters, we minimize the squared L_2 error, weighted by a probability density function $p(\mathbf{i},\mathbf{o})$:

$$\min \iint_{\mathcal{H}} p(\mathbf{i},\mathbf{o}) \cdot \left(\hat{f}^S(\mathbf{i},\mathbf{o}) - f^S(\mathbf{i},\mathbf{o})\right)^2 d\mathbf{i}d\mathbf{o} \tag{10}$$

Where we write $p(\mathbf{i},\mathbf{o}) = p(\mathbf{i} \mid \mathbf{o})p(\mathbf{o})$ according to Bayes' rule and we choose $p(\mathbf{i} \mid \mathbf{o}) \sim f^S(\mathbf{i},\mathbf{o})$. The integral is approximated by a finite sum using Monte Carlo integration where the samples $\mathbf{i} \propto p$ are generated using rejection sampling.

In Unity's global illumination system, the environment map is filtered under the assumption that the surface normal coincides with the outgoing direction. For simplicity, we inherit this assumption here and fix the outgoing direction $\mathbf{o} = \mathbf{n}$. Therefore, $p(\mathbf{o}) = \delta(\mathbf{o} - \mathbf{n})$.

3.2.1. Roughness

Valid roughness values in Unity are in the interval $[0,1]$ and typically there is only a single roughness value that minimizes the error in that range. We observe that finding the best roughness value is likely to be a convex problem and can be solved using gradient descend. For each candidate roughness value, we compute the specular intensity and measure the error.

3.2.2. Specular Intensity

Given a roughness value, we still have to specify the specular intensity, i.e. the scaling factor of the specular lobe. For this purpose, we might use a linear solver to find the scaling factor that minimizes the squared error. However, this does not conserve the energy, which results in large brightness differences between the AxF model and the fitted Unity model. In our experiments, we observed that energy conservation is a more important property than minimizing the error function.

Energy conservation is achieved by setting s to the integral over the AxF car paint BRDF and dividing by the integral over the Unity BRDF. We reduce the amount of computation needed by exploiting the fact that the samples $\mathbf{i}$ are sampled from a distribution proportional to the AxF car paint BRDF, i.e.

$$\frac{\int_{\mathcal{H}} \hat{f}^S(\mathbf{i},\mathbf{o})\, d\mathbf{i}}{\int_{\mathcal{H}} f^S(\mathbf{i},\mathbf{o})\, d\mathbf{i}} = \frac{\int_{\mathcal{H}} \frac{\hat{f}^S(\mathbf{i},\mathbf{o})}{c \cdot f^S(\mathbf{i},\mathbf{o})} p(\mathbf{i},\mathbf{o})\, d\mathbf{i}}{\int_{\mathcal{H}} \frac{f^S(\mathbf{i},\mathbf{o})}{c \cdot f^S(\mathbf{i},\mathbf{o})} p(\mathbf{i},\mathbf{o})\, d\mathbf{i}} \tag{11}$$

$$\approx \frac{\sum_{\mathbf{i}} \frac{\hat{f}^S(\mathbf{i},\mathbf{o})}{c \cdot f^S(\mathbf{i},\mathbf{o})}}{\sum_{\mathbf{i}} \frac{f^S(\mathbf{i},\mathbf{o})}{c \cdot f^S(\mathbf{i},\mathbf{o})}} = \sum_{\mathbf{i}} \frac{\hat{f}^S(\mathbf{i},\mathbf{o})}{f^S(\mathbf{i},\mathbf{o})} \tag{12}$$

3.3. Color Table Sampling

In the AxF car paint model, material color is modeled independently of the BRDF lobes. Since the color table is parameterized by angles that also depend on the incoming light direction, we cannot directly use it with Unity's real-time global illumination system or deferred rendering. In this subsection, we describe how a diffuse and specular color is computed from the color table, which is then multiplied with the corresponding BRDF lobe during rendering. This way, we drop the dependency on the color table and incidentally save some memory.

