UCSD CSE 272 Assignment 1:

Disney Principled BSDF

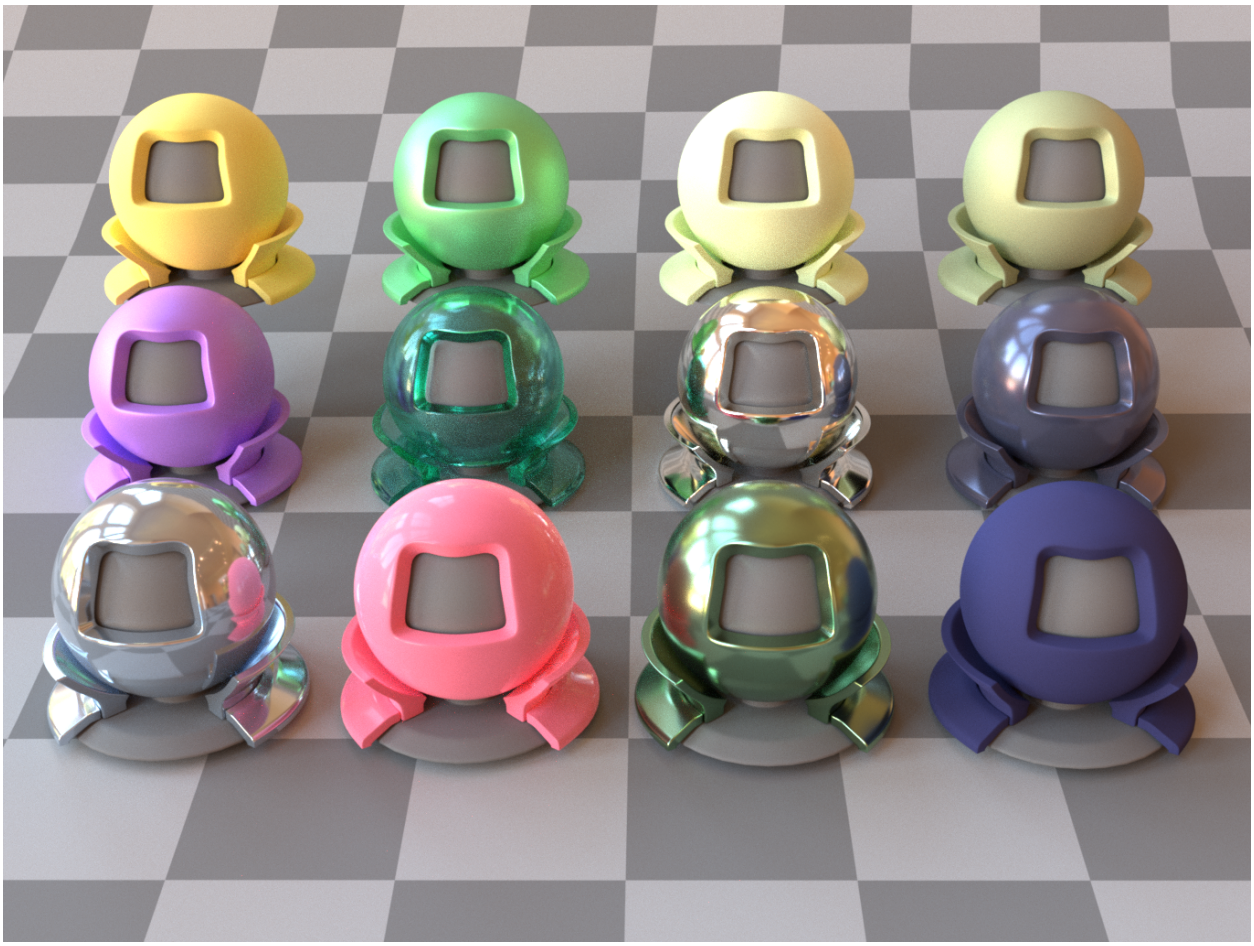

Figure 1: Disney principled BSDF [1, 2] is a *Uber shader* that can express a very wide range of materials.

In this homework, we will implement a Bidirectional Scattering Distribution Function, called the *Disney principled BSDF* (Figure 1), in lajolla. Disney principled BSDF is an attempt to have a one-size-fits-all solution to cover most common materials using a single BSDF. It is *principled* since it is (mostly) based on physical principles and observations from measured data [8]. However, physical correctness is not the utmost priority of the BSDF: it is a useful guideline for parameterizing the space of plausible materials for artistic expression. Ultimately, Disney BSDF is for making visual effects: as long as it makes graphics artists express what they want with the least effort, it achieves its goal. Disney BSDF has been extremely influential since its inception in 2012. Nowadays, many commercial and non-commercial rendering engines feature a similar material: Blender's principled BSDF, Autodesk's Standard Surface, Unreal Engine 4's physically-based materials, Substance's physically-based shaders, and Appleseed standard surface, are all heavily inspired, if not directly borrowed from the Disney BSDF.

We will implement the Disney BSDF with slight simplifications: first, we won't implement the full volumetric absorption/scattering model (we will implement something similar in the next homework!). Second, we remove a sheen transmissive lobe to make sampling easier. Finally, we do not implement the *thin BSDF* model. When implementing the BSDF, feel free to reference code from the internet. However, be aware that due to the ambiguity in the Disney course note, all implementations I found are slightly different from

each other, and some are flat out wrong.[1] I found the following links to be useful: the official implementation of the BRDF (lacks the transmission component and no sampling procedures), pbrt's implementation, Joe Schutte's walkthrough, a GLSL implementation, and Blender's open shading language implementation. The model we will implement is the closest to Blender's version. You should also read Burley's notes and presentations.[2]

A Disney BSDF is made of five components: a **diffuse** lobe that captures the base diffusive color of the surface, a **metallic** lobe that features major specular highlights, a **clearcoat** lobe that models the heavy tails of the specularity, a **sheen** lobe that addresses retroreflection, and a **glass** lobe that handles transmission. We will first implement each individual component; then we will combine all of them into a single material.