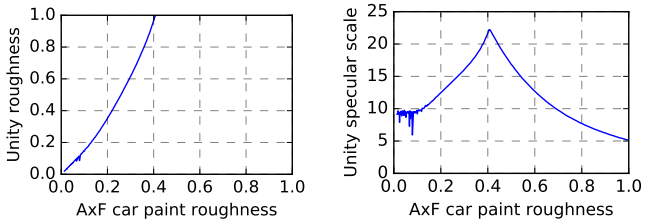
We filter the color table weighted by the diffuse and specular BRDF lobe, respectively. The weight of each $\theta_{\bar{\mathbf{h}}}$ and $\theta_{\bar{\mathbf{i}}}$ combination is induced by the probability distribution on incoming and outgoing directions according to the BRDF lobe. Instead of computing the weight for each color table entry, we sample pairs of incoming and outgoing directions, compute $\theta_{\bar{\mathbf{h}}}$ and $\theta_{\bar{\mathbf{i}}}$ and sample the color table accordingly.

We model the probability distribution on $\mathbf{i}$ and $\mathbf{o}$ as

$$p(\mathbf{i},\mathbf{o}) = p(\mathbf{i} \mid \mathbf{o})p(\mathbf{o}) \tag{13}$$

since the outgoing direction is independent of the shown material. The probability distribution on $\mathbf{o}$ is assumed to be $p(\mathbf{o}) \sim \mathbf{o}_z$ above the clear coat. $p(\mathbf{i} \mid \mathbf{o})$ then is given by the BRDF. For each color table sample, we draw $\mathbf{o} \propto p(\mathbf{o})$, and sample $\mathbf{i} \propto p(\mathbf{i} \mid \mathbf{o})$ using rejection sampling.

4. Results



(a) (b)

Figure 2: *Illustration of the fitted roughness in the Unity model, as well as the scaling factor for the specular BRDF lobe for a single lobe in the AxF car paint model with varying roughness,* $F_{0,1} = 1$ *and* $s_1 = 1$*. Fitting was done with* $10,000$ *samples.*

To validate our approach, we analyze the behaviour of our BRDF fitting for single lobes with varying AxF car paint roughness values. Figure 2 illustrates how the roughness of a specular lobe in the AxF car paint model maps to the Unity model, and how the specular intensity is changed. The roughness in the Unity model increases rapidly with increasing roughness in the AxF model, which is likely the case because the Unity model does not model refraction. A lobe with a certain width below the clear coat will result in a lobe with a wider roughness above the clear coat. For a roughness value ≥ 0.4 in the AxF car paint model, the Unity BRDF model cannot represent such a wide lobe, since the required roughness would be larger than 1. On the other hand, for very small roughness values our method is not stable. However, the real-world AxF car paints we encountered do not exhibit such extreme values.

Furthermore, we evaluate the visual quality of the fitted models. In this context, Figure 3 shows renderings of real-world car paints using the AxF car paint model, as well as fitted Unity BRDF models illuminated by Unity's real-time global illumination system. The overall appearance and brightness of the car paints is closely captured in the fitted Unity model. The appearance is not exactly preserved, but that is to be expected. The flakes in the fitted Unity model are more pronounced. This is likely the case, because we assume $\mathbf{n} = \mathbf{i} = \mathbf{o}$ for the flakes, which is a situation of ideal reflection. In the original flake BTF, we observed that flakes are strongest in such situations and weaker otherwise. However, during rendering the aforementioned assumption is not true.

5. Conclusion

In this paper, we demonstrated an approach for the automatic integration of measured car paints given in the industry-standard AxF format within the widely used Unity game engine. To achieve compatibility with the constraints imposed by Unity, we exploit the engine's underlying BRDF model and fit its parameters so that the resulting appearance approximates to the data of the AxF car paint model. With this approach we believe to foster an important future avenue of research to integrate measured materials in widely used game engines.

6. Acknowledgements

We would like to thank Volkswagen for providing measurements of the metallic paints.