Submission and grading. Please upload a zip file to Canvas including your code and a folder `images` containing your rendering of the scenes below. For the questions, answer them on Gradescope. For the questions, as long as you say something plausible, you will get full scores. Some questions do not have a single correct answer. Do think hard about the questions though. We want you to get the high-level concepts correct, rather than trying to match every single detail.

Notation and convention. In the following, ω_{in} is the *incoming* direction of the BSDF (usually represent the view direction), and ω_{out} is the *outgoing* direction of the BSDF (usually represent the light direction), both pointing *outwards* from the surface. n is the shading normal, n_g is the geometric normal, h is the half-vector $h = \frac{\omega_{\text{in}}+\omega_{\text{out}}}{\|\omega_{\text{in}}+\omega_{\text{out}}\|}$. All of our BSDF include the cosine term $|n \cdot \omega_{\text{out}}|$. η is the ratio of index of refraction of the medium below divided by the medium above the surface $\frac{\text{IOR}_{\text{internal}}}{\text{IOR}_{\text{external}}}$. All parameters of Disney BSDF are normalized within $[0, 1]$, except for the index of refraction whose acceptable range is $(1, 2]$.

Variant-based material systems. As mentioned in homework 0, lajolla applies variant-based polymorphism instead of object-oriented polymorphism. The material `structs` in lajolla looks like the following:

```
struct DisneyDiffuse {
    Texture<Spectrum> base_color;
    Texture<Real> roughness;
    Texture<Real> subsurface;
};
```

They are aggregated to a `Material` type using variant.

```
using Material = std::variant<Lambertian,
                              RoughPlastic,
                              RoughDielectric,
                              DisneyDiffuse,
                              DisneyMetal,
                              DisneyGlass,
                              DisneyClearcoat,
                              DisneySheen,
                              DisneyBSDF>;
```

Each material needs to implement the following operators:

```
struct eval_op {
    Spectrum operator()(const Lambertian &bsdf) const;
    Spectrum operator()(const RoughPlastic &bsdf) const;
    Spectrum operator()(const RoughDielectric &bsdf) const;
    Spectrum operator()(const DisneyDiffuse &bsdf) const;
    // ...
```

[1] Even the implementation in pbrt appears to be wrong. See here.

[2] https://blog.selfshadow.com/publications/s2012-shading-course/ and https://blog.selfshadow.com/publications/s2015-shading-course/

```
    const Vector3 &dir_in;
    const Vector3 &dir_out;
    const PathVertex &vertex;
    const TexturePool &texture_pool;
    const TransportDirection &dir;
};

struct pdf_sample_bsdf_op {
    Real operator()(const Lambertian &bsdf) const;
    Real operator()(const RoughPlastic &bsdf) const;
    Real operator()(const RoughDielectric &bsdf) const;
    Real operator()(const DisneyDiffuse &bsdf) const;
    // ...

    const Vector3 &dir_in;
    const Vector3 &dir_out;
    const PathVertex &vertex;
    const TexturePool &texture_pool;
    const TransportDirection &dir;
};

struct sample_bsdf_op {
    std::optional<BSDFSampleRecord> operator()(const Lambertian &bsdf) const;
    std::optional<BSDFSampleRecord> operator()(const RoughPlastic &bsdf) const;
    std::optional<BSDFSampleRecord> operator()(const RoughDielectric &bsdf) const;
    std::optional<BSDFSampleRecord> operator()(const DisneyDiffuse &bsdf) const;
    // ...

    const Vector3 &dir_in;
    const PathVertex &vertex;
    const TexturePool &texture_pool;
    const Vector2 &rnd_param_uv;
    const Real &rnd_param_w;
    const TransportDirection &dir;
};
```

---

1 Diffuse

By looking at the MERL measured BRDF [8], Burley found that at grazing retroreflection (when the half vector is roughly orthogonal to the normal), smooth materials (materials that are more specular) tend to have their reflectance dropped, and rough materials tend to have a peak at the grazing angle. The dropped reflectance can be predicted by Fresnel reflection: at grazing angles, (dielectric) Fresnel equation predicts low transmittance, and fewer lights are scattered inside the surfaces and thus less diffusion. Thus they design the following diffuse BRDF based on a modified version of the Schlick Fresnel approximation [10]:

$$f_{\text{baseDiffuse}} = \frac{\text{baseColor}}{\pi} F_D(\omega_{\text{in}}) F_D(\omega_{\text{out}}) |n \cdot \omega_{\text{out}}|, \tag{1}$$

where

$$\begin{aligned} F_D(\omega) &= \left(1 + (F_{D90} - 1)(1 - |n \cdot \omega|)^5\right) \\ F_{D90} &= \frac{1}{2} + 2 \cdot \text{roughness} \cdot |h \cdot \omega_{\text{out}}|^2. \end{aligned} \tag{2}$$

When roughness $= 0$ (smooth materials), at grazing view angle $|n \cdot \omega_{\text{in}}| \approx 0$, and at grazing lighting angle $|n \cdot \omega_{\text{out}}| \approx 0$, thus the BSDF attenuates the diffuse response by a factor of $\frac{1}{2}$ for each F_D term (modelling the Fresnel reflection). At high roughness (roughness ≈ 1), at retroreflection, $\omega_{\text{out}} \approx \omega_{\text{in}}$, thus $h \cdot \omega_{\text{out}} \approx 1$,

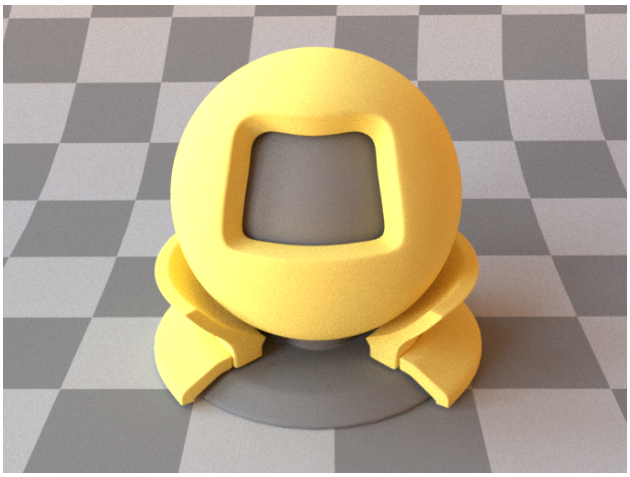

Figure 2: Diffuse component of the Disney BSDF.

and the BSDF reproduces the peak retroreflection observed in the MERL data by multiplying 2.5 for each F_D term.

In addition to the base diffuse model, the diffuse component of Disney BSDF also blends it with a subsurface scattering lobe for surfaces with strong multiple scattering inside like skin, milk, or marble. In the 2015 version of the Disney BSDF, they simulate *real* subsurface scattering that requires volumetric path tracing to simulate the scattering, but this is too much work for the first homework. Instead, we follow the 2012 version and use a BRDF approximation of the subsurface scattering by modifying the Lommel-Seeliger law:

$$f_{\text{subsurface}} = \frac{1.25\text{baseColor}}{\pi}\left(F_{SS}(\omega_{\text{in}})F_{SS}(\omega_{\text{out}})\left(\frac{1}{|n\cdot\omega_{\text{in}}|+|n\cdot\omega_{\text{out}}|}-0.5\right)+0.5\right)|n\cdot\omega_{\text{out}}|, \tag{3}$$

where

$$\begin{aligned} F_{SS}(\omega) &= \left(1+(F_{SS90}-1)(1-|n\cdot\omega|)^5\right) \\ F_{SS90} &= \text{roughness}\cdot|h\cdot\omega_{\text{out}}|^2. \end{aligned} \tag{4}$$

See here for a nice sketch of derivation of the Lommel-Seeliger law (brought to graphics by Hanrahan and Kruger [5]). The $\frac{1}{|n\cdot\omega_{\text{in}}|+|n\cdot\omega_{\text{out}}|}$ term models the volumetric absorption of the scattering media below the surface.

The final diffuse BRDF is:

$$f_{\text{diffuse}} = (1-\text{subsurface})\cdot f_{\text{baseDiffuse}} + \text{subsurface}\cdot f_{\text{subsurface}}, \tag{5}$$

where subsurface is a parameter.

Note that the physical model here is a dielectric coating on top of a diffusive scattering media (similar to the `roughplastic` material in lajolla/Mitsuba), but f_{diffuse} does not model the specular reflection of the dielectric coating. We will include the specular reflection in the final assembly.

Questions (7%). Answer these questions on Gradescope:

1. Compare the two BRDFs $f_{\text{baseDiffuse}}$ and $f_{\text{subsurface}}$ with a Lambertian BRDF (try to render images with all three BRDFs): what differences do you see? Why?
2. Play with the roughness parameter: how does it affect the appearance?
3. Compare the base diffuse BRDF ($f_{\text{baseDiffuse}}$) with the subsurface BRDF ($f_{\text{subsurface}}$) by playing with the subsurface parameter. What differences do you see? Why? In what lighting condition does the base diffuse BRDF differ the most from the subsurface BRDF? (Play with the light position in `simple_sphere.xml` for your experimentation)

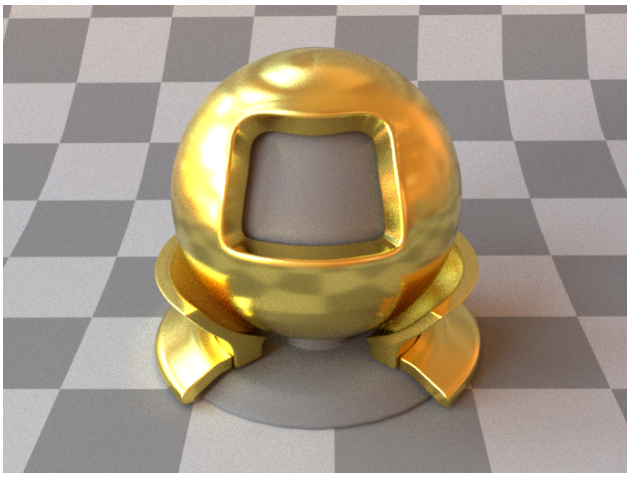

Figure 3: Metal component of the Disney BSDF.

4. (Optional, bonus 3%) Another popular option for modeling diffuse surfaces is the Oren-Nayar BRDF [9], which is used in the Autodesk's Standard Surface BSDF. What is the difference between the Oren-Nayar BRDF and the Disney diffuse BRDF? What are the pros and cons? What is your preference?

Think about these questions as you implement the BRDF.