References

[BS63] BECKMANN P., SPIZZICHINO A.: *The Scattering of Electromagnetic Waves from Rough Surfaces.* Pergamon Press, 1963. 2

[CT82] COOK R. L., TORRANCE K. E.: A reflectance model for computer graphics. *ACM Transactions on Graphics (TOG) 1*, 1 (1982), 7–24. 2

[GCG*05] GÜNTHER J., CHEN T., GOESELE M., WALD I., SEIDEL H.-P.: Efficient acquisition and realistic rendering of car paint. In *Vision, Modeling, and Visualization* (2005), vol. 5, pp. 487–494. 2

[GK17] GOLLA T., KLEIN R.: An efficient statistical data representation for real-time rendering of metallic effect car paints. In *Virtual Reality and Augmented Reality: 14th EuroVR International Conference, EuroVR 2017* (2017), Springer International Publishing, pp. 51–68. URL: https://doi.org/10.1007/978-3-319-72323-5_4, doi:10.1007/978-3-319-72323-5_4. 1

[ML15] MÜLLER G., LAMY F.: *AxF - Appearance exchange Format.* Tech. rep., X-Rite, Inc., 4300 44th St. SE, Grand Rapids, MI 49505, 2015. Version 1.0. 1, 2

[RMS*08] RUMP M., MÜLLER G., SARLETTE R., KOCH D., KLEIN R.: Photo-realistic rendering of metallic car paint from image-based measurements. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 527–536. 2

[Sch94] SCHLICK C.: An inexpensive brdf model for physically-based rendering. In *Computer graphics forum* (1994), vol. 13, Wiley Online Library, pp. 233–246. 2

[WMLT07] WALTER B., MARSCHNER S. R., LI H., TORRANCE K. E.: Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics conference on Rendering Techniques* (2007), Eurographics Association, pp. 195–206. 2

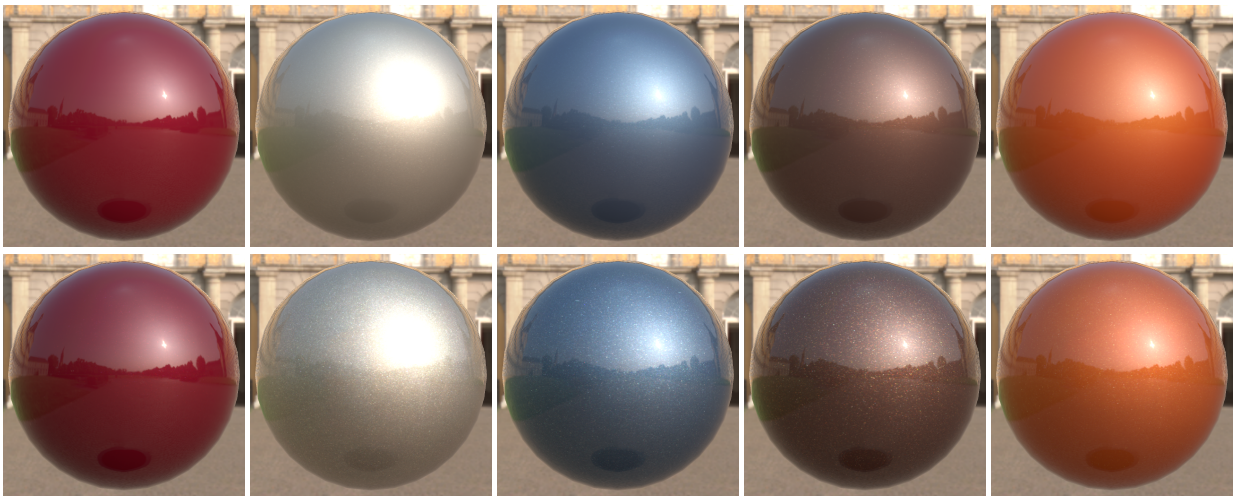


Figure 3: *Rendering of various car paints. The top row contains reference renderings using the AxF car paint model, and the bottom row contains renderings of the fitted Unity model.*