Task (8%). You will implement the `DisneyDiffuse` BRDF (Equation 5)

```
// in material.h
struct DisneyDiffuse {
    Texture<Spectrum> base_color;
    Texture<Real> roughness;
    Texture<Real> subsurface;
};
```

You need to implement the following three functions in `materials/disney_diffuse.inl`:

```
Spectrum eval_op::operator()(const DisneyDiffuse &bsdf) const;
Real pdf_sample_bsdf_op::operator()(const DisneyDiffuse &bsdf) const;
std::optional<BSDFSampleRecord> sample_bsdf_op::operator()(const DisneyDiffuse &bsdf) const;
```

For sampling, we will simply use a cosine-weighted hemisphere sampling. Look at `materials/lambertian.inl` to see how it is done. Feel free to copy-paste the code and modify anything. Also observe how the Lambertian BRDF implementation handles the discrepancy between geometric normals and shading normals.

Try out the scene `scenes/disney_bsdf_test/simple_sphere.xml` (you'll need to modify the scene file to use the `DisneyDiffuse` material) and `scenes/disney_bsdf_test/disney_diffuse.xml` to see how the material look like. Play with the parameters.

Store your rendering of the `disney_diffuse` scene in

```
images/disney_diffuse.exr
```

2 Metal

For specular reflection, Burley uses a standard Cook-Torrance microfacet BRDF [3]:

$$f_{\text{metal}} = \frac{F_m D_m G_m}{4|n \cdot \omega_{\text{in}}|}, \quad [3] \tag{6}$$

where F_m is the Fresnel reflection, D_m is the probability density of the distribution of a microfacet normal, and G_m is the masking-shadowing term [6] that models the occlusion between microfacets.

For the Fresnel term F_m, Burley uses the Schlick approximation:

$$F_m = \text{baseColor} + (1 - \text{baseColor})(1 - |h \cdot \omega_{\text{out}}|)^5. \tag{7}$$

Note that the Fresnel term depends on the micronormal $h = \frac{\omega_{\text{in}} + \omega_{\text{out}}}{|\omega_{\text{in}} + \omega_{\text{out}}|}$ instead of the macro shading normal n. The reason why they use an approximation instead of the true Fresnel equation is not just for the performance. For metallic surfaces, or *conductors*, Fresnel equation requires us to have the complex index of refraction of the conductor for each wavelength. The complex index of refraction is not only unintuitive, but nor is it physically accurate when we only consider the RGB spectrum.[4]

For the normal distribution function D_m, Burley uses the anisotropic Trowbridge-Reitz distribution [11], popularized by Walter et al. [12] in graphics and it is known as *GGX* (Ground Glass X):

$$D_m = \frac{1}{\pi \alpha_x \alpha_y \left(\frac{{h_x^l}^2}{\alpha_x^2} + \frac{{h_y^l}^2}{\alpha_y^2} + {h_z^l}^2 \right)^2}, \tag{8}$$

where h^l is the half-vector projected to the local shading frame. Physically, this models the distribution of the normals of an ellipsoid. Trowbridge-Reitz distribution was found to be fitting the MERL measured data excellently thanks to its heavy tails compared to a Gaussian or a cosine (some materials have even longer tails, and those will be modeled by the clearcoat component later).

α_x, α_y are parameters for modeling the smoothness of the material. If we directly use these parameters, we need very small α to represent highly specular materials. Burley found that the following mapping is more intuitive:

$$\begin{aligned} \text{aspect} &= \sqrt{1 - 0.9\text{anisotropic}} \\ \alpha_x &= \max(\alpha_{\min}, \text{roughness}^2 / \text{aspect}) \\ \alpha_y &= \max(\alpha_{\min}, \text{roughness}^2 \cdot \text{aspect}) \end{aligned}, \tag{9}$$

where anisotropic and roughness are the parameters, and $\alpha_{\min} = 0.0001$.

Given a normal distribution function and a microfacet configuration, it is possible to derive the average occlusion factor G_m given a viewing angle [6]. Burley uses the Smith model, which enables a closed-form solution under the assumption that all the microfacets are independently oriented:

$$\begin{aligned} G_m &= G(\omega_{\text{in}}) G(\omega_{\text{out}}) \\ G(\omega) &= \frac{1}{1 + \Lambda(\omega)} \\ \Lambda(\omega) &= \frac{\sqrt{1 + \frac{(\omega_l.x \cdot \alpha_x)^2 + (\omega_l.y \cdot \alpha_y)^2}{\omega_l.z^2}} - 1}{2} \end{aligned}. \tag{10}$$

Combining all of these, and you will get a nice metallic BRDF.

Questions (7%). Answer these question(s) on Gradescope:

1. Compare `DisneyMetal` with the `roughplastic` material (try to render images with both BRDFs). What differences do you see? Why?

[3] The $|n \cdot \omega_{\text{out}}|$ term in the denominator cancels out with the cosine in rendering equation.

[4] See "Fresnel Equations Considered Harmful" by Naty Hoffman. Here is his presentation video.

2. Change the roughness parameters. Apart from how specular the surface it, do you observe any other differences?

3. A popular alternative over the Trowbridge-Reitz normal distribution function is the Beckmann distribution (a Gaussian distribution on the *slopes* $\frac{h_z^l}{h_x^l}$ and $\frac{h_z^l}{h_y^l}$ of the normals). What are the differences between Trowbridge-Reitz and Beckmann? Why did Disney folks choose to use Trowbridge-Reitz instead of Beckmann? (You might want to read the awesome article Slope Space in BRDF Theory from Nathan Reed.)

4. (Optional, bonus 3%) What are the pros and cons of the Schlick approximation compared to the actual Fresnel equation? What is your preference? (You may want to read/watch the Naty Hoffman presentation in the footnote above.)

Think about these questions as you implement the BRDF.

Task (8%). You will implement the `DisneyMetal` BRDF (Equation 6)

```
// in material.h
struct DisneyMetal {
    Texture<Spectrum> base_color;
    Texture<Real> roughness;
    Texture<Real> anisotropic;
};
```

You need to implement the following three functions in `materials/disney_metal.inl`:

```
Spectrum eval_op::operator()(const DisneyMetal &bsdf) const;
Real pdf_sample_bsdf_op::operator()(const DisneyMetal &bsdf) const;
std::optional<BSDFSampleRecord> sample_bsdf_op::operator()(const DisneyMetal &bsdf) const;
```

For sampling, we will use the visible normal sampling developed by Heitz [7], which importance samples $\frac{D_m G(\omega_{\text{in}})}{4|n \cdot \omega_{\text{in}}|}$. See Heitz's presentation slides for a really nice illustration of the method. This sampling is also used in the `roughplastic` material in lajolla, so you might want to look at it as well.

Try out the scene `scenes/disney_bsdf_test/simple_sphere.xml` and the scene `scenes/disney_bsdf_test/disney_metal.xml` to see how the material look like. Play with the parameters.

Store your rendering of the `disney_metal` scene in

```
images/disney_metal.exr
```

3 Clearcoat

Burley found that the Trowbridge-Reitz distribution used by the metallic component above, while already having a wide tail comparing to most other normal distribution functions, is still not wide enough. Therefore they propose to have another achromatic specular component with a modified normal distribution function:

$$f_{\text{clearcoat}} = \frac{F_c D_c G_c}{4|n \cdot \omega_{\text{in}}|}, \tag{11}$$

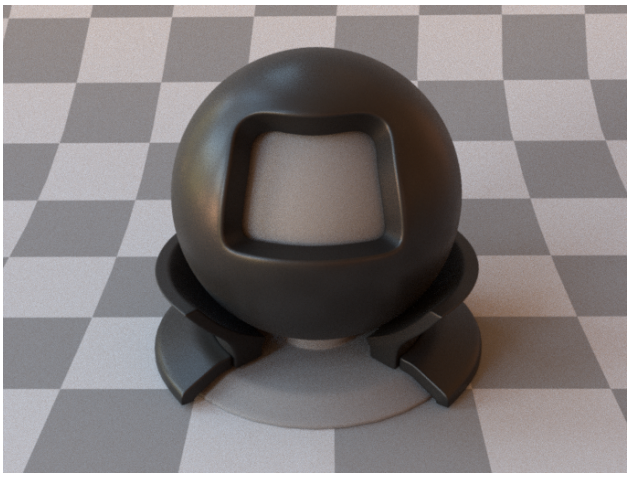

Figure 4: Clearcoat component of the Disney BSDF.

where

$$\begin{aligned} F_c &= R_0(\eta = 1.5) + (1 - R_0(\eta = 1.5))\left(1 - |h \cdot \omega_{\text{out}}|\right)^5 \\ D_c &= \frac{\alpha_g^2 - 1}{\pi \log(\alpha_g^2)\left(1 + (\alpha_g^2 - 1)\left(h_z^l\right)^2\right)} \\ G_c &= G_c(\omega_{\text{in}})G_c(\omega_{\text{out}}) \\ G_c(\omega) &= \frac{1}{1 + \Lambda_c(\omega)} \\ \Lambda_c(\omega) &= \frac{\sqrt{1 + \frac{(\omega_l.x \cdot 0.25)^2 + (\omega_l.y \cdot 0.25)^2}{\omega_l.z^2}} - 1}{2} \end{aligned} \quad . \tag{12}$$

The Schlick Fresnel F_c has a hard-coded index of refraction $\eta = 1.5$. The normal distribution function D_c uses an isotropic roughness $\alpha = \alpha_g$. The masking-shadowing term G_c uses a fixed roughness 0.25. Note that this is an ad-hoc fit, and there is no clear geometric meaning of this microfacet BRDF.

The Schlick approximation maps the index of refraction η to R_0 with the following equation:

$$R_0(\eta) = \frac{(\eta - 1)^2}{(\eta + 1)^2} \tag{13}$$

The α_g parameter is mapped to a parameter clearcoatGloss with the following equation:

$$\alpha_g = (1 - \text{clearcoatGloss}) \cdot 0.1 + \text{clearcoatGloss} \cdot 0.001. \tag{14}$$

The higher the clearcoatGloss, the lower the α_g.

Questions (7%). Answer these question(s) on Gradescope:

1. Compare `DisneyClearcoat` with `DisneyMetal` using similar roughness. What differences do you see? Where do the differences come from?

2. For coating, Autodesk Standard Surface uses a standard Trowbridge-Reitz microfacet distribution, instead of the modified normal distribution function proposed by Burley. What are the pros and cons? What is your preference?

3. Why do Burley limit the clearcoat BRDF to be isotropic?

Think about these questions as you implement the BRDF.

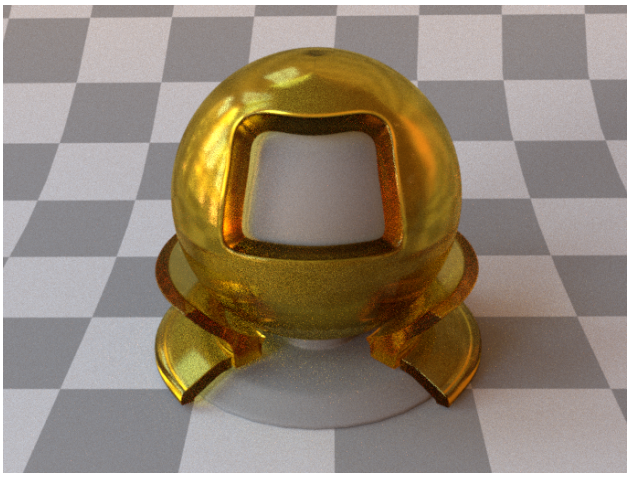

Figure 5: Glass component of the Disney BSDF.

Task (8%). You will implement the `DisneyClearcoat` BRDF (Equation 11)

```
// in material.h
struct DisneyClearcoat {
    Texture<Real> clearcoat_gloss;
};
```

You need to implement the following three functions in `materials/disney_clearcoat.inl`:

```
Spectrum eval_op::operator()(const DisneyClearcoat &bsdf) const;
Real pdf_sample_bsdf_op::operator()(const DisneyClearcoat &bsdf) const;
std::optional<BSDFSampleRecord> sample_bsdf_op::operator()(const DisneyClearcoat &bsdf) const;
```

For sampling, there is no known visible normal sampling for this BRDF (since it does not correspond to a meaningful physical configuration). We will importance sample $\frac{D_c|n\cdot h|}{4|h\cdot\omega_{\text{out}}|}$ by choosing a micro normal proportional to D_c then reflect the incoming direction. To sample a normal h^l in the local shading frame using a 2D random number (u_0, u_1):

$$\begin{aligned}
\cos(h_{\text{elevation}}) &= \sqrt{\frac{1-(\alpha^2)^{1-u_0}}{1-\alpha^2}} \\
h_{\text{azimuth}} &= 2\pi u_1 \\
h^l_x &= \sin(h_{\text{elevation}})\cos(h_{\text{azimuth}}) \\
h^l_y &= \sin(h_{\text{elevation}})\sin(h_{\text{azimuth}}) \\
h^l_z &= \cos(h_{\text{elevation}})
\end{aligned} \tag{15}$$

Try out the scene `scenes/disney_bsdf_test/simple_sphere.xml` and the scene `scenes/disney_bsdf_test/disney_clearcoat.xml` to see how the material look like. Play with the parameters.

Store your rendering of the `disney_clearcoat` scene in

```
images/disney_clearcoat.exr
```

4 Glass

The 2012 version of the Disney BRDF did not support glasses. In the 2015 version, Burley added a dielectric lobe that is both transmissive and reflective using a standard microfacet-based refraction model [12]:

$$f_{\text{glass}} = \begin{cases} \frac{\text{baseColor} F_g D_g G_g}{4|n\cdot\omega_{\text{in}}|} & \text{if } (n_g\cdot\omega_{\text{in}})\,(n_g\cdot\omega_{\text{out}}) > 0 \\ \frac{\sqrt{\text{baseColor}}(1-F_g) D_g G_g |h\cdot\omega_{\text{out}} h\cdot\omega_{\text{in}}|}{|n\cdot\omega_{\text{in}}|(h\cdot\omega_{\text{in}}+\eta h\cdot\omega_{\text{out}})^2} & \text{otherwise} \end{cases} \tag{16}$$

Note the square root of the albedo in the refractive case.

For the Fresnel F_g, Burley found that the Schlick approximation was very inaccurate for $\eta \approx 1$ (up to $40\times$ brighter than actual Fresnel), and decided to use the actual Fresnel equation for the dielectric materials. Unlike conductors, the Fresnel reflection for dielectric materials does not require complex indices of refraction and is much more intuitive to control. For completeness, the Fresnel equation is:

$$\begin{aligned} F_g &= \frac{1}{2}\left(R_s^2 + R_p^2\right) \\ R_s &= \frac{h\cdot\omega_{\text{in}} - \eta h\cdot\omega_{\text{out}}}{h\cdot\omega_{\text{in}} + \eta h\cdot\omega_{\text{out}}} \\ R_p &= \frac{\eta h\cdot\omega_{\text{in}} - h\cdot\omega_{\text{out}}}{\eta h\cdot\omega_{\text{in}} + h\cdot\omega_{\text{out}}} \end{aligned} \tag{17}$$

again, note that we use the half-vector h instead of the normal n, due to the microfacet assumption.

Sometimes we need to compute the Fresnel term solely using either the incoming direction or the outgoing direction. This can be done by using the Snell-Descartes law to convert $n\cdot\omega$ to the other one.

The normal distribution function D_g and the masking-shadowing term G_g are the same as the metal case.

Questions (7%). Answer these question(s) on Gradescope:

1. Why do we take a square root of `baseColor` in the refractive case?
2. Play with the index of refraction parameter η (the physically plausible range is $[1, 2]$). How does it affect appearance?
3. If a refractive object is not a closed object and we assign it to be a glass BSDF, will everything still work properly? Why? Propose a solution if it will not work properly (hint: you may want to read the *thin-surface BSDF* in Burley's note [2].).
4. (Optional, 3%) Replace the dielectric Fresnel equation with a Schlick approximation (see Burley's course notes [2] on the fix to the Schlick approximation to make it work for $\eta < 1$). Do you observe any differences when $\eta = 1.5$? What about $\eta = 1.01$?

Think about these questions as you implement the BSDF.

Task (8%). You will implement the `DisneyGlass` BRDF (Equation 16)

```
// in material.h
struct DisneyGlass {
    Texture<Spectrum> base_color;
    Texture<Real> roughness;
    Texture<Real> anisotropic;

    Real eta; // internal IOR / externalIOR
};
```

You need to implement the following three functions in `materials/disney_glass.inl`:

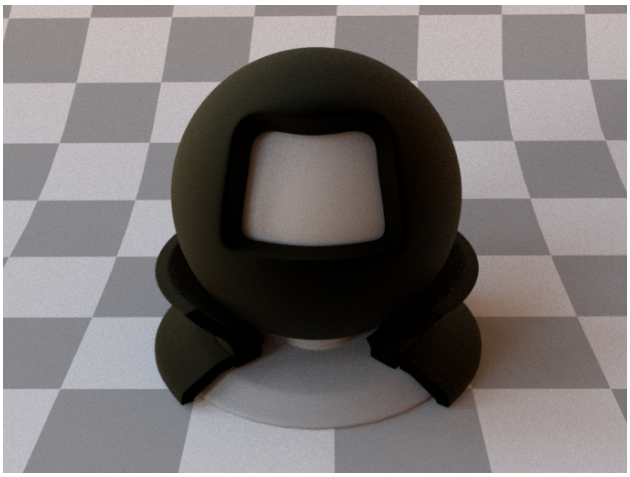

Figure 6: Sheen component of the Disney BSDF.

```
Spectrum eval_op::operator()(const DisneyGlass &bsdf) const;
Real pdf_sample_bsdf_op::operator()(const DisneyGlass &bsdf) const;
std::optional<BSDFSampleRecord> sample_bsdf_op::operator()(const DisneyGlass &bsdf) const;
```

Lajolla already has an implementation of a dielectric glass in `roughdielectric`. The only two differences are 1) the reflectance/transmittance color are handled differently and 2) the BSDF in `roughdielectric` is isotropic. I suggest you start by copy-pasting or referencing the code from `roughdielectric` and modify it.

Try out the scene `scenes/disney_bsdf_test/disney_glass.xml` to see how the material look like. Play with the parameters.

Store your rendering of the `disney_glass` scene in

```
images/disney_glass.exr
```

5 Sheen

Materials such as clothes often exhibit strong responses at grazing angles. To compensate for this, Burley adds another component called the *sheen* BRDF. Anything related to grazing angles can be done by hacking the Fresnel response (just like f_{diffuse}). The sheen BRDF is again a modified Schlick Fresnel:

$$\begin{aligned} f_{\text{sheen}} &= C_{\text{sheen}}(1 - |h \cdot \omega_{\text{out}}|)^5 |n \cdot \omega_{\text{out}}| \\ C_{\text{sheen}} &= (1 - \text{sheenTint}) + \text{sheenTint} \cdot C_{\text{tint}} \\ C_{\text{tint}} &= \text{baseColor/luminance(baseColor) if luminance(baseColor)} > 0 \text{ else } 1 \end{aligned} . \tag{18}$$

Here the color C_{sheen} is a blending between the hue and saturation of the base color and white color based on the parameter sheenTint. Physical-wise, specular reflection caused by a dielectric material is usually achromatic when the index of refraction is the same across different wavelengths (and it often is). However, sheen is a BSDF for modeling strong retroreflection from clothes that are caused by the mesostructure of the cloth – a much more complicated phenomonon. It is thus the best to give artists control here for how achromatic they want the retroreflection to be.

In the 2015 version of the Disney BSDF, the sheen component is applied for both the reflection and transmission. We only apply the sheen for the reflection here, since it is easier to sample and code.

Questions (7%). Answer these question(s) on Gradescope:

1. Render the `simple_sphere` scene with the sheen BRDF. What do you see? Why? What happens if you change the position of the light source?
2. Play with the parameter `sheenTint`, how does it affect the appearance? Why?
3. In Autodesk Standard Surface, the sheen is modeled by a microfacet BRDF [4]. What are the pros and cons between the Autodesk approach and the Disney approach? What is your preference?

Think about these questions as you implement the BSDF.

Task (8%). You will implement the `DisneySheen` BRDF (Equation 18)

```
// in material.h
struct DisneySheen {
    Texture<Spectrum> base_color;
    Texture<Real> sheen_tint;
};
```

You need to implement the following three functions in `materials/disney_sheen.inl`:

```
Spectrum eval_op::operator()(const DisneySheen &bsdf) const;
Real pdf_sample_bsdf_op::operator()(const DisneySheen &bsdf) const;
std::optional<BSDFSampleRecord> sample_bsdf_op::operator()(const DisneySheen &bsdf) const;
```

For sampling we will simply apply a cosine hemisphere sampling.

Try out `scenes/disney_bsdf_test/simple_sphere.xml` and `scenes/disney_bsdf_test/disney_sheen.xml` to see how the material look like. Play with the parameters.

Store your rendering of the `disney_sheen` scene in

```
images/disney_sheen.exr
```

6 Putting everything together

Finally, we combine the five components we have and weigh them to get our final BSDF:

$$\begin{aligned} f_{\text{disney}} =& (1 - \text{specularTransmission}) \cdot (1 - \text{metallic}) \cdot f_{\text{diffuse}} + \\ & (1 - \text{metallic}) \cdot \text{sheen} \cdot f_{\text{sheen}} + \\ & (1 - \text{specularTransmission} \cdot (1 - \text{metallic})) \cdot \hat{f}_{\text{metal}} + \\ & 0.25 \cdot \text{clearcoat} \cdot f_{\text{clearcoat}} + \\ & (1 - \text{metallic}) \cdot \text{specularTransmission} \cdot f_{\text{glass}} \end{aligned} \tag{19}$$

We need to modify the metal BRDF $\hat{f}_{\text{metal}}$ a bit: recall that our diffuse material is missing a dielectric specular reflection. We include that in our metal BRDF. To do this, we modify the Fresnel term F_m to include an achromatic specular component, with a control parameter specularTint to potentially make it closer to the base color:

$$\begin{aligned} \hat{F}_m &= C_0 + (1 - C_0)\left(1 - (h \cdot \omega_{\text{out}})\right)^5 \\ C_0 &= \text{specular} \cdot R_0(\eta)(1 - \text{metallic})K_s + \text{metallic} \cdot \text{baseColor} \\ K_s &= (1 - \text{specularTint}) + \text{specularTint} \cdot C_{\text{tint}} \end{aligned} \tag{20}$$

One complication when combining the glass BSDF and other BRDFs is that now we need to define the behavior of the BRDF when the ray is *inside* the object. The Disney BSDF technical notes do not specify

this. Experiments show that removing all lobes except for the glass lobe when ray is inside the object (note that the glass still reflect inside the object) gives more visually pleasing results. Therefore we define:

$$\begin{aligned} f_{\text{diffuse}} &= 0 \text{ if } \omega_{\text{in}} \cdot n_g \leq 0 \\ f_{\text{metal}} &= 0 \text{ if } \omega_{\text{in}} \cdot n_g \leq 0 \\ f_{\text{clearcoat}} &= 0 \text{ if } \omega_{\text{in}} \cdot n_g \leq 0 \\ f_{\text{sheen}} &= 0 \text{ if } \omega_{\text{in}} \cdot n_g \leq 0 \end{aligned} \tag{21}$$

Questions (7%). Answer these question(s) on Gradescope:

1. What are the differences between the `specular` and `metallic` parameters? How do they affect the appearance?
2. What are the differences between the `roughness` and `clearcoat_gloss` parameters? How do they affect the appearance?
3. Play with the `specularTint` parameter. How does it affect the appearance?
4. The `roughness` parameter affects many components of the BSDFs at once (e.g., both the diffuse and metal BRDF use the roughenss parameter). How do you feel about this? If you are an artist using the Disney BSDF, would you want to have a separate roughness parameter for each component?

Think about these questions as you implement the BSDF.

Task (8%). You will implement `DisneyBSDF` (Equation 19)

```
// in material.h
struct DisneyBSDF {
    Texture<Spectrum> base_color;
    Texture<Real> specular_transmission;
    Texture<Real> metallic;
    Texture<Real> subsurface;
    Texture<Real> specular;
    Texture<Real> roughness;
    Texture<Real> specular_tint;
    Texture<Real> anisotropic;
    Texture<Real> sheen;
    Texture<Real> sheen_tint;
    Texture<Real> clearcoat;
    Texture<Real> clearcoat_gloss;

    Real eta;
};
```

You need to implement the following three functions in `materials/disney_bsdf.inl`:

```
Spectrum eval_op::operator()(const DisneyBSDF &bsdf) const;
Real pdf_sample_bsdf_op::operator()(const DisneyBSDF &bsdf) const;
std::optional<BSDFSampleRecord> sample_bsdf_op::operator()(const DisneyBSDF &bsdf) const;
```

For importance sampling, ignore the sheen component since it is relatively weak compared to other lobes. Randomly choose between a diffuse lobe, a metal lobe, a clearcoat lobe, and the glass lobes based on the following weights:

$$\begin{aligned} \text{diffuseWeight} &= (1 - \text{metallic}) \cdot (1 - \text{specularTransmission}) \\ \text{metalWeight} &= (1 - \text{specularTransmission} \cdot (1 - \text{metallic})) \\ \text{glassWeight} &= (1 - \text{metallic}) \cdot \text{specularTransmission} \\ \text{clearcoatWeight} &= 0.25 \cdot \text{clearcoat} \end{aligned} \tag{22}$$

However, when the ray is from inside the object ($\omega_{\text{in}} \cdot n_g \leq 0$), we set all weights to 0 and only leave the glass lobes (note that glass still both reflect and refract).

When implementing, add one component at a time. Don't rush and put everything in there at once. Do sanity check with only enabling one component at a time. Note that even if everything other than the base color is set to zero, there is still a specular component from the dielectric specular response of the modified metal lobe.

Also recall from homework 0: you might need to rescale your random number w for selecting reflection/refraction.

Try out the scene `scenes/disney_bsdf_test/disney_bsdf.xml` to see how the material look like. Play with the parameters.

Store your rendering of the `disney_bsdf` and the `disney_bsdf_array` scenes in

`images/disney_bsdf.exr`

Task (10%). Render an image that is not in the test scenes with your Disney BSDF! Play with textures and geometry to make it look interesting. Use the Mitsuba-Blender add-on for converting the scenes (lajolla can parse the Disney BSDF parameters exported from Blender). Have fun!

References

[1] Brent Burley. Physically-based shading at Disney. In *SIGGRAPH Course*, 2012.

[2] Brent Burley. Extending the Disney BRDF to a BSDF with integrated subsurface scattering. *SIGGRAPH Course*, 19, 2015.

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

[4] Alejandro Conty Estevez and Christopher Kulla. Production friendly microfacet sheen BRDF. *SIGGRAPH Course*, 2017.

[5] Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. In *SIGGRAPH*, pages 165–174, 1993.

[6] Eric Heitz. Understanding the masking-shadowing function in microfacet-based BRDFs. *J. Comput. Graph. Tech*, 3(2):32–91, 2014.

[7] Eric Heitz. Sampling the GGX distribution of visible normals. *J. Comput. Graph. Tech*, 7(4), 2018.

[8] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and Leonard McMillan. A data-driven reflectance model. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 22(3):759–769, 2003.

[9] Michael Oren and Shree K Nayar. Generalization of Lambert's reflectance model. In *SIGGRAPH*, pages 239–246, 1994.

[10] Christophe Schlick. An inexpensive BRDF model for physically-based rendering. *Comput. Graph. Forum*, 13(3):233–246, 1994.

[11] TS Trowbridge and Karl P Reitz. Average irregularity representation of a rough surface for ray reflection. *J. Opt. Soc. Am.*, 65(5):531–536, 1975.

[12] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. *Rendering Techniques (Proc. EGSR)*, pages 195–206, 2007